FACULTY OF ENGINEERING AND ARCHITECTURE

From Visualization to Machine Learning: Advancing Time Series Analytics on Wearable Sensor Data

Jonas Van Der Donckt

Doctoral dissertation submitted to obtain the academic degree of
Doctor of Electronics and ICT Engineering Technology

Supervisors

Prof. Sofie Van Hoecke, PhD* - Prof. Femke Ongenae, PhD**

* Department of Electronics and Information Systems
Faculty of Engineering and Architecture, Ghent University

** Department of Information Technology
Faculty of Engineering and Architecture, Ghent University

June 2025

GHENT UNIVERSITY

ISBN 978-94-6355-999-7
NUR 991, 982
Wettelijk depot: D/2025/10.500/59

Members of the Examination Board

Chair

Prof. Hennie De Schepper, PhD, Ghent University

Other members entitled to vote

Prof. Jefrey Lijffijt, PhD, Ghent University
Prof. Koen Paemeleire, PhD, Ghent University
Prof. Kristof Van Laerhoven, PhD, Universität Siegen, Germany
Prof. Steven Verstockt, PhD, Ghent University

Supervisors

Prof. Sofie Van Hoecke, PhD, Ghent University
Prof. Femke Ongenae, PhD, Ghent University

Dankwoord

"Echte kennis laat zich niet afmeten in diploma's, Joanne. Trouwens ook niet in het aantal boeken dat erin gestampt is. Laat hem de sterren zien, de planten die ontkiemen en weer verwelken, de schoonheid van een zonsondergang. Laat hem de seringen ruiken en de branding horen."

–Mélissa Da Costa. *Al het blauw van de hemel.*

Dit proefschrift vormt (voorlopig) het hoogtepunt van een academisch avontuur waarin ik bijna zes jaar lang mijn nieuwsgierigheid en leergierigheid de vrije loop kon laten. Hoewel mijn motivatie en gedrevenheid grotendeels uit mezelf kwamen, had ik dit traject nooit op zo een aangename kunnen verwezenlijken zonder de vele fijne samenwerkingen, warme steun, en het grenzeloos vertrouwen van de mensen rondom mij.

Allereerst wil ik mijn ouders en mijn familie bedanken. Jullie hebben er steeds voor gezorgd dat mijn studies en toekomst centraal stonden. Mama en papa, jullie onvoorwaardelijke steun, vertrouwen en geduld hebben mij gebracht tot waar ik nu ben; ik ben jullie daar ontzettend dankbaar voor.

Jeroen, jouw doorzettingsvermogen heeft duidelijk op mij afgegeven. Ik ben enorm dankbaar voor onze fijne en intense samenwerking, en voor de vele mooie (conferentie)reizen die we samen beleefd hebben. Het was de perfecte mix tussen hard werken en grappen & grollen, wat alles zo moeiteloos deed aanvoelen. Ik hoop uit het diepst van mijn hart dat we in de toekomst nog vaak zo intensief mogen samenwerken, want het zijn zonder twijfel de mooiste momenten uit mijn doctoraat.

Louise, dankjewel voor jouw onvoorwaardelijke steun gedurende de afgelopen vier en een half jaar, zelfs op de momenten waarop ik lange dagen maakte en minder aangenaam gezelschap was. Jouw warmte, geduld en zelfs jouw (soms overdreven) trots betekenen veel meer voor me dan je je kunt voorstellen. Jij herinnert mij telkens opnieuw aan wat er écht telt in het leven. Ik hoop dat ik evenveel voor jou kan betekenen als jij steeds voor mij doet.

Wilco en Pitou, bedankt om af en toe op mijn toetsenbord te gaan liggen en mij de nodige afleiding te geven (al was dat soms iets te vaak). Jullie aanwezigheid maakt telewerken een stuk aangenamer.

Emiel, in onze studententijd heb je mijn leergierigheid en gedrevenheid tot kritisch denken meer naar boven gebracht, waar het hoogtepunt bereikt werd toen we op eenzelfde dag GEMS in ijsboerke-dozen, zowel in de diepvries, oven, als in een bos

geplaatst hebben. Ik houd goede herinneringen over aan onze late-night talks en kooksessies. Onze trektochten in de bergen waren de perfecte aanzet om mijn sportieve natuurnieuwsgierigheid verder aan te wakkeren. Het was daarom ook zo heerlijk om jou ook als collega op de bureau te hebben. Ik hoop oprecht dat onze avonturen hier niet stoppen en we nog vele mooie momenten samen mogen beleven.

Dit proefschrift zou niet tot stand gekomen zijn zonder de vruchtbare interdisciplinaire samenwerking van de afgelopen jaren. De laatste hoofdstukken in dit boek kwamen voort uit een bijzonder fijne samenwerking met het departement neurologie van het UZ Gent. Nicolas, ondanks jouw drukke agenda vond je steeds de tijd en het geduld om samen diep in de mBrain-resultaten te duiken. Je hebt mij getoond hoe natuurlijk en inspirerend interdisciplinair samenwerken kan zijn wanneer mensen gepassioneerd zijn door hun vak. Professor Koen Paemeleire, dank u voor de boeiende gesprekken waarin we resultaten grondig bespraken en nieuwe hypothesen ontwikkelden. Uw inzichten waren van onschatbare waarde bij het uitvoeren van de vele analyses op de mBrain-dataset. Aan het hele mBrain-IDLab team—Bram, Mathias, Vic, en Marija—dank jullie wel voor de fijne samenwerking. Het was een uitdaging om de brug te slaan tussen technologie, artsen en patiënten, maar onze motivatie zorgde ervoor dat we die kloof konden verkleinen.

Met een voldaan gevoel kijk ik ook terug op de samenwerking met het GHEP-labo van professor Marie-Anne Vanderhasselt. Hoewel deze bijdragen uiteindelijk niet in dit boek terechtgekomen zijn, heeft onze samenwerking geleid tot meerdere mooie publicaties waar ik met trots op terugkijk. Marie-Anne, bedankt dat ik mocht proeven van de boeiende wereld van de (klinische) experimentele psychologie en kennis kon maken met jullie vele fascinerende paradigma's en meetmethodes. Mitchel, jouw heldere communicatie en indrukwekkende gedrevenheid waren bijzonder inspirerend. Ik wens je alle succes toe in de Verenigde Staten, en heb er alle vertrouwen dat je talent daar ten volle bloei zal komen. Cédric, jij bouwt verder op deze studies en probeert onze inzichten toe te passen in uitdagende real-world omgevingen. Ik ben dankbaar dat ik je (zij het beperkt) mocht begeleiden—onze meetings zijn steeds to-the-point en efficiënt.

Aan mijn huidige bureaugenoten Diego, Marija, Emiel, Julie, Bert, Deressa, Glenn en Hannes: bedankt voor de bijzonder fijne tijden in het IDLab-Media bureau—het sneeuwloopje blijft een absoluut hoogtepunt.

Ook de (in mijn ogen) oudere garde van (PreDiCT-)IDLab—Gilles, Martijn, Bram, Sander, Dieter, Jelle, Matthias, Mathias, Colin, Pieter, Emile, Nathan, David, Jarne, Thomas, Thibault, Sandeep, Vic, Jennifer, Stijn, Mohammed, Tom en Stef—dank jullie wel. De vele praatjes aan de koffiemachine, cluster-meetings en IDLab-evenementen zijn stuk voor stuk mooie herinneringen. Ik heb ervan genoten intensief met velen van jullie samen te werken en jullie (misschien iets te fanatiek) de tools van mij en Jeroen op te dringen.

Aan de recentere collega's—Arne, Cédric, Jef, Kyana, Stefanie, Ayko, Ayah, Ahmed, Emma, Warre—ook jullie bedankt voor de aangename gesprekken. Jullie gedrevenheid geeft me alle vertrouwen in de toekomst van team PreDiCT.

Dank aan voorzitter professor Hennie de Schepper en de juryleden van mijn doctoraatscommissie—professor Jefrey Lijfijt, professor Koen Paemeleire, professor

Kristof Van Laerhoven en professor Steven Verstockt. Jullie constructieve suggesties in de leesverslagen hebben dit boek mee naar een hoger niveau getild. De uitdagende vragen tijdens de interne verdediging zag ik als het summum van mijn werk, en vond dit dus ook een eer om dit voor jullie te mogen afleggen.

Professor Kris Demuynck, bedankt voor het potentieel dat u in mij zag, alsook de aangename begeleiding samen met Brecht en Jenthe tijdens mijn masterthesis. Jullie technische kennis en enthousiasme wakkerden mijn passie voor onderzoek verder aan.

Tot slot wil ik mijn twee promotoren bedanken: professor Femke Ongenae en professor Sofie Van Hoecke. Femke, dankzij de gezellige sfeer in het iGent-kantoor—samen met Bram, Kyana, Stijn, Femke (De Backere) en Philip,—werd iedere werkdag daar bijzonder aangenaam. Jouw vermogen om op exotische tijdstippen nauwkeurig werk af te leveren, blijft bewonderenswaardig. Bovendien zorgen jouw positiviteit en schaterlach ervoor dat ik mij altijd veilig voelde om eventuele problemen te delen met jou, bedankt hiervoor. Sofie, bedankt om Jeroen en mij de kans te geven om ons verder te ontwikkelen in de stimulerende PreDiCT-IDLab omgeving. Ik ben je bijzonder dankbaar voor de vrijheid die je mij gaf om mijn eigen accenten te kunnen leggen, zowel in mijn doctoraatsonderzoek als in dit boek. Ik bewonder hoe jij, samen met Femke, erin slaagt om zo'n grote groep op een coherente manier aan te sturen, en hoe je een breed scala aan taken weet te combineren. Jouw grenzeloze toewijding en betrokkenheid maken van team PreDiCTeen unieke en warme plek waar ik met veel plezier deel van mocht uitmaken.

Gent, juni 2025
Jonas Van Der Donckt

Table of Contents

List of Acronyms

Symbols

AI^{ABS} absolute activity index.

A

Acc accelerometer.

ACF autocorrelation.

AEE activity energy expenditure.

AI activity index.

AI artificial intelligence.

ApEn approximate entropy.

ASAP automatic smoothing for attention prioritization.

B

BVP blood volume pulse.

C

CCH chronic cluster headache.

CH cluster headache.

CNN convolutional neural network.

CV cross-validation.

D

DCT discrete cosine transform.

DEBS ACM international conference on distributed and event□based systems.

DL deep learning.

E

ECG electrocardiogram.

ECH episodic cluster headache.

EDA electrodermal activity.

EDA exploratory data analysis.

EEG electroencephalography.

EMA ecological momentary assessment.

EMG electromyography.

ETF exchange-traded fund.

F

FPCS feature-preserving compensated sampling.

G

Gyr gyroscope.

H

HAR human activity recognition.

HR heart rate.

HUNT Nord-Trøndelag health survey.

I

IBI interbeat interval.

ICHD-3 international classification of headache disorders, third edition.

IGM4 inter-pixel gradient-based M4.

L

LLB longest line bucket.

LMM linear mixed (effect) model.

LOSO leave-one-subject-out.

LR-swapping left-right swapping.

LSTM long short-term memory.

LTOB largest triangle one bucket.

LTTB largest triangle three buckets.

M

MAE mean absolute error.

MAEP mean absolute error at the peaks.

Mag magnetometer.

MDD major depressive disorder.

ML machine learning.

MOH medication overuse headache.

MSE mean squared error.

MV majority voting.

N

NN neural network.

O

OOF out-of-fold.

P

PA physical activity.

PIP perceptually important points.

PPG photoplethysmography.

PSD power spectral density.

PSG polysomnography.

PSNR peak signal to noise ratio.

R

RDP Ramer–Douglas–Peucker.

RL reinforcement learning.

S

SD standard deviation.

SHL Sussex-Huawei locomotion-transportation.

SMV signal magnitude vector.

SotA state-of-the-art.

SQI signal quality index.

SSIM structured similarity index measure.

SSL self-supervised learning.

STFT short-time Fourier transform.

SVM support vector machine.

T

TTH tension-type headache.

U

UCR University of California, Riverside.

UL-pairing upper-lower limb pairing.

V

VW Visvalingam–Whyatt.

Samenvatting

De voorbije decennia hebben dalende kosten voor sensoren, dataopslag en rekenkracht geleid tot een sterke toename van sensortechnologieën. Deze sensoren worden vandaag op grote schaal ingezet voor het monitoren van industriële processen, omgevingsparameters, en fysiologische kenmerken van mensen. Ze registreren vaak meerdere variabelen aan hoge frequenties—tot wel 100 metingen per seconde—wat resulteert in enorme hoeveelheden tijdreeksdata.

Een belangrijk subdomein binnen deze technologie zijn draagbare sensoren of wearables, zoals smartwatches, die continue lichaamsgerichte monitoring mogelijk maken. Doordat wearables rechtstreeks op het lichaam worden gedragen, verzamelen ze data op een niet-intrusieve manier. Dankzij verbeterde rekenefficiëntie en batterijduur kunnen wearables voor steeds langere periodes data verzamelen, wat ze bijzonder geschikt maakt voor langdurige monitoring in alledaagse omgevingen. Zo kunnen ze bijvoorbeeld objectief analyseren hoe medische aandoeningen—zoals primaire hoofdpijnaandoeningen—het dagelijks functioneren van patiënten beïnvloeden.

Ondanks de toenemende beschikbaarheid van grootschalige tijdreeksgegevens uit wearables, blijft het uitdagend om inzichten te halen uit die data. In tegenstelling tot tekst of afbeeldingen, die mensen intuïtief kunnen interpreteren, ontbreekt bij tijdreeksgegevens directe interpreteerbaarheid. Bovendien is het vaak niet duidelijk of de gemeten signalen (bijvoorbeeld bewegingsdata) daadwerkelijk de gewenste informatie (zoals dagelijkse activiteiten) accuraat vastleggen. Relevante patronen kunnen namelijk beperkt zijn tot kleine delen van de tijdreeks, verborgen zitten in complexe, niet-lineaire relaties tussen verschillende signalen, of een combinatie van beide vormen.

Visualisatietechnieken vormen een krachtig hulpmiddel om deze uitdagingen aan te gaan. Door het weergeven van ruwe signalen of het benadrukken van tussentijdse signaaltransformaties, ondersteunen ze de initiële data-analyse en helpen ze bij het debuggen van signaalverwerkingsstappen.

Hoewel geavanceerde visualisaties kunnen helpen bij het interpreteren van (grote) tijdreeksen, wordt hun effectiviteit uiteindelijk begrensd door de kwaliteit van de onderliggende gegevens. Monitoring via wearables in alledaagse omgevingen brengt specifieke uitdagingen met zich mee—zoals ontbrekende datasegmenten, signaalartefacten, of gebruikersfouten (bijvoorbeeld een verkeerd gedragen wearable)—die de analytische resultaten aanzienlijk verstoren. Daarom zijn strikte procedures voor dataverwerking en kwaliteitscontroles essentieel om betrouwbare en betekenisvolle inzichten uit deze databronnen te verkrijgen.

Tegenwoordig is deep learning (DL) de dominante aanpak geworden binnen machine learning (ML) toepassingen met wearable data, terwijl meer klassieke technieken

minder aandacht krijgen. Hoewel DL-modellen uitblinken in het extraheren van complexe, hoog-dimensionale patronen, zijn ze vaak niet de meest efficiënte of interpreteerbare oplossing. Daarom blijft het zinvol om traditionele ML-methodes grondig te evalueren, zodat modelkeuzes beter afgestemd kunnen worden op de specifieke doelen, beperkingen en karakteristieken van de data.

Dit doctoraat richt zich op drie belangrijke uitdagingen binnen de analyse van wearable tijdreeksdata: (1) het schaalbaar visualiseren van grote datasets, (2) het evalueren van klassieke ML-technieken voor activiteitsherkenning met wearables, en (3) het verbeteren van datakwaliteit en betrouwbaarheid bij analyses van wearable monitoringstudies.

Hoofdstuk 1 biedt de nodige achtergrond voor de bijdragen die in de volgende hoofdstukken worden gepresenteerd. Het situeert tijdreeksdata binnen het bredere domein van data-analyse, met bijzondere aandacht voor de inherente heterogeniteit van dit datatype. Vervolgens worden de beperkingen van bestaande visualisatietechnieken voor tijdreeksen besproken en worden mogelijke onderzoekslijnen uitgezet. Zo wordt voornamelijk gefocust op het potentieel van *shape-preserving* data-aggregatie algoritmes om schaalbaarheidsproblemen aan te pakken. Daarna worden de mogelijkheden en uitdagingen van draagbare sensortechnologie in *real-world* monitoring besproken, toegepast op patiënten met primaire hoofdpijnaandoeningen. Tot slot wordt er een overzicht gegeven van ML-methodologieën, met bijzonder aandacht voor traditionele technieken, zoals *multi-resolution feature extraction*.

Hoofdstuk 2 focust op de eerste uitdaging: het schaalbaar maken van tijdreeksvisualisaties. Hiervoor wordt Plotly-Resampler voorgesteld, een productieklare uitbreiding van de `plotly.py` visualisatietool met een configureerbare interface voor dynamische data-aggregatie. Door tijdreeksdata te aggregeren vóór het renderen, vermindert Plotly-Resampler het aantal datapunten dat naar de front-end gestuurd wordt, wat leidt tot snellere weergave en lagere netwerkbelasting. Bij interactie met de visualisatie, zoals zoomen, wordt automatisch een nieuwe aggregatie berekend voor het geselecteerde bereik, wat een vloeiende *detail-on-demand* verkenning mogelijk maakt. Dankzij een geheugen-efficiënte implementatie vermijdt Plotly-Resampler overbodige datakopieën. De meer dan tien miljoen installaties tonen de grote nood aan schaalbare visualisatieoplossingen. Daarnaast biedt Plotly-Resampler een intuïtieve interface die interactieve verkenning van bestaande en nieuwe aggregatie algoritmes vergemakkelijkt, wat het zeer aantrekkelijk maakt voor onderzoekers binnen dit domein.

In Hoofdstuk 3 worden bestaande *shape-preserving* aggregatie-algoritmen systematisch geëvalueerd op visuele representativiteit en stabiliteit, met duidelijke richtlijnen voor praktisch gebruik. Hierbij blijken algoritmes die verticale extrema benadrukken, zoals Ramer–Douglas–Peucker (RDP), MinMax en largest triangle three buckets (LTTB), bijzonder effectief. Om reproductie en verder onderzoek mogelijk te maken, is de evaluatiecode open-source beschikbaar gesteld.

Appendix A bouwt verder op de inzichten uit Hoofdstuk 3, waar verticale extrema belangrijk bleken voor zowel representativiteit als stabiliteit. Het behandelt de schaalbaarheidsbeperkingen van het LTTB-algoritme door een MinMax-preselectiestap te introduceren die de zoekruimte vooraf beperkt. Deze optimalisatie maakt een gedeeltelijke parallellisatie mogelijk en resulteert in een snelheidswinst van $\pm 30\times$ ten opzichte

van de originele implementatie. Dit verbeterde algoritme is momenteel de standaard configuratie in Plotly-Resampler.

De tweede onderzoekslijn richt zich op het evalueren van de effectiviteit van traditionele ML technieken voor activiteitsherkenning op basis van wearable data. Appendix B introduceert `tsflex`, een Python bibliotheek ontworpen voor efficiënte feature-extractie op heterogene tijdreeksgegevens. In tegenstelling tot veel bestaande tools maakt `tsflex` geen impliciete aannames over de regelmaat of continuïteit van de data, waardoor het robuust is voor zowel ontbrekende en irreguliere data. Bovendien ondersteunt het multi-window (of multi-resolutie) feature-extractie, een techniek die tot op heden weinig aandacht kreeg.

Hoofdstukken 4 en 5 bespreken de inzichten uit twee ML-wedstrijden rond activiteitsherkenning op basis van bewegingsdata van wearables waarbij de `tsflex`-toolkit werd ingezet. In beide gevallen bleek grondige data-analyse cruciaal voor het detecteren van distributie- en oriëntatieverschillen tussen de train- en testdata. Om deze distributie verschillen aan te pakken, werden nieuwe augmentatiestrategieën ontwikkeld op zowel signaal- als featureniveau. Onze rotatie-invariante feature-aggregatie bleek hierbij systematisch beter te presteren dan de gangbare signal magnitude vector (SMV)-transformatie—wat zowel het potentieel van onze methode als de beperkingen van SMV aantoont. In Hoofdstuk 5 wordt bovendien aangetoond dat multi-window features de expressiviteit van de featurevectoren kunnen verhogen. Onze aanpak behaalde in beide competities de eerste plaats en overtrof DL-gebaseerde methoden. Deze resultaten bevestigen dat klassieke ML, in combinatie met degelijke data-analyse en doordachte feature-engineering, nog steeds uiterst performant kan zijn voor tijdreeksclassificatie.

Hoofdstuk 6 richt zich op derde en laatste uitdaging: het verbeteren van data-analyse in *real-world* wearable monitoringstudies. Dit hoofdstuk bouwt op mijn directe betrokkenheid bij de *mBrain*-studie, een wearable-gebaseerde observatiestudie bij patiënten met primaire hoofdpijnaandoening, uitgevoerd via een samenwerking tussen de afdeling Neurologie van het UZ Gent en IDLab. Tijdens zowel de dataverzameling als analyse kwamen verschillende problemen rond datakwaliteit aan het licht. Daarom worden in dit hoofdstuk concrete strategieën aangereikt om hiermee om te gaan, met een sterke focus op visualisatiegerichte aanpakken en praktische codevoorbeelden. Deze worden geïllustreerd aan de hand van twee *real-world* datasets, waaronder de *mBrain*-dataset.

Hoofdstukken 7 en 8 bouwen verder op de *mBrain*-dataset, die zowel wearable-data als informatie over hoofdpijnaanvallen verzamelt via een digitale applicatie. Hoofdstuk 7 richt zich op patiënten met clusterhoofdpijn, en analyseert bewegingspatronen tijdens aanvallen en vergelijkt deze met hoofdpijnvrije periodes. Hoofdstuk 8 bestudeert migrainepatiënten, met meer aandacht voor symptomen en kenmerken van de hoofdpijnaanvallen zelf. Uit beide studies komt een belangrijk inzicht naar voren: er is geen eenduidige relatie tussen klinische hoofdpijnkenmerken en bewegingsgedrag in het dagelijks leven. Daarnaast bemoeilijkt het gebruik van acute en preventieve medicatie de interpretatie van de bewegingspatronen. Beide hoofdstukken leverden zowel methodologische inzichten op over het analyseren van *real-world* data, als eerste bevindingen over de relatie tussen hoofdpijnkenmerken en gedragsdata.

Tot slot vat Hoofdstuk 9 de voornaamste onderzoeksbevindingen samen, reflecteert

het op de bredere impact ervan, en wordt er een toekomstvisie gegeven voor verder onderzoek.

Samenvattend draagt dit doctoraat bij aan het domein van wearable tijdreeksdata-analyse door uitdagingen aan te pakken op vlak van schaalbare visualisatie, machine-learning gebaseerde activiteitsherkenning, en kwaliteitsanalyse bij *real-world* data-analyse. Door de ontwikkeling van nieuwe tools, grondige empirische evaluaties en praktische toepassingen, tonen we aan hoe een doordachte balans tussen klassieke en moderne methodes kan leiden tot robuuste inzichten uit complexe wearable-data. Naarmate deze technologieën verder evolueren, zal ook de scope van wearable data-analyse uitbreiden, met nieuwe kansen en ook uitdagingen op vlak van datakwaliteit, interpretatie en maatschappelijke relevantie. Voortgezet onderzoek op de intersectie tussen visualisatie, machine learning en data-analyse zal cruciaal zijn om het volledige potentieel van deze sensortechnologie te realiseren, zowel in klinische context als daarbuiten.

Summary

Over the past decades, decreasing costs in sensor production, computing power, and data storage have led to a surge in ubiquitous sensing devices. These devices are now widely used to monitor industrial processes, environmental conditions, and human physiology. Typically, they capture multiple data modalities at moderate to high sampling rates, generating large volumes of time series data over extended periods.

Wearable technologies, such as smartwatches, form a specialized subset of ubiquitous devices designed for continuous, body-centric monitoring. Advancements in computational efficiency and battery capacity have enabled prolonged, unobtrusive data collection, making wearables particularly well-suited for longitudinal, real-world monitoring. For example, they can be used to objectively evaluate how medical conditions—such as primary headache disorders—affect patients' daily activities.

Despite the increasing availability of large-scale time series data from wearable technologies, extracting actionable insights remains challenging. Unlike text or images, which humans can intuitively interpret, raw time series sensor data lacks immediate comprehensibility. Moreover, it is often unclear whether the available data (e.g., wearable movement data) captures the intended information (e.g., daily life activity). Relevant patterns for this desired objective may be highly localized, embedded in complex nonlinear relationships across multiple modalities, or arise from a combination of both.

Visualization techniques offer a powerful means for mitigating these challenges associated with time series interpretation. By presenting raw data, highlighting intermediate signal transformations, or illustrating the sequential predictions of machine learning models, visualizations support exploratory data analysis, facilitate debugging processing pipelines, and enhance model validation.

While advanced visualizations can aid researchers in interpreting (large) time series, their effectiveness is ultimately constrained by the quality of the underlying data. In real-world wearable monitoring, issues such as missing data segments, sensor drift, signal artifacts, and user noncompliance (e.g. incorrect attachment of the wearable on the body) are common and can significantly distort analytical outcomes. As a result, rigorous data cleaning, preprocessing, and quality assurance procedures are essential to ensure that insights drawn from wearable datasets are both reliable and meaningful.

Meanwhile, deep learning (DL) has emerged as a dominant approach for addressing a variety of wearable-based machine learning (ML) tasks, often overshadowing traditional methods. While DL models are highly effective at capturing complex, high-dimensional features, they are not always the most efficient or interpretable choice for every use case. As such, revisiting and re-evaluating established techniques remains

valuable to ensure that model selection aligns with the specific goals, constraints, and characteristics of the data.

This dissertation tackles three central challenges in wearable time series data analysis: (1) scaling line chart visualization techniques to handle large time series datasets effectively, (2) reevaluating the suitability of traditional machine learning techniques (for activity recognition using wearable sensors), and (3) improving the quality and reliability of data analytics applied to real-world wearable-monitoring studies.

Chapter 1 establishes the necessary background for the contributions presented in the subsequent chapters. Specifically, it first situates time series data within the broader landscape of data analytics, emphasizing its inherent heterogeneity. Next, the limitations of existing time series visualization approaches are delineated, and research directions are outlined. Particularly, the potential of shape-preserving data aggregation methods—such as shape-preserving subsampling algorithms—are outlined to overcome scalability issues. This is followed by discussing the challenges and potential of wearable sensing devices, focusing on real-world wearable monitoring studies, particularly in the context of primary headache disorders. Lastly, this chapter provides an overview of the machine learning methodologies for time series data, highlighting the challenges posed by its heterogeneity and the uncertainty regarding whether a given dataset sufficiently captures the intended objective. It also underscores how traditional machine learning techniques, particularly multi-resolution feature extraction, are often overlooked in favor of more complex deep learning approaches.

Chapter 2 tackles the first challenge—scaling time series visualization techniques—by introducing Plotly-Resampler, a production-ready library that extends `plotly.py` with a configurable interface for dynamic data aggregation. By subsampling time series data before rendering, Plotly-Resampler reduces the number of data points sent to the front-end, significantly lowering network latency and improving rendering performance. When users interact with the visualization (e.g., by zooming), a new aggregation is computed for the selected range, ensuring detail-on-demand-based data exploration. Importantly, Plotly-Resampler is designed to be memory-efficient, avoiding bottlenecks caused by redundant data copies. With over ten million installations, its widespread adoption underscores the demand for scalable time series visualization tools. Additionally, Plotly-Resampler provides an intuitive interface that facilitates interactive exploration of new and existing time series aggregation algorithms.

Chapter 3 expands on this last remark by systematically evaluating and comparing the most prominent shape-preserving subsampling algorithms, which are widely used to maintain the scalability and interpretability of time series visualizations. Two core aspects are analyzed, (1) visual representativeness, which assesses how well an aggregated line chart preserves the visual appearance of non-aggregated version, and (2) visual stability, which evaluates the consistency of selected data points when minor range changes occur, such as small zooms or panning are applied—crucial for streaming-based applications. Both aspects are thoroughly analyzed across multiple time series datasets, leading to practical guidelines for selecting appropriate subsampling techniques. Generally, capturing vertical extrema is essential for good visual representativeness, with the Ramer–Douglas–Peucker (RDP), MinMax, and largest triangle three buckets (LTTB) algorithms offering the highest data efficiency. To promote reproducibility and

further research, the evaluation framework and experiments are made available in an open-source repository, allowing other researchers to assess new aggregation methods.

Appendix A builds on insights from Chapter 3, recognizing that vertical extrema are crucial for both visual representativeness and stability. It addresses the scalability limitations of the LTTB algorithm by introducing a MinMax-preselection step, which reduces the search space before sampling. This optimization enables parallelization, achieving a 30× speedup compared to the original LTTB algorithm. This improved algorithm is currently the default configuration in Plotly-Resampler.

Our second objective is concerned with analyzing the efficacy of traditional machine learning for wearable-based human activity recognition. Appendix B introduces `tsflex`, a Python library designed for efficient feature extraction from heterogeneous time series data. Unlike many existing tools, tsflex does not make implicit assumptions about data regularity or continuity, enabling robust processing even when data contains gaps. Additionally, it supports multi-window feature extraction, a technique that has received limited attention in previous research.

Chapter 4 and 5 outline the insights gained from our participation in two machine learning competitions focused on human (transportation) activity recognition using wearable movement data, where the tsflex toolkit was leveraged. In both challenges, extensive data analysis revealed distributional and orientational discrepancies between training and test data. To address these issues, we devised novel augmentation strategies at both the signal and feature levels. Notably, our rotation-invariant feature aggregation approach consistently outperformed the commonly used signal magnitude vector (SMV) transformation—highlighting not only the potential of our method but also the limited representational capacity of the SMV approach. Furthermore, in Chapter 5, we leveraged multi-window size features to enhance the expressiveness of our feature representations. These feature vectors were then used as input to traditional machine learning models.

Our approaches secured first place in both competitions, significantly outperforming deep learning-based methods. These results demonstrate that, with rigorous data analysis and well-engineered features, traditional machine learning remains a powerful tool for time series classification—directly contributing to our second research objective.

Chapter 6 shifts the focus to our final challenge—enhancing data analytics in real-world wearable monitoring studies. This challenge arose from my direct involvement in the *mBrain* study, a wearable-based observational study on patients diagnosed with primary headache disorders, conducted through an interdisciplinary collaboration between the neurology department of Ghent University Hospital and IDLab. During both data collection and analysis, several challenges in data quality were noted. Consequently, this chapter identifies and addresses key issues in wearable data analysis, including participant compliance monitoring, wearable usage behavior analysis, and handling missing or artifact-contaminated data, by using visualization-oriented approaches, guidelines, and practical code examples. These challenges are illustrated using two real-world wearable datasets, including the *mBrain* dataset.

Chapter 7 and 8 further leverage the *mBrain* dataset, which combines wearable data from patients with primary headache disorders and headache attack information

collected via a digital headache application. Chapter 7 focuses on cluster headache patients, analyzing movement patterns during headache episodes compared to headache-free periods. Similarly, Chapter 8 examines migraine patients, placing greater emphasis on associated symptoms and headache attack characteristics. Across both studies, a key insight emerges: there is no straightforward one-to-one relationship between clinical headache characteristics and real-world movement behavior. Moreover, participants' use of acute and preventive treatments further complicates the interpretation of wearable-based movement patterns. These findings contributed both methodological advances for extracting insights from real-world data and initial results on the relationship between clinical headache characteristics and wearable-based movement behavior, concluding the dissertation's contributions to this objective.

To conclude, Chapter 9 presents a summary of our main research findings and other achieved impacts through the above chapter contributions, and a perspective on future work that could be done in these areas.

In summary, this dissertation advances the field of wearable time series data analysis by addressing key challenges in scalable visualization, machine learning-based activity recognition, and real-world data quality assurance. Through the development of new tools, rigorous empirical evaluations, and practical applications to real-world datasets, it demonstrates how a careful balance between traditional and modern approaches can yield robust insights from complex wearable data. As wearable technologies continue to evolve, the scope of wearable data analytics will expand accordingly, bringing new opportunities—and challenges—in ensuring data reliability, interpretability, and relevance. Continued research at the intersection of visualization, machine learning, and real-world data analysis will be crucial to fully realize the potential of ubiquitous sensing in both clinical and broader societal contexts.

1
Introduction

Time series comprise sequences of data points recorded at successive time intervals, making it a fundamental data type across many domains and applications where measurements evolve over time [1]. Common examples include stock closing prices, historical weather data, heart rate measurements, power consumption levels, and motor vibration data.

Among these domains, wearable devices have emerged as a prominent source of time series data. From smartwatches and fitness trackers to medical-grade sensors, these devices continuously monitor various physiological and contextual signals. Their ubiquity offers immense potential for personalized health monitoring, activity recognition, and behavioral analysis [2].

Despite its widespread availability, analyzing time series data remains challenging due to its inherent complexity and heterogeneity. For instance, time series datasets often contain multiple data modalities, each potentially recorded at different—and sometimes irregular—intervals, leading to less-structured and asynchronous formats that complicate standard analysis methods. Furthermore, some time series data is captured at high sampling rates, such as at 1 kHz for electrocardiogram (ECG) recordings, generating 3.6 million data points per hour. This high data density increases computational demands and makes it more challenging to detect meaningful patterns within large volumes of data.

Another major challenge is that time series data is often difficult for humans to interpret intuitively. Unlike images or text, which we encounter and make sense of in everyday life, raw time series signals are less accessible to human perception [3]. For example, understanding wearable movement data, i.e., accelerometer, is complicated

by multiple interacting factors, including sensor placement (e.g., chest vs. wrist), contextual variables (e.g., the surface on which a user is walking), and the complex, multi-joint, multi-planar nature of human motion. Crucially, these variables are not always fully captured in the raw measurements, making interpretation even more challenging.

This PhD dissertation addresses key challenges in time series data analysis, with a particular focus on wearable data. The work is organized around three primary contributions. First, we address the scalability of line-chart visualization, designing and optimizing techniques for efficiently exploring large time series datasets and extracting meaningful insights. Second, we highlight the often overlooked relevance of traditional machine learning (ML) approaches for time series analysis. By leveraging our enhanced visualization techniques, with an efficient and flexible framework for time series processing and feature-extraction, we participate in two wearable human activity recognition (HAR) competitions. In both contests, a combination of visual analytics and classical ML leads to the best-performing solutions. Third, we propose optimized methods for analyzing highly heterogeneous longitudinal time series data collected from wearable devices in real-world settings. These methods address key challenges such as missing data and signal artifacts. Their practical utility is validated through an interdisciplinary, self-conducted real-life headache monitoring study, where they enhance both data collection methodologies and the analysis of wearable movement patterns.

Before exploring the specific contributions of this dissertation, it is essential to establish relevant foundational knowledge. Therefore, the remainder of this chapter provides a high-level overview of data analytics, structured around three areas: (i) time series analysis, emphasizing visualization, processing, and feature extraction; (ii) ubiquitous devices, particularly wearable technologies, and their role in real-world (headache) monitoring studies; and (iii) ML methodologies, particularly applied to wearable-generated time series data.

1.1 Data Analytics

Data analytics is the systematic process of examining data to extract meaningful insights, identify patterns, and support decision-making. In time series analysis, it can enable us to understand temporal patterns, detect anomalies, forecast future events, and even recommend optimal actions. Data analytics is typically categorized into four main types: Descriptive, Diagnostic, Predictive, and Prescriptive Analytics [4]. Each type builds upon the previous one, progressing from understanding past events to making data-driven recommendations for future actions. In this section, we define each type and illustrate its application using wearable movement data, e.g., via accelerometer data of a smartwatch.

Descriptive Analytics focuses on summarizing and describing historical data to answer "*what*" has occurred over a specified period. For time series data, this often involves calculating measures, such as mean, variance, or trends, to offer a clear picture of past behavior. Descriptive analytics provides essential context, establishing a baseline and revealing patterns, i.e. recurring phenomena in the data, or anomalies, i.e. deviating phenomena in the data. Visualization also plays a crucial role, portraying complex data into human-friendly formats, allowing to uncover patterns missed in descriptive statistical analyses [5, 6]. Moreover, descriptive analytics can be valuable to evaluate the quality or execution of tasks performed within other categories (e.g., visual analysis of feature distributions).

> *Example*: For wearable movement data, descriptive analytics might involve summarizing activity levels over time to reveal trends, such as changes in movement intensity throughout the day.

Diagnostic Analytics goes further by identifying the underlying causes of observed patterns or anomalies, seeking to answer "*why*" something happened. Diagnostic analytics often involve finding correlations, identifying patterns, or contributing factors that may explain trends or outliers. While some of these methods can suggest causality, they do not confirm it.

> *Example*: For wearable movement data, diagnostic analytics could explore the reasons for a sudden drop in movement on specific days. Analyzing associated data, such as weather conditions or health-related events, can help identify factors that potentially contributed to these changes.

Predictive Analytics involves constructing models to *predict* future events or behavior. Although generally aimed at predicting future outcomes, predictive analytics is often applied to *detect* (quantify or classify) past or ongoing events [7]. A broad range of machine learning (ML) models and statistical techniques can be used, from traditional models like linear regression, tree-based approaches, and clustering methods to advanced deep learning (DL) approaches. A key assumption in predictive analytics is that the available data is representative for the task at hand.

> *Example*: In the context of wearable movement data, predictive analytics could involve creating a model to detect an individual's activity based on current and past accelerometer data.

Prescriptive Analytics goes a step further by recommending the optimal *action* based on insights derived from predictive models. By integrating forecasting with optimization techniques, prescriptive analytics suggests potential actions and evaluates the likely outcomes of different choices, guiding decision-making. In time series contexts, this approach often involves recommending responses to projected trends to achieve desired outcomes.

> *Example*: Using wearable movement data, prescriptive analytics might recommend specific actions to increase physical activity, such as sending exercise reminders or suggesting ideal times for activity based on past behavior.

Each category has a distinct focus, yet in practice, they often intersect as insights from one can enhance another. For example, patterns (descriptive) or associations (diagnostic) revealed in data can inform models for predicting events (predictive).

In this dissertation, we mainly focus on descriptive analytics (through feature extraction and visualization), diagnostic analytics (by uncovering associations), and predictive analytics applied to time series data collected from wearable devices.

1.2 Time Series Data

In data analytics, data is typically categorized as structured or unstructured.

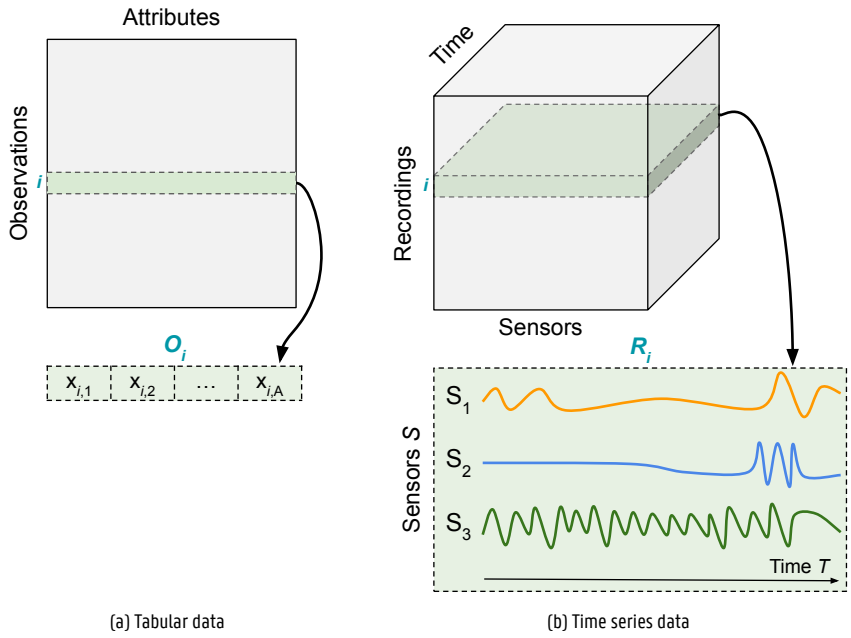
Unstructured data does not follow a consistent format and includes diverse forms such as text, images, audio, and video. Without a clear relational structure, extracting patterns and analyzing such data is more difficult. To process and interpret unstructured data, advanced techniques—such as tokenization for text or convolutions for images—are often required to impose structure and uncover meaningful insights.

In contrast, **structured data** follows a predefined format, with the most common being a "tabular structure", where rows represent observations O and columns correspond to attributes A associated with each observation. This representation allows structured data to be expressed as a matrix $X \in \mathbb{R}^{O \times A}$, as illustrated in Figure 1.1 (a).

The tabular format is widely adopted due to its clear organization of data into rows and columns, which facilitates efficient querying, sorting, and filtering. This structure is easily represented in various data formats, such as database tables or CSV files, making it accessible across a wide range of platforms and applications. Moreover, the tabular format aligns naturally with statistical techniques and machine learning algorithms, which rely on clear feature representations to uncover patterns and make predictions. Overall, its simplicity and versatility have made it a standard in data management and analysis.

Time series data represents a unique type of structured data, distinguished by its temporal dimension. It typically consists of multiple recordings (R), each capturing observations from one or more sources, e.g. sensors (S), with these observations ordered sequentially over time (T). Unlike traditional tabular data, which is structured as a 2D matrix (Figure 1.1 (a)), time series data can be conceptualized as a 3D matrix, denoted as $X \in \mathbb{R}^{R \times S \times T}$ (Figure 1.1 (b)). Note that this representation serves primarily as a conceptual framework for understanding the structure in time series data, rather than as an efficient data storage format. Unlike traditional structured data, which often assumes observations are independent, time series data frequently exhibits autocorrelation, where past values influence future values.

The temporal nature of time series data presents several challenges. Time series often exhibit trends, seasonality, and noise, requiring specialized techniques for preprocessing, modeling, and visualization [8]. Moreover, time series data can vary greatly in sampling frequency, resolution, and length. Some sequences can be recorded at regular



(a) Tabular data (b) Time series data

Figure 1.1: Illustration of structured data in a (a) tabular format (2D), and (b) time series format (3D).

intervals (e.g., daily stock prices), while others are collected irregularly (e.g., wearable health data recorded only upon event detection). Such inconsistencies complicate direct comparisons, require careful temporal alignment, and challenge standard data processing pipelines.

This variability makes the conceptual 3D matrix representation impractical for actual use. In reality, the varying lengths and sampling rates of individual sequences make it difficult to align them into a uniform temporal grid without either losing information (through aggressive resampling) or introducing artificial distortions.

1.2.1 Visualization

To better investigate the above-mentioned complexities associated with time series, which cannot be fully understood through summary statistics alone, visualization arises as an essential tool for identifying patterns and extracting insights. Among various visualization techniques, *line charts* have proven to be particularly effective in analyzing time series [3], serving as a strong starting point for most tasks, such as detecting outliers or observing trends. Figure 1.2 visualizes three distinct time series signals, each exhibiting different characteristics in terms of periodicity and stationarity.

To facilitate effective visual exploration, *interactive visualization* toolkits are crucial. Features like zooming, panning, and hover-based tooltips enhance the ability to analyze

time series data efficiently [3, 9].

However, traditional interactive line chart systems struggle with complex visualization tasks, such as analyzing trends across hundreds of time series or handling high-frequency longitudinal data containing millions of points. These systems often fail to render such datasets in a timely manner [10].

One approach to tackle these challenges is to improve the scalability of visualization systems by directly optimizing the rendering process. For instance, by leveraging GPU-

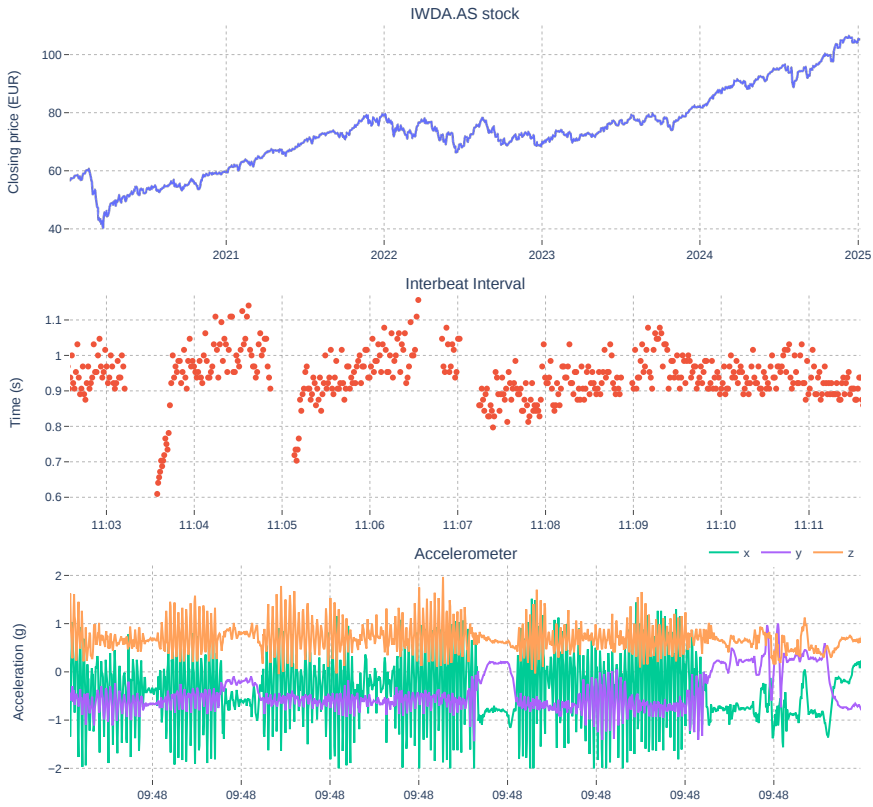


Figure 1.2: Visual representation of three independent time series data sources using the Plotly.py visualization library. The upper subplot shows the daily closing price of the IWDA.AS ETF, a regularly sampled time series with one data point for each weekday, except on weekends and bank holidays. This line chart reveals the ETF's overall trend. The middle subplot displays an excerpt of interbeat interval (IBI) data, which represents the time between successive heartbeats, using a scatter plot. Since each IBI observation is calculated as the time difference between two successive heartbeats, the data is irregularly sampled. In the lower subplot, each axis $\in \{x, y, z\}$ represents the acceleration (in g) along the corresponding dimension. More frequent and intense acceleration changes are observed across all three axes for the first 80%, while the last part shows fewer acceleration changes, suggesting reduced movement.

accelerated rendering tools, such as openGL [11], the number of data points that can be rendered increases significantly. Yet, increasing the number of rendered data points introduces new challenges related to data transfer and user interface responsiveness.

Most modern interactive visualization libraries (Plotly, Bokeh, D3.js) are web-based, primarily to maximize accessibility, ease of deployment, and cross-platform compatibility. This design choice also introduces bottlenecks: large datasets must be transferred over the network and fully loaded into the browser's memory, leading to network latency, slow rendering performance, or even browser crashes when handling very large data volumes [12].

Since front-end rendering optimizations alone cannot fully resolve these issues, data aggregation is considered to be a more effective solution [13]. Data aggregation algorithms reduce the number of points that are rendered, preserving interactivity and responsiveness by minimizing network latency and rendering time, which are both crucial for effective visual exploration [14]. Specifically, for the aggregation of a single time series line chart, the following aggregation taxonomy exists: density-wise aggregation, characteristic aggregation, and subsampling.

Density-wise aggregation or bin-count rasterization visualizes data by encoding density using a shared color scheme [17]. This process converts (time series) data into a fixed-size matrix (i.e., grid rasterization), where each element represents the bin counts at that location. An optimized implementation is available in the open-source Datashader toolkit [15]. The resulting bin-count matrix can be analyzed or rendered as an image, which represents a density-based heatmap. For example, Figure 1.3 shows such a density-based aggregation of multiple time series, generated by Datashader. While this technique is effective for visualizing single, large data modalities such as maps and point clouds, it poses challenges when there are multiple distinct modalities

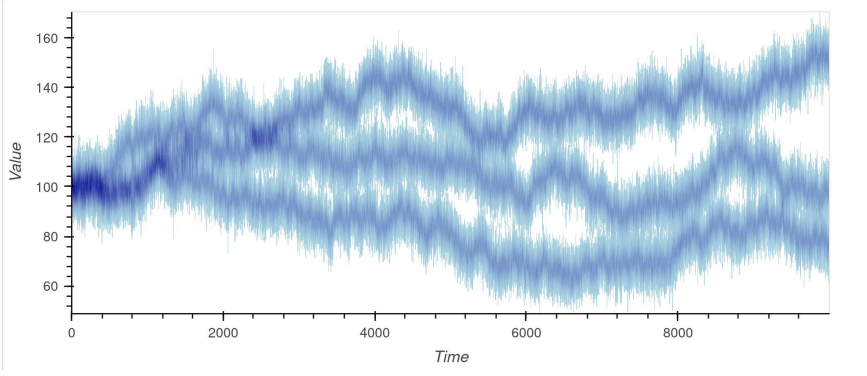


Figure 1.3: Visual representation of density-wise aggregation applied to 100 randomly generated time series signals using Datashader [15]. The Bokeh-Holoviews [16] integration enables interactive redraws of the bin-count matrix in response to user interactions such as zooming and panning.

in the data. Each modality would then require a distinguishable color density coding, making it much harder to detect patterns in the image. Moreover, converting data into images restricts differentiation and interactivity, such as hover tooltips or toggling time series traces, further reducing its effectiveness for time series visualization.

Characteristic aggregation aims to aggregate data by highlighting properties or trends, using methods such as mean, median, and smoothing. In the context of line chart visualization, Rong et al. [18] introduced automatic smoothing for attention prioritization (ASAP), which smooths time series by adaptively optimizing noise reduction and trend retention, directing users' attention towards significant deviations. Figure 1.4 provides a visual illustration of the ASAP algorithm. For the (noisy) sine and the stationary noise signal, ASAP is able to remove the noise component and illustrate the underlying deviation. The power and cinecg signal contains several (non consistent reoccurring outliers), resulting in ASAP not being able to detect patterns to smooth, and displaying the original data.

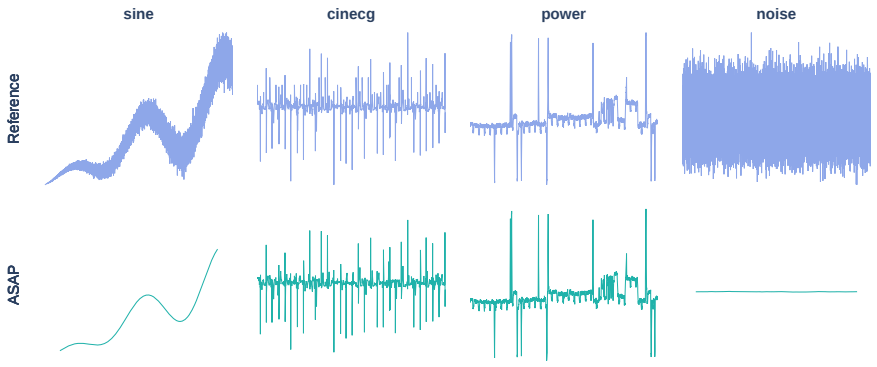


Figure 1.4: Visual comparison of the ASAP aggregation relative to the original data line chart applied to four distinct time series signals. The visualization was created using Plotly, with a line width of 2 pixels and anti-aliasing enabled. More information regarding the time series signals is provided in Section 3.3.1.1

Subsampling, in contrast to the above two categories, is a value-preserving aggregation technique that selects specific data points from the original time series [19]. A popular branch of subsampling is *shape-preserving subsampling*, which aims to retain the overall visual shape of the full-resolution line chart. To do so, shape-preserving subsampling algorithms leverage linear interpolation in the line chart visualization between adjacent selected data points.

Shape-preserving subsampling is particularly appealing because it produces a time series composed of existing data points, which supports interactive tasks such as hover tooltips and trace differentiation. At the same time, it preserves high visual fidelity, maintaining the overall appearance of the original line chart. Given these advantages, we will explore shape-preserving subsampling methods in more detail.

1.2.1.1 Shape-Preserving Subsampling Algorithms

Table 1.1 provides an overview of the most prominent shape-preserving subsampling algorithms. The first four algorithms,—EveryNth, MinMax, largest triangle three buckets (LTTB) [13], and M4 [20]—are considered to be *uniform* subsampling algorithms. These algorithms share the characteristic of selecting a fixed number of data points per sub-chunk (also referred to as bucket or bin). The number of sub-chunks, or bins, is specified by the end user.

The lower two algorithms are considered to be *non-uniform* as they iteratively select or eliminate data points from the whole data range. Both the Ramer–Douglas–Peucker (RDP) [21] and Visvalingam–Whyatt (VW) [22] algorithms stem from the cartographic domain, with the main objective of line simplification.

EveryNth, also known as uniform subsampling or decimation, is a highly efficient but rather naive algorithm that selects data points at regular intervals; i.e., every n^{th} data point from the input data array, with $n = \frac{N}{n_{out}}$. Remark that this is the only algorithm among those listed in Table 1.1 that scales linearly with n_{out} instead of N, resulting in a time complexity of $O(n_{out})$ since it only requires sampling n_{out} data points. The algorithm has a memory complexity of $O(n_{out})$, which means that it has a linear execution time and does not require additional memory beyond the input and output arrays. Although the EveryNth algorithm can be easily parallelized, multi-threading may not yield significant benefits, if any, since the only computational task here is to sample the data at exact intervals. However, this algorithm tends to have the poorest representation capabilities as it does not take the actual value of the selected samples into account.

MinMax aims to preserve the shape of time series data by selecting vertical extrema for each bucket (i.e., bin). The algorithm selects both the minimum and maximum value within each bin, but does not enforce extrema alternation. This can result in repeated neighboring minima (or maxima), potentially failing to correctly display alternations occurring in the original time series. The MinMax algorithm has a time complexity of $O(N)$, as all values of the time series must be considered once when

Table 1.1: Overview of shape-preserving subsampling algorithms, where N denotes the time series length and n_{out} represents the aggregation output size.

	Time	Memory	Parallelizable	Data points per bucket
EveryNth	$\mathcal{O}(n_{out})$	$\mathcal{O}(n_{out})$	✓	1
MinMax	$\mathcal{O}(N)$	$\mathcal{O}(n_{out})$	✓	2
M4 [20]	$\mathcal{O}(N)$	$\mathcal{O}(n_{out})$	✓	4
LTTB [13]	$\mathcal{O}(N)$	$\mathcal{O}(n_{out})$	×	1
RDP [21]	$\mathcal{O}(N * n_{out})$	$\mathcal{O}(n_{out})$	×	N.A.
VW [22]	$\mathcal{O}\left(N\left(N - n_{out}\right)\right)$	$\mathcal{O}(N)$	×	N.A.

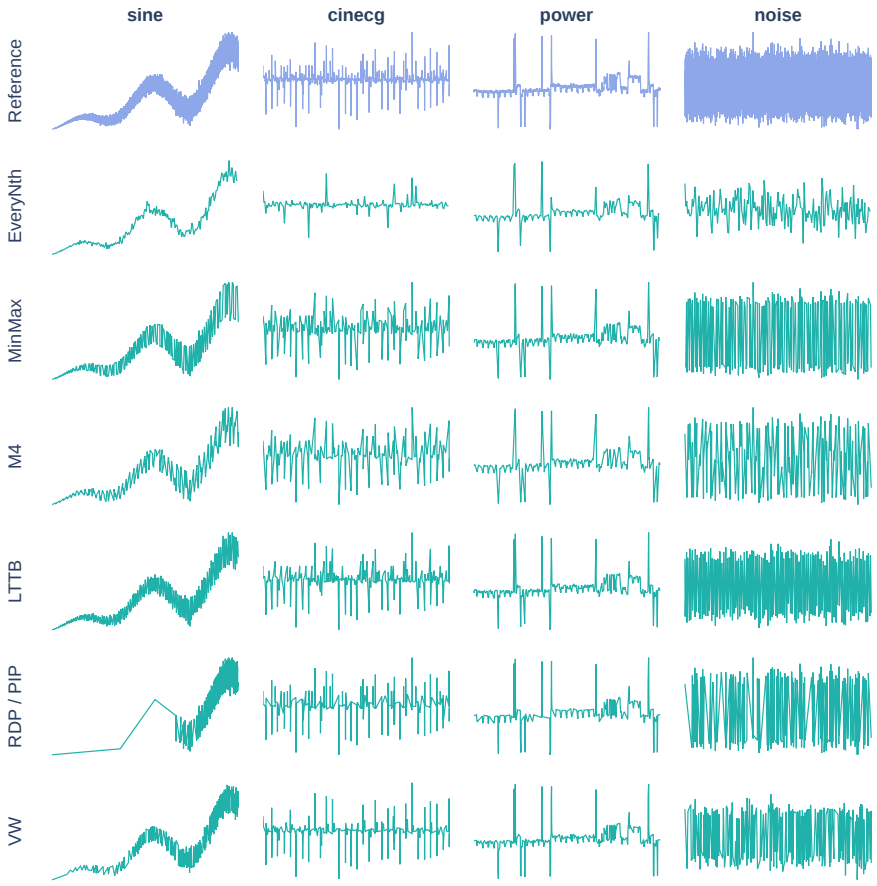


Figure 1.5: Visual comparison of six shape-preserving subsampling algorithms applied to four different time series signals, shown relative to the original (non-aggregated) line charts. Each aggregation is limited to 200 data points, while the reference visualizations contain 50,000 data points. The charts were created using Plotly with a line width of 2 pixels and anti-aliasing enabled. Additional information regarding the time series signals can be found in Section 3.3.1.1

performing comparison operations to find the minimum and maximum of the array. The memory complexity of the algorithm is $O(n_{out})$, and the algorithm is easily parallelizable over the bins, enabling efficient execution on multi-core processors.

M4, proposed by Jugel et al. [20], is a subsampling method that can achieve pixel-perfect data aggregation, implying that no pixel differences are present when comparing with the full-resolution line chart. For each bin, M4 selects two horizontal (first, last) and two vertical (min, max) extrema. Selecting these data points can be done in linear time complexity ($O(N)$) with $O(n_{out})$ memory complexity. The algorithm is also easily parallelizable over the buckets. Remark that M4 selects more data points per bin

than the other uniform algorithms.

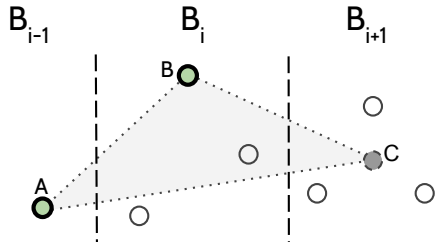


Figure 1.6: Illustration of the LTTB subsampling algorithm. The dashed vertical lines represent the boundaries of each bucket $\mathbf{B}$. Point A corresponds to the selected sample from the previous bucket $\mathbf{B}_{i-1}$, while point C is a fictive point representing the average of the next bucket's data points. The algorithm selects the data point from bucket B_i that forms the largest triangular surface with A and C.

Largest-Triangle-Three-Buckets (LTTB) is the most widely used algorithm among the several shape-preserving subsampling algorithms that Steinarsson introduced in his master's thesis [13]. The other proposed algorithms have either unfavorable time complexity or poorer visual representativeness compared to LTTB [23]. LTTB is based on the concept of effective triangular area, which is often employed in cartographic line simplification algorithms. The algorithm starts by selecting the first data point in the time series. Then, for each bucket, the algorithm selects the data point that forms the largest triangular surface with the previously selected data point and the average value of the next bucket, as illustrated by Figure 1.6. Finally, the last data point is included as well. With a time complexity of $O(N)$—since each point requires a triangular surface computation—and a memory complexity of $O(n_{out})$, LTTB scales linearly with data size. However, the slope of this linear scaling is steeper than those of MinMax and M4 due to the higher computational cost of area calculations as opposed to simple comparisons. The LTTB algorithm cannot be parallelized as the algorithm requires a sequential pass over the data since the previous bucket's selected data point is part of the local surface calculation.

RDP is a non-uniform subsampling technique that optimizes the l^{∞}-norm of the residual error [24, 21]. The algorithm is initialized with the boundary points of the input data and connects these with linear interpolation, as shown in Figure 1.7 (a, 1). New data points are then iteratively added by inserting the point with the greatest (orthogonal) distance to the interpolated line between already selected points (a, 2). This continues until either the desired number of output points or a specified minimal orthogonal distance threshold (ϵ) is reached. RDP excels in displaying spikes or large changes due to this interpolation-based distance selection. Remark that the iterative selection process, along with the orthogonal distance computation, results in higher time complexity than the above outlined techniques, and offers no opportunities for parallelization [25].

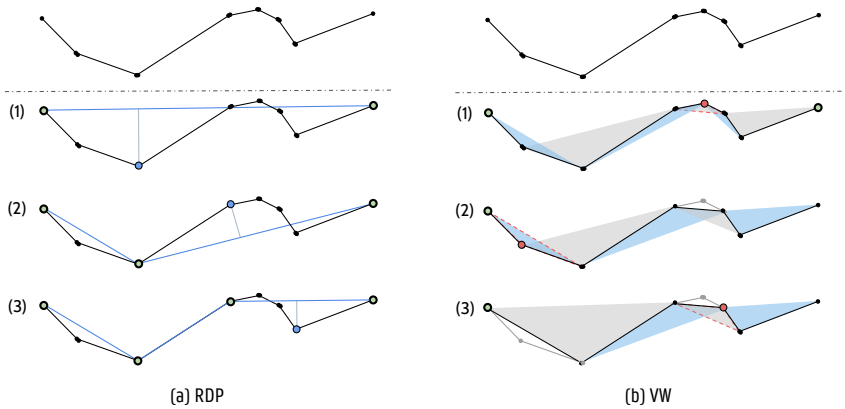


Figure 1.7: Illustration of the (a) RDP and (b) VW algorithms. RDP performs an iterative *selection* process based on the largest perpendicular distance to the already chosen points. VW performs an iterative *elimination* based on the smallest effective area. Selected data points are represented with green markers, whereas (to-be-)eliminated points are shown via red markers.

Visvalingam-Whyatt (VW) is, similar to RDP, a non-uniform line simplification algorithm [22]. However, instead of selecting points, VW iteratively removes the data point with the smallest effective area relative to its two adjacent (non-eliminated) points, illustrated by Figure 1.7 (b). This elimination process is repeated until the desired number of points remains. VW's effective area computation results in a more uniform data sampling than RDP, since the latter is solely concerned with orthogonal distances to a line segment, whereas VW also takes the horizontal distance into account when calculating the triangular area. As a consequence, VW may eliminate spikes in favor of covering larger horizontal spaces, rather than vertical ones. After each removal, the effective areas of the neighboring points are recalculated to determine the next point for elimination. This iterative process of recalculating and removing points results in the high time and memory complexity of VW. In practice, this makes the use of VW infeasible for large time series data, characterized by a large N to n_{out} ratio, requiring $\pm N$ removals and effective area recomputations.

Although these shape-preserving subsampling algorithms are widely used in dedicated applications, such as server-side downsampling of time series data within database systems [26]—they are not integrated in popular open-source data visualization libraries like Matplotlib, Bokeh, or Plotly. Additionally, no comprehensive study has compared these algorithms in terms of their visual approximation performance and their ability to handle key time series properties such as outliers, streaming data, and noise. Furthermore, as summarized in Table 1.1, several algorithms exhibit computational limitations, such as LTTB's lack of parallelization.

1.2.2 Processing

During the descriptive and exploratory analysis phase of time series data, hidden patterns, inconsistencies, or irregularities—such as gaps, noise, and signal artifacts—may become evident.

Processing refers to the transformation of one or multiple time series signals into new signals with the goal of cleaning the data or extracting more informative signals. This can range from basic operations, such as clipping or resampling the data, to more advanced techniques, such as combining signals into their signal magnitude vector (Euclidean norm) or filtering a signal into its high- and low-frequency components.

Figure 1.8 presents examples of processing steps applied to a skin conductance signal from a wearable, which aims to detect and remove artifacts. Note how visualizing intermediate outputs (processing steps) helps in evaluating and refining the effectiveness of the overall processing pipeline.

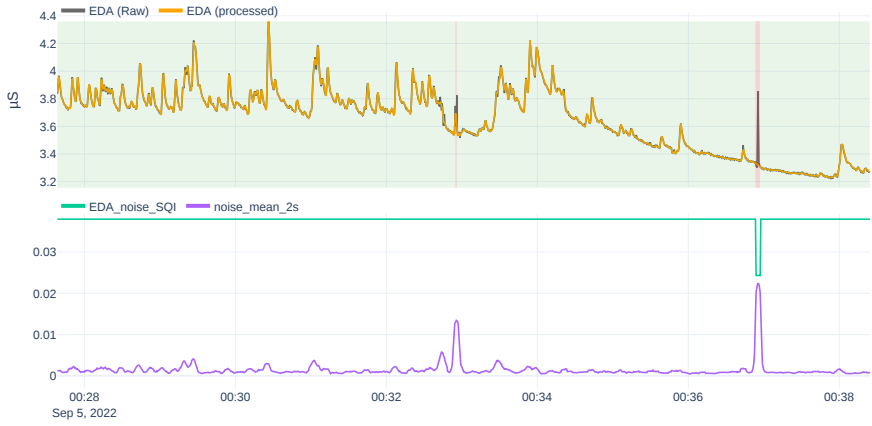


Figure 1.8: Visual representation of a signal processing methodology applied to an electrodermal activity (EDA) signal of wearable device. The upper subplot shows both the raw and processed signal, with the background shading indicating the signal quality—where red regions highlight signal artifacts. The lower subplot displays two intermediate signals: the purple "noise_mean_2s" (a 2-second rolling mean absolute difference between the raw and band-passed EDA signal) and the "EDA_noise_SQI" (a threshold-based signal quality index (SQI) on the "noise_mean_2s", indicating high-noise regions in the electrodermal activity (EDA) signal).

1.2.3 Feature Extraction

Time series data is often high-frequent and contains substantial information redundancy due to self-correlation. In order to reduce the dimensionality whilst retaining relevant characteristics, time series feature extraction arose as an established technique. Features typically describe specific aspects of a single time series signal, such as the slope,

goodness-of-fit to a curve, or energy within certain frequency bands. After extraction, i.e., calculation of the features, they can serve as input for downstream tasks such as classification or clustering [1]. Effective feature extraction typically leverages domain knowledge about the underlying system, the objective, and the measurement apparatus that generates the data stream.

In practice, data scientists often begin with analyzing classical features that capture statistical or physical properties of the time series [27]. Alternatively, they may employ extensive feature extraction methods, resulting in a large collection of features, followed by feature filtering methods to retain the most informative features for the task—a key aspect of the `tsfresh` toolkit [28].

An important consideration in feature extraction for time series data is that it relies on implicit assumptions, which may not always hold in real-world scenarios. A fundamental assumption is stationarity, where statistical properties of the time series—such as mean, variance, and autocorrelation—are expected to remain constant over time. However, real-world time series, such as the stock price chart (illustrated in Figure 1.2), often exhibit non-stationary behavior (i.e., general increasing trend). In such cases, the distributions or ranges of future values may differ significantly from historical data, making it challenging to develop data-driven models that generalize effectively.

To address this issue, a common strategy is to segment the time series into smaller (non-)overlapping frames or windows, often referred to as sliding windows. Within each window, the signal is typically assumed to approximate stationarity and ergodicity, allowing for more reliable feature extraction. These features capture local variations in the statistical properties of the signal over time, facilitating the analysis of its temporal evolution. In this context, ergodicity implies that time-averaged statistics within a window closely approximate the ensemble averages of the underlying process. This assumption ensures that features extracted from individual windows provide meaningful representations of broader statistical characteristics, even when the overall signal is non-stationary. Figure 1.9 illustrates this sliding window-based feature extraction process, which transforms the data into a more structured tabular format by aggregating both the sensor and temporal dimensions into a feature dimension. This window-based approach has proven to be effective in real-world applications where temporal dynamics are complex and global stationarity cannot be assumed. This localized perspective enables models to better capture transient patterns and adapt to the dynamic nature of time series data [1].

However, the choice of window size is critical. It must balance capturing meaningful local patterns while maintaining computational efficiency and ensuring sufficient data within each segment to estimate robust statistics. Moreover, despite the widespread use of sliding-window-based feature extraction, current feature extraction toolkits predominantly support only the use of a single window size. As a result, multi-resolution feature extraction methodologies (i.e., using multiple window sizes) remain understudied. Yet,

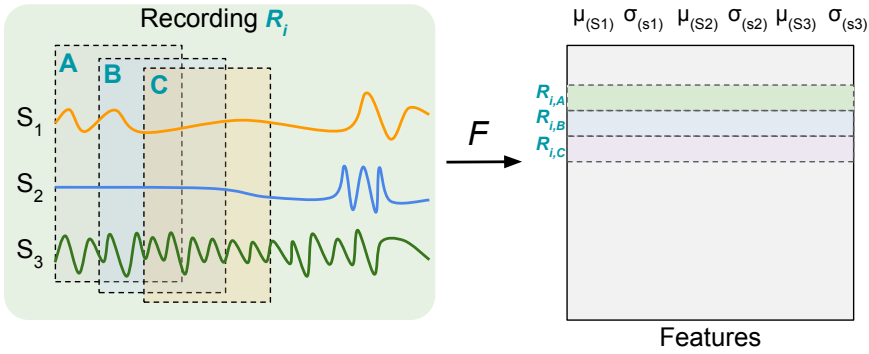


Figure 1.9: Sliding window-based feature extraction for a single recording R_i, consisting of three signal modalities S_1 to S_3. The left plot displays three feature windows—A, B, and C—over which feature functions F are applied. This results in feature vectors $R_{i,A}$, $R_{i,B}$, and $R_{i,C}$ each containing the mean (μ) and standard deviation (σ) of each signal S within the respective window.

this approach offers the potential to capture both short-term patterns and longer-term dynamics, resulting in richer representations than single-resolution methods. For example, in HAR, certain activities such as running may be better captured with shorter windows, while other activities, such as stretching, may benefit from longer ones. A single-window approach must compromise between these temporal scales, often leading to suboptimal representations.

1.2.4 Time Series Visualization and Feature Extraction Research Challenges

In this dissertation, we address several key research challenges related to time series visualization and feature extraction, as outlined in the previous sections:

Challenge C1: Connecting Downsampling Algorithms to visualization Libraries
While shape-preserving subsampling algorithms are widely used in specialized domains, such as server-side downsampling in database systems [26], they are notably absent in popular open-source visualization libraries like Matplotlib, Bokeh, and Plotly. This gap limits their accessibility and hinders widespread adoption in data analysis workflows. The first challenge, therefore, is to seamlessly integrate these algorithms into mainstream visualization tools in a user-friendly and intuitive way. This is particularly difficult for interactive libraries such as Plotly and Bokeh, which lack built-in hooks to dynamically trigger data operations in response to user interactions like zooming or panning.

Challenge C2: Evaluating Visual Representativeness
Despite their application in practice, there has been no comprehensive evaluation com-

paring shape-preserving subsampling algorithms on their ability to visually approximate time series data. Specifically, it remains unclear how well different algorithms preserve important characteristics such as outliers, trends, and noise patterns. As such, a second challenge is to systematically assess and benchmark the visual fidelity of these methods across a variety of time series scenarios.

Challenge C3: Addressing Computational Limitations
Many shape-preserving subsampling algorithms suffer from computational inefficiencies. For instance, LTTB is difficult to parallelize, limiting its scalability to large datasets (as summarized in Table 1.1). The third challenge is to design algorithmic improvements that reduce runtime complexity and enhance scalability, without compromising visual quality.

Challenge C4: Facilitating Convenient Time Series Data Wrangling
Real-world time series data is often heterogeneous, multivariate, and may contain irregular sampling or missing data. The fourth challenge is to develop a flexible framework for time series feature extraction and processing. This framework should support the convenient specification of multi-resolution feature extraction configurations and enable robust handling of multivariate and irregular time series data.

1.3 Wearable Sensing

Wearable sensing is a rapidly growing field in consumer and medical technology, driven by the decreased cost of compact electronic devices. These devices are integrated into gadgets, accessories, or clothing designed to be worn on the human body [29]. Unlike mobile devices, such as smartphones, wearables offer unique advantages by providing continuous, body-centric monitoring. Examples include smartwatches, rings, and fitness bands, which often come equipped with sensors such as accelerometers, gyroscopes, heart rate monitors, and electrodermal activity (EDA) sensors.

1.3.1 Real-World Monitoring Studies

The non-intrusive nature of wearable devices makes them particularly suitable for longitudinal monitoring studies, which typically serve one of two purposes:

1. **Inter-subject Analysis** compares data from a target group to that of a control group. For example, individuals diagnosed with major depressive disorder may be compared to healthy controls to identify distinctive patterns or deviations [30].

2. **Intra-subject Analysis** examines variations within the same individual. This can involve (i) discrete comparisons, such as analyzing periods around specific events (e.g., epilepsy or headache episodes) versus baseline periods distant from

such events [31], or (ii) continuous variables, such as monitoring changes in the severity of a chronic disorder over time [32].

Intra-subject analysis provides several advantages. By comparing the data within the same individual, it eliminates participant selection biases that may arise when comparing across groups [33]. It also requires smaller sample sizes, making it particularly suitable for studies with limited participant availability. However, when the goal is to evaluate the efficacy of an intervention, such as a medication or treatment, inter-subject analysis remains the preferred methodology.

1.3.2 Data Characteristics of Wearables

Wearable sensing devices generally exhibit certain time series characteristics in their sensor data output:

- **Multi-sensor input**: Wearables are often equipped with multiple sensors that capture diverse data types, such as heart rate, movement, and skin temperature. Additionally, a single sensor, like an accelerometer, can measure multiple dimensions simultaneously, such as x, y, and z axes, as illustrated in Figure 1.2.

- **Heterogeneous sampling rates**: Sensors in wearable devices typically operate at different sampling frequencies. For instance, skin temperature may be sampled at 1 Hz, while motion sensors like accelerometers or gyroscopes often sample data at rates exceeding 32 Hz. Moreover, derived metrics may even have irregular sampling intervals. For instance, RR-intervals (the time between consecutive heartbeats, see Figure 1.2) are time-stamped based on detected heartbeats rather than following a fixed sampling rate.

- **Discontinuous data (sharding)**: Due to battery constraints, wearable devices often produce data intermittently, resulting in discontinuous or "sharded" data streams. This behavior requires careful handling during analysis to properly handle these gaps or inconsistencies [34].

- **Signal artifacts**: Many physiological signals, such as skin conductance or optical blood volume pulse (BVP), are highly susceptible to noise and artifacts caused by movement or environmental factors [2]. These artifacts require either signal enhancement techniques, robust processing algorithms, or specialized methods for artifact identification and removal.

The combination of these characteristics highlights the unique challenges in analyzing wearable time series data.

1.3.3 Wearable Data Quality Research Challenges

This dissertation focuses on the following research challenges aimed at improving the quality of wearable time series data, as outlined in the previous sections:

Challenge C5: Improving Data Quality of Wearable Monitoring Studies
Although wearables offer tremendous potential for monitoring human physiology and behavior in real-world environments, they are subject to various limitations. This challenge focuses on systematically identifying key data quality issues that arise during both data collection and retrospective analysis. Furthermore, it seeks to propose practical countermeasures to mitigate these challenges and improve the reliability and utility of wearable datasets.

Challenge C6: Assessing the Effectiveness of Proposed Countermeasures
To assess the impact of the proposed countermeasures from Challenge 5, this challenge focuses on mapping these interventions onto real-world data analysis scenarios. The objective is to empirically evaluate their effectiveness and provide guidelines for data analysis methodologies in future wearable monitoring studies.

1.4 Machine Learning

Artificial intelligence (AI) encompasses a wide range of techniques aimed at enabling machines to perform tasks that demonstrate intelligent behavior. It plays a fundamental role across all categories of data analytics, from dimensionality reduction for visualizing large, high-dimensional datasets (descriptive analytics) to machine learning techniques applied in diagnostic, predictive, and prescriptive analytics.

Early AI systems predominantly relied on rule-based or symbolic approaches, where human experts encoded domain knowledge into structured logical rules [35]. However, as computational power grew and data became more abundant, data-driven approaches—collectively referred to as ML—have become the dominant paradigm within AI.

Unlike rule-based systems that depend on manually encoded logic and heuristics, ML algorithms learn patterns directly from data by tuning mathematical representations to predict outcomes or reveal underlying structures [36]. Essentially, ML can be conceptualized as parameterized function learning, where an algorithm—referred to as a model—learns to map inputs to outputs through a set of adjustable parameters [37]. These parameters are typically optimized iteratively, often by minimizing a loss function that quantifies the discrepancy between the model's predictions and the actual data.

The adoption of ML has grown rapidly in fields like computer vision, healthcare, and wearable technology [38]. Devices such as fitness trackers and smartwatches generate vast amounts of physiological and behavioral data, often exhibiting complex temporal

patterns that are challenging for human experts to fully interpret. By leveraging ML techniques, it becomes possible to deduce meaningful insights, such as identifying human activities (e.g., walking, cycling) or transportation modes (e.g., bus, car, train) based on movement signals from wearable devices [39].

1.4.1 Machine Learning Taxonomy

Machine learning models are typically categorized based on the nature of the data and the learning objective [36]. Two primary categories are supervised and unsupervised learning:

1. **Supervised Learning**
   In supervised learning, each training sample or observation x is accompanied by a label or target value y (e.g., the presence or absence of a certain medical condition or a headache intensity score). The goal is to learn a mapping function g that predicts labels from input features:

   $$g(x) \rightarrow y$$

   Supervised learning tasks are typically divided into two subcategories:

   - **Regression**: The target variable y is continuous, such as predicting the perceived intensity of a headache episode on a numerical scale.
   - **Classification**: The target variable is discrete, such as classifying whether a patient is likely to experience a migraine attack within the next 24 hours (binary classification) or detecting which activity they are performing (multiclass classification).

2. **Unsupervised Learning**
   In unsupervised learning, no explicit labels or target values are provided. Instead, the algorithm seeks to uncover hidden structures or patterns in the data. Techniques such as clustering, dimensionality reduction (e.g., Principal Component Analysis (PCA)), and outlier detection (e.g., isolation forest) fall in this category. Examples include grouping patients with similar wearable signal profiles (clustering) or detecting anomalies in physiological data that may correlate with headache onset.

While supervised and unsupervised learning form the foundation of most ML tasks, alternative paradigms are gaining traction. **Semi-supervised learning** leverages a small amount of labeled data in combination with large unlabeled datasets, effectively bridging the gap between supervised and unsupervised methods. **Reinforcement learning (RL)**, in contrast, is centered on sequential decision-making by maximizing cumulative rewards. In wearable and clinical contexts, time series data is typically

abundant, but labeled examples are scarce and costly to obtain. This has spurred interest in **self-supervised learning (SSL)**—particularly in the light of recent advances in generative AI—as a means of extracting meaningful representations from unlabeled data. SSL models are trained using pretext tasks, such as predicting masked segments, reconstructing input signals, or distinguishing between different augmented views of the same time series. Despite this, traditional supervised ML methods remain the dominant approach because clinical applications often demand high reliability, interpretability, and regulatory compliance [40].

1.4.2 Traditional Supervised Machine Learning Methods

Traditional ML methods are generally not designed to operate directly on raw, high-dimensional data such as time series signals, images, or complex sensor outputs. Instead, they rely on manually engineered features that transform raw inputs into compact, informative representations. Feature engineering thus plays a critical role in enabling these models to perform effectively, often requiring domain expertise to extract meaningful statistical, temporal, or spatial characteristics from the data. Below, we outline the most prominent traditional supervised ML methods, highlighting their fundamental principles, strengths, and limitations.

Linear Models, such as Linear Regression and Logistic Regression, are foundational in machine learning, partly because they blend historical roots in statistics with straightforward interpretability. Linear regression predicts continuous outputs by modeling a linear relationship between the input features (i.e., the attributes of each observation for tabular data) and the target variable. Logistic regression extends this framework to classification tasks by modeling the probability of a class label using the logistic function. While these methods can be powerful baselines, they often struggle with complex, non-linear relationships.

support vector machine (SVM) models aim to find an optimal decision boundary (or hyperplane) that separates classes with maximum margin. Through the use of kernels (e.g., radial basis function), SVMs can handle non-linearly separable data by projecting it into higher-dimensional spaces. They often perform well in settings with moderate-sized feature sets, but can become computationally intensive for very large datasets.

Decision Trees are a machine learning algorithm that reflects the structure of early rule-based AI systems by organizing decisions into a hierarchical, tree-like format. Each node within a tree represents a decision based on a feature, with branches corresponding to possible outcomes, and leaf nodes providing the final prediction or classification. Unlike the early rule-based systems, where experts manually define rules, decision trees automatically learn these rules from data during training, making them more flexible and data-driven. Their step-by-step decision-making process ensures interpretability, allowing users to trace how predictions are made.

Ensemble Methods combine multiple "weak" or base learners (i.e., relatively simple machine learning models) to achieve improved performance and robustness. A key factor in their success is introducing diversity among the learners. For instance, **Random Forests** average predictions from a collection of decision trees. Diversity is achieved through "bagging", where each tree is trained on a random subset of the data. In contrast, **Gradient Boosted Trees** (e.g., XGBoost, LightGBM) build decision trees sequentially, with each new tree focusing on correcting the errors of the previous ones. This iterative approach is called "boosting" and often delivers state-of-the-art (SotA) performance, particularly in tabular data use-cases [41, 42].

Overall, traditional machine learning methods are highly relevant in real-world applications. They are well-suited to accommodate heterogeneous feature sets, can efficiently handle small to moderately large datasets, and offer interpretable outputs, such as feature importances, which are critical in domains like clinical decision-making [43]. Moreover, when coupled with well-designed feature engineering, these methods can achieve competitive performance compared to deep learning approaches [41]. Nevertheless, as data volumes grow and the complexity of tasks increases, more advanced techniques—especially deep learning—have gained traction, offering end-to-end learning capabilities directly from raw, high-dimensional inputs such as time series signals and images.

1.4.3 Deep Learning

Deep learning is a specialized area within machine learning that focuses on using deep artificial neural networks to automatically learn data representations [37]. Neural networks (NNs) form the foundation of many modern machine learning approaches. At their core, NNs are computational models consisting of layers of interconnected nodes (or "neurons"), where each neuron transforms its inputs through weighted connections and non-linear activation functions. By stacking multiple layers of these transformations, deep learning models can learn increasingly abstract and complex representations of the data. As such, they do not require a manual feature engineering step. These networks process input data through multiple layers of non-linear transformations, successively refining features automatically to produce more accurate predictions. The training process is driven by backpropagation, which iteratively adjusts the model parameters using gradient descent to minimize prediction errors.

A key advantage of deep learning is its ability to perform representation learning [37], where meaningful features are extracted from raw data through layered transformations [44]. This capability makes deep learning particularly effective for tasks involving complex data structures, such as time series analysis. Moreover, deep learning models support transfer learning, allowing pre-trained networks (i.e., a deep learning model trained on a large dataset to perform a particular task, e.g., image

classification) to be fine-tuned for related tasks with minimal data and computational effort [45, 46].

Despite these advantages, deep learning also presents several important limitations. These models are highly data-hungry, often requiring large amounts of labeled data to achieve competitive performance. In addition, deep networks are generally considered less interpretable than traditional methods, making it difficult to understand clearly what the model has learned. Unlike traditional machine learning, where practitioners can guide the learning process by explicitly selecting and engineering relevant features, deep learning operates as a largely opaque, end-to-end system. This lack of transparency can pose challenges in domains like healthcare, where explainability and trustworthiness are critical requirements.

1.4.4 Time Series Specific Machine Learning

When training supervised machine learning algorithms, either via classical ML or deep learning (DL), two key assumptions are made: (i) the ML algorithm is capable of picking up the relationship between the data samples and the labels [47], and (ii) the data is representative for the task at hand. The second assumption is often overlooked, particularly in the case of time series data, where the connection between a label and the data (or a representation thereof) is hard to analyze.

As discussed in Section 1.2, time series data is inherently heterogeneous and presents unique challenges. However, preprocessing and feature extraction can standardize the data, transforming it into a tabular format (see Figure 1.9). This transformation enables the application of traditional machine learning methods on time series data and should be considered as a baseline before investigating deep learning approaches.

Given the immense success of deep learning in other domains, such as for images and text, and the structured nature of time series data (e.g., high self-correlation), deep learning methods—particularly those employing 1D convolutions or recurrent connections such as long short-term memory (LSTM) models—are a promising fit. Here, the term *structured* no longer refers to a predefined tabular schema, but rather to the presence of intrinsic temporal dependencies that deep learning models can exploit. However, achieving effective and generalizable results requires addressing several key factors: gathering a sufficiently large and accurately labeled dataset, applying sensible augmentations to improve generalization, and selecting an appropriate deep learning architecture. Correctly navigating these different factors is non-trivial and should not be underestimated.

Interestingly, traditional machine learning methods, such as boosted tree-based models applied to feature-engineered tabular representations of time series, often match or even surpass deep learning models when well-crafted features are extracted [48]. Additionally, for specific tasks, such as time series anomaly detection, simpler methods like the Matrix Profile—which measures self-similarity within a sensor signal using

a sliding window approach—continue to demonstrate strong effectiveness and generalizability [49]. For example, in a co-authored study, which is not included in this dissertation, I demonstrated together with the co-authors that linear models, when combined with multi-resolution (multi-window) feature extraction, can perform on par with deep learning approaches for sleep-stage detection using polysomnography (PSG) data [50]. Despite ongoing advancements in DL, classical methods frequently outperform their more complex counterparts in terms of efficiency and robustness on small- to moderate-sized datasets. This contributes to the perception of time series analysis as the "Wild West" of machine learning (ML), where no single approach consistently dominates across all tasks [51].

1.4.5 Machine learning for Time Series Research Challenges

Within this dissertation, we will focus on the following research challenges as highlighted in the previous sections:

Challenge C7: Evaluating Effectiveness of Traditional ML on Wearable Data
Building on prior work that demonstrated the competitiveness of traditional ML approaches for data-intensive tasks such as PSG-based sleep stage scoring [50], this challenge investigates the performance of traditional machine learning baselines on supervised wearable time series problems. Specifically, it explores how their effectiveness can be enhanced through multi-resolution feature extraction and advanced signal processing techniques.

1.5 Real-World Use Case: Primary Headache Disorders Monitoring

Several research challenges and goals in this dissertation are evidenced through a real-world use case that was performed in close collaboration with the domain experts at Ghent University Hospital, i.e., monitoring primary headache disorders through wearables. Primary headache disorders, including migraines and cluster headaches (CHs), rank among the most prevalent and exhausting neurological conditions, affecting the quality of life of millions worldwide [52]. The episodic nature of these conditions, combined with the influence of various physiological and environmental factors, makes them particularly well-suited for study through wearable monitoring.

According to the international classification of headache disorders, third edition (ICHD-3), headaches can be classified as either primary or secondary [53]. Primary headaches are conditions in which the headache itself, along with its associated characteristics, is the disorder. In contrast, secondary headaches are the result of an underlying

condition, for example medication overuse headache (MOH) [54]. The most common primary headache disorders are migraine, CH, and tension-type headache (TTH).

Table 1.2: Migraine and cluster headache characteristics.

Feature	**Migraine** [55, 56]	**Cluster headache [57]**
Prevalence	Common (∼12% of the population)	Rare (∼0.1% of the population)
Pain type	Throbbing, pulsating	Piercing, stabbing
Pain location	Typically unilateral, can shift sides	Strictly unilateral, behind/around one eye
Attack duration	4–72 hours	15 minutes to 3 hours
Attack frequency	Variable (days to weeks between attacks)	Multiple attacks per day in clusters
Associated symptoms	Nausea, vomiting, photophobia, phonophobia	Eye redness/tearing, nasal congestion
Aura Presence	Present in ∼25% of cases	Absent
Triggers	Stress, hormones, certain foods, sleep changes	Alcohol, strong smells, seasonal changes
Relief during attack	Resting in a dark, quiet room	Movement, restlessness

Migraines are characterized by recurrent episodes of moderate-to-severe headaches that often manifest as a throbbing or pulsating pain, typically localized to one side of the head (i.e., unilateral pain) [55]. These headaches are frequently accompanied by nausea, vomiting, and heightened sensitivity to light (photophobia) and sound (phonophobia). In approximately 25% of cases, migraines are preceded by an aura, a transient sensory disturbance, such as visual changes or speech disruptions, which serves as a warning of the approaching headache phase.

Cluster headaches, on the other hand, are less common but are recognized as one of the most severe forms of headache [58]. They are characterized by excruciating, unilateral pain, focused around or behind one eye. CH attacks typically occur in patterns or "clusters" that can last weeks to months, followed by remission periods with no symptoms. During attacks, patients often experience additional symptoms, such as red or watery eyes, nasal congestion, and restlessness. Unlike migraines, cluster headache attacks are shorter in duration, typically lasting 15 minutes to 3 hours, but they can occur multiple times a day, often at consistent and predictable times [57].

Table 1.2 provides a comparison of the main characteristics of migraine and cluster headache attacks.

Wearable devices provide a promising approach for monitoring headache disorders in real-world settings, enabling both inter- and intra-subject analyses. For instance, wearable sensors can capture data on physiological changes during or around headache episodes and compare them to baseline periods, contributing to a better understanding and management of these conditions. However, research assessing behavioral, physi-

ological, and movement changes in longitudinal, real-world contexts—for both CH and migraine—remains limited. Existing studies often suffer from observation periods, inconsistent methodologies, or insufficient sample sizes [59, 60, 61].

This dissertation utilizes data from the *mBrain* study [62], an interdisciplinary collaboration between the neurology department of Ghent University Hospital and IDLab-imec. In the *mBrain* study, patients diagnosed with primary headache disorders were equipped with a wearable (Empatica E4) and a dedicated application to log headache attacks, facilitating the collection of synchronized physiological and self-reported data in daily life contexts. Figure 1.10 provides a high-level overview of the mBrain setup.

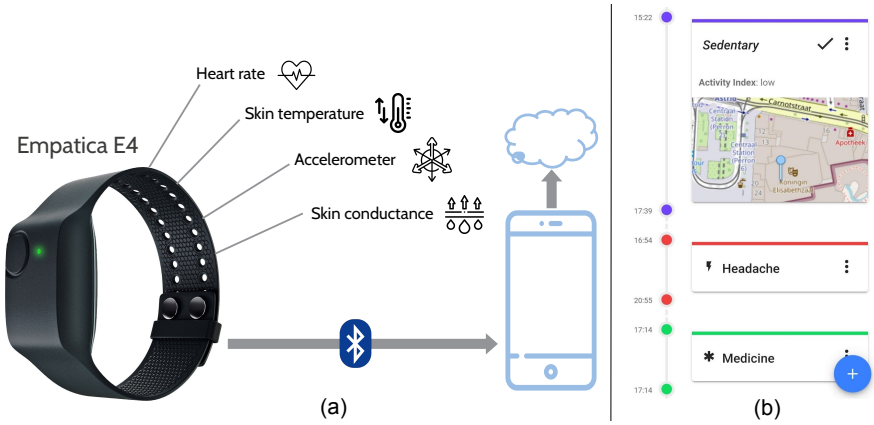


Figure 1.10: Schematic overview of the mBrain study setup. Panel (a) shows how data from the various modalities of the Empatica E4 wearable are transmitted via Bluetooth to a smartphone, which subsequently uploads the data to the cloud in near real time. Panel (b) illustrates how this data stream is used to generate timeline predictions, such as activity inference. Manually entered EMA events, such as headache reports and medication intake, are also visualized alongside these predictions.

1.5.1 Real-World Primary Headache Disorder Monitoring Research Challenges

Challenge C8: Evaluating the Translatability of Existing Clinical Knowledge on Behavior During Migraine and CH Attacks to Real-World Settings

This challenge focuses on evaluating how well existing clinical insights regarding behavioral patterns during migraine and CH attacks translate to real-world observations derived from wearable monitoring studies. Given the substantial heterogeneity of everyday environments, the influence of acute treatments, and the chronic nature of these conditions, it is likely that direct translation of clinical insights is limited. To bridge this gap, rigorous analyses that account for latent variables—such as medication

type, perceived headache intensity, and individual variability—are essential. These efforts may help identify which clinical findings remain robust outside controlled environments and uncover novel behavioral signatures specific to real-world settings.

1.6 Research Goals

Building on the challenges outlined earlier, this dissertation seeks to advance large-scale time series data analytics by focusing on two main areas: (i) time series visualization and (ii) wearable data analysis. The latter is further divided into two subfields: applied machine learning and real-world data analysis. The research objectives associated with these focus areas are structured as follows:

Effective Visualization

- **RG1**: *Facilitating scalable time series visualization*
  As discussed in Section 1.2, visualization is essential for understanding the complexity and nuances of time series data, with line charts being particularly effective for most tasks. To visualize large time series data effectively, a visualization toolkit must satisfy four key requirements: interactivity (to enable effective exploration of large datasets), scalability (through data aggregation), integrability, and configurability. Additionally, given the focus on data analytics, the toolkit should include a Python interface, which is the predominant programming language for data science. A review of the open-source landscape revealed a lack of solutions meeting these criteria, leading to Challenge **C1**, which we aim to tackle through **RG1-A**.

  Moreover, existing research on evaluating shape-preserving subsampling algorithms suffers from inconsistencies in metrics and methodologies. For example, $M4$ measures data efficiency based on the number of buckets rather than the data points [20], leading to limited consensus on algorithm effectiveness. This issue, expressed in Challenge **C2**, will be addressed in **RG1-B**. Finally, the widely used LTTB aggregation algorithm has scalability limitations due to its computational characteristics, which are tackled in **RG1-C**, thereby tackling Challenge **C3**.

  * **RG1-A**: *Enhance the scalability of existing visualization toolkits*

    Hypothesis: Extending an interactive, web-based toolkit with an interface towards subsampling algorithms, and coupling graph interactions to trigger subsampling, will significantly improve scalability while maintaining interactivity, integrability, and configurability, thereby satisfying the four key requirements for effective visualization.

  * **RG1-B**: *Evaluate and compare the visual representativeness of shape-preserving subsampling algorithms*

Hypothesis: Standardizing evaluation metrics and validating visual similarity across a wide range of time series signals with different properties will reveal differences in the representativeness of existing shape-preserving subsampling algorithms, thereby identifying methods that outperform other techniques and clarifying under which data properties they do so.

* **RG1-C**: *Improve the algorithmic efficiency of the LTTB algorithm.*

  Hypothesis: Reducing the search space of the LTTB algorithm by selecting a subset of vertical extrema for effective area computation will significantly improve its scalability without compromising visual fidelity.

Wearable Data Analytics

- **RG2**: *Improving traditional time series machine learning*
  Section 1.2 covered how heterogeneous time series data can be standardized through processing and transformed into a tabular format via feature extraction. However, as highlighted in Subsection 1.2.3 and Challenge **C4**, no convenient toolkit exists for multi-resolution feature extraction, which we aim to tackle in **RG2-A**.

  Furthermore, Section 1.4 outlined the ML landscape, positioning time series machine learning as the "Wild West" due to the high variability in method effectiveness across datasets and tasks. Traditional machine learning approaches often perform on par with more complex models, yet their potential remains underexplored. In **RG2-B**, we aim to (re)evaluate the relevance of traditional machine learning when using multi-resolution feature extraction and advanced processing techniques, thereby tackling Challenge **C7**.

  * **RG2-A**: *Enable flexible and efficient processing and feature extraction*

    Hypothesis: Designing a framework that utilizes a shared time index across multiple sensor data streams, and specifies windows based on this time index (rather than sample indices), will enable holistic feature extraction for modalities with different, and even irregular, sampling rates.

  * **RG2-B**: *Evaluate the competitiveness of traditional machine learning approaches*

    Hypothesis: By participating in two wearable HAR ML competitions, focusing on well-crafted feature extraction and signal processing pipelines combined with boosted tree-based models, we expect to demonstrate that properly tuned traditional machine learning baselines remain highly competitive for certain time series tasks.

- **RG3**: *Advancing the analysis of real-world wearable data*
  As discussed in Section 1.3, wearables offer valuable opportunities for real-world monitoring studies but also present significant challenges, particularly in data quality and methodological variability. These challenges arise from variations in study requirements, sensor reliability, data availability (including sparsity), and differences in data preprocessing techniques. To address this, **RG3-A** focuses on devising actionable strategies to improve the reliability of wearable-based data analysis, tackling Challenge **C5**. These strategies will then be applied in **RG3-B** to investigate the movement behavior of migraine and cluster headache patients using the real-world *mBrain* dataset [62], contributing towards Challenge **C6** and **C8**. These case studies exemplify the types of real-world wearable data quality challenges **RG3-A** aims to address—such as dealing with missing data, non-wear periods, and signal artifacts. Interestingly, the advancements of **RG1** can also be leveraged throughout **RG3**, as visualization of time series data plays a crucial role in the analysis of real-world wearable data.

  * **RG3-A**: *Devise methodologies for improving the reliability of real-world wearable data analysis*

    Hypothesis: Leveraging interdisciplinary first-hand experience from the real-world monitoring studies facilitates a systematic identification of key wearable- and study-related challenges, leading to the development of practical methodologies that enhance the reliability of wearable data analysis.

  * **RG3-B**: *Utilize the methodology to a real-world data analysis use case*

    Hypothesis: Applying the developed methodology to the *mBrain* dataset will demonstrate its practical applicability and reveal new insights into the movement behavior of migraine and cluster headache patients under real-world conditions.

1.7 Outline and Overview of Contributions

As outlined in the previous section, this dissertation explores two key topics within the domain of large-scale time series data analytics: *Visualization* and *Wearable Data Analytics*. The latter is further divided into two subfields: *Applied Machine Learning* and *Real-World Data Analysis*. Figure 1.11 illustrates the chapters within each (sub)topic and highlights their interconnections.

Effective Time Series Visualization

Visualization is a crucial component in every aspect of data analytics, especially when dealing with time series data. Line charts have proven to be particularly effective for

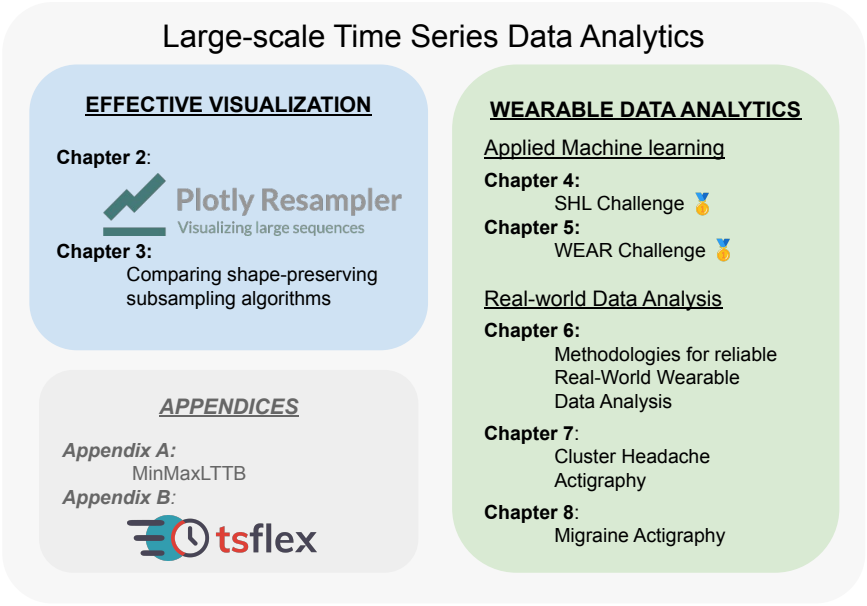


Figure 1.11: Schematic overview of the dissertation chapters.

most time series tasks, and utility is further enhanced when combined with the *zoom plot* visualization-interaction technique [63]. This allows users to explore an overview while controlling zoom levels and accessing detailed views on demand [9].

However, to support data analytics on large-scale time series data, scalable visualization tools are required. A common limitation of existing interactive visualization frameworks is their default approach to render all data points in the front-end or canvas, which results in network and rendering latencies. Moreover, as the number of rendered data points increases, user interactions such as zooming and panning become less responsive, hindering effective data exploration [14]. The limited canvas size also results in overplotting, where multiple data points visually overlap [64].

Shape-preserving time series subsampling methods, such as LTTB [13], RDP [21] or M4 [20] can be used to address these challenges by selecting a reduced set of data points that preserve the visual characteristics of the full time series signal when rendered. These algorithms provide a near-optimal heuristic to overcome scalability challenges. In this dissertation, two chapters and one appendix are dedicated to exploring and facilitating scalable time series visualization using shape-preserving subsampling algorithms (**RG1**).

Chapter 2 presents `Plotly-Resampler`, an open-source extension to the Python bindings of the popular `plotly.js` visualization library. `Plotly-Resampler` enables dynamic aggregation of time series data by coupling shape-preserving subsampling algorithms to interactive graphs. Specifically, for every user-graph interaction, such as

zooming, panning, or scrolling, a new aggregation is dynamically computed for the new data range according to the specified configuration. The accompanying chapter and demo showcase `Plotly-Resampler`'s ability to render 110 million data points (distributed across five signals) without any significant rendering delays, achieving **RG1-A**.

Chapter 3 explores the domain of shape-preserving subsampling algorithms. It presents an extensible evaluation framework I designed to assess these algorithms' visual representativeness. Two aspects are evaluated: (1) the effectiveness of subsampling at various sample sizes (n_{out}) across diverse time series characteristics (e.g., stationarity, noise, spikes), providing insights into data efficiency, i.e., visual representativeness at low n_{out}; and (2) a novel concept called *visual stability*, which measures an algorithm's ability to select similar data points during minor user interactions, such as limited panning or small zooms. The *visual stability* concept is closely linked to aggregations performed on streaming data. Based on these evaluations and computational considerations, the MinMax and LTTB algorithms are identified as the most effective for a wide range of time series data. Moreover, the proposed framework can be applied to assess the visual representation capabilities of newly developed aggregation algorithms, thereby realizing **RG1-B**.

Appendix A presents `MinMaxLTTB`, a heuristic extension of LTTB that uses MinMax-preselection to reduce the search space before applying LTTB. In contrast to LTTB, MinMax-preselection, which applies the MinMax algorithm to retain a certain multitude of the desired number of subsampled points, is easily parallelizable and less computationally intensive. Experiments show that MinMaxLTTB can be up to 30 times faster than LTTB while maintaining the same visual representation quality, as validated using the evaluation framework from Chapter 3. This achieves **RG1-C**.

Wearable Data Analytics

The second part of this dissertation focuses on predictive and descriptive analytics of real-world wearable data. The predictive analytics component evaluates the performance of traditional ML methods on two wearable movement datasets through competing in two HAR ML competitions. However, because these datasets are already preprocessed and of high-quality—featuring aligned labels and image-based ground truth—they are less representative of real-world, event-based studies such as those involving the monitoring of primary headache disorders. Therefore, the latter half of this part draws on our direct experience with such studies, presenting practical methodologies to improve data quality. In addition, it introduces two analysis-focused papers, each accompanied by detailed methodological documentation aimed at enhancing reproducibility and guiding future research in this domain.

Applied Traditional ML

Section 1.4 outlined the machine learning landscape, highlighting the unique challenges of time series ML. One fundamental difficulty is determining whether the target variable (e.g., acute psychosocial stress) is meaningfully captured in the input data (e.g., wearable physiology data), which complicates the evaluation of ML techniques. Additionally, limited research exists on the effectiveness of multi-resolution (or multi-window) feature extraction, likely due to the absence of a convenient toolkit for extracting such features.

Appendix B addresses this gap with the creation of `tsflex`, a toolkit designed for both time series feature extraction and signal processing. Its feature extraction interfaces allow users to define features for specific signals across multiple windows, enabling efficient multi-window feature extraction. Unlike many existing tools, `tsflex` is agnostic to sampling rates and data regularity. Benchmark results demonstrate its minimal computational overhead and superior efficiency compared to alternatives like `tsfresh`, fulfilling **RG2-A**.

Chapters 4 and 5 present solutions and insights from two supervised wearable ML challenges. In both cases, `plotly-resampler` (**RG1-A**) was used for efficient data exploration, facilitating informed feature extraction and processing with `tsflex`. In both challenges, exploration-driven processing along traditional ML methods was employed, leading to first-place finishes in both competitions. The resulting models achieved macro F1 scores approximately 10% higher than those of competing approaches, successfully fulfilling **RG2-B**.

Chapter 4 describes our solution for the Sussex-Huawei locomotion-transportation (SHL) challenge, where participants were tasked with classifying eight locomotion and transportation activities (e.g., walking, train, subway) from shuffled, non-overlapping 5-second windows of real-world smartphone sensor data (accelerometer, gyroscope, magnetometer). To make it even more challenging, one sensor modality was randomly omitted in each window. The dataset contained over 4 billion sensor data points. Through exploratory data visualization, we identified a distribution shift between the train and test sets. To mitigate this shift, targeted feature extraction and augmentations were applied, aimed at improving robustness and thereby leading to better generalization, which secured us the first place in the competition. Additionally, we observed that the commonly used SMV transformation led to the worst performance, highlighting the need for further research into alternative rotation-invariant feature and signal transformations.

Chapter 5 outlines our approach to the WEAR challenge, which involves classifying 18 workout activities, along with the transitional phase (i.e., no workout activity), using continuous data from four wearable sensors (one on each limb). In contrast to the SHL challenge, the continuous nature of the WEAR dataset allowed for multi-resolution feature extraction. Data analysis revealed inconsistent sensor orientations across participants, leading us to investigate rotation-invariant augmentation techniques. Additionally, post-processing with Gaussian temporal smoothing further improved our

performance, contributing to another first-place result. This chapter demonstrates that even simple features—requiring limited manual feature engineering—combined with multi-resolution feature extraction and signal processing, can achieve highly competitive results.

Real-World Wearable Data Analysis

Section 1.3 introduced wearables as low-intrusive sensing devices capable of capturing rich physiological and behavioral data, making them well-suited for longitudinal monitoring studies. These studies produce large volumes of time series data, necessitating scalable visualization and analysis techniques, addressed in **RG1** and **RG2-A**, respectively. Nonetheless, real-world deployments present significant challenges, including uncontrolled environments that introduce signal artifacts and declining participant compliance over time, both of which adversely affect data quality and completeness.

Chapter 6 identifies several key challenges in real-world wearable monitoring studies, particularly regarding participant data entries and wearable data analysis. For each challenge, practical countermeasures are proposed, supplemented with visualizations and code examples where applicable. These countermeasures are materialized by applying them to two datasets from real-world monitoring studies: the mBrain21 dataset [62] and the ETRI lifelog 2020 dataset [65]. Additionally, a part of the *mBrain* dataset is made open-access to promote reproducibility. The proposed actionable and reusable methodologies fulfill **RG3-A**.

Chapter 7 and 8 apply the proposed countermeasures by conducting qualitative analyses on the *mBrain* dataset, fulfilling **RG3-B**.

In **Chapter 7**, movement behavior during cluster headache attacks is analyzed by comparing movement intensity levels during headache periods to eligible non-headache periods. Given that cluster headaches are associated with restlessness and agitation during attacks [56], it was hypothesized that movement would increase during the ictal (i.e., headache) phase. However, the analysis revealed reduced movement during attacks, prompting further investigation. The findings suggest that most cluster attacks were acutely treated, which may have restricted movement due to treatment constraints (e.g., oxygen mask user) while simultaneously alleviating symptoms through acute interventions. This discrepancy between established clinical knowledge and real-world wearable data highlights the importance of contextual interpretation in wearable-based studies.

In **Chapter 8**, we conducted a similar movement analysis for the migraine subgroup within the *mBrain* dataset. Based on the mechanosensitivity characteristic of migraines, it was hypothesized that movement would decrease during the prodromal, ictal, and postdromal headache phases. While a significant reduction in movement was confirmed during the ictal phase, no significant changes were observed during the prodromal and postdromal phases. To further investigate these findings, additional subgroup analyses were performed, leveraging the study's longitudinal design to examine factors

such as acute treatment use and efficacy, motion sensitivity symptoms, and perceived headache intensity. In conclusion, this work demonstrates the potential of wrist-worn actigraphy to observe changes in physical activity during migraine attacks. Moreover, this chapter also discusses how the developed analysis methodology could be applied to other event-related chronic disorders, such as epilepsy and mood disorders.

1.8 Publications

Below is an overview of the publications written during my PhD, along with the open-source libraries I (co-)created in the process. An asterisk (*) next to the author's name signifies equal contribution.

1.8.1 Publications in International Journals

1. **Jonas Van Der Donckt***, Jeroen Van Der Donckt*, Emiel Deprost, and Sofie Van Hoecke. "tsflex: Flexible time series processing & feature extraction". In: *SoftwareX* 17 (2022), p. 100971

2. Jeroen Van Der Donckt*, **Jonas Van Der Donckt***, Emiel Deprost, Nicolas Vandenbussche, Michael Rademaker, Gilles Vandewiele, and Sofie Van Hoecke. "Do not sleep on traditional machine learning: Simple and interpretable techniques are competitive to deep learning for sleep scoring". In: *Biomedical Signal Processing and Control* 81 (2023), p. 104429

3. Jeroen Van Der Donckt, **Jonas Van Der Donckt**, and Sofie Van Hoecke. "tsdownsample: High-performance time series downsampling for scalable visualization". In: *SoftwareX* 29 (2025), p. 102045

4. Mathias De Brouwer, Nicolas Vandenbussche, Bram Steenwinckel, Marija Stojchevska, **Jonas Van Der Donckt**, Vic Degraeve, Jasper Vaneessen, Filip De Turck, Bruno Volckaert, Paul Boon, Koen Paemeleire, Sofie Van Hoecke, and Femke Ongenae. "mBrain: towards the continuous follow-up and headache classification of primary headache disorder patients". In: *BMC Medical Informatics and Decision Making* 22.1 (2022), p. 87

5. **Jonas Van Der Donckt**, Nicolas Vandenbussche, Jeroen Van Der Donckt, Stephanie Chen, Marija Stojchevska, Mathias De Brouwer, Bram Steenwinckel, Koen Paemeleire, Femke Ongenae, and Sofie Van Hoecke. "Mitigating data quality challenges in ambulatory wrist-worn wearable monitoring through analytical and practical approaches". In: *Scientific Reports* 14.1 (2024), p. 17545

6. Marija Stojchevska, Bram Steenwinckel, **Jonas Van Der Donckt**, Mathias De Brouwer, Annelies Goris, Filip De Turck, Sofie Van Hoecke, and Femke Ongenae.

"Assessing the added value of context during stress detection from wearable data". In: *BMC Medical Informatics and Decision Making* 22.1 (2022), p. 268

7. Marija Stojchevska, **Jonas Van Der Donckt**, Nicolas Vandenbussche, Mathias De Brouwer, Bram Steenwinckel, Koen Paemeleire, Femke Ongenae, and Sofie Van Hoecke. "Uncovering the Potential of Smartphones for Behavior Monitoring during Migraine Follow-up". In: *BMC Medical Informatics and Decision Making* 25 (2025), p. 88

8. Nicolas Vandenbussche*, **Jonas Van Der Donckt***, Mathias De Brouwer, Bram Steenwinckel, Marija Stojchevska, Femke Ongenae, Sofie Van Hoecke, and Koen Paemeleire. "Patients with chronic cluster headache may show reduced activity energy expenditure on ambulatory wrist actigraphy recordings during daytime attacks". In: *Brain and Behavior* 14.1 (2024), e3360

9. **Jonas Van Der Donckt***, Nicolas Vandenbussche*, Mathias De Brouwer, Bram Steenwinckel, Marija Stojchevska, Femke Ongenae, Koen Paemeleire, and Sofie Van Hoecke. "Analysis of Free-living Daytime Movement in Patients with Migraine with Access to Acute Treatment". In: *Journal of Headache and Pain* 26.33 (2025)

10. Nicolas Vandenbussche*, **Jonas Van Der Donckt***, Mathias De Brouwer, Bram Steenwinckel, Marija Stojchevska, Femke Ongenae, Sofie Van Hoecke, and Koen Paemeleire. "Tracking Migraine Symptoms: A Longitudinal Comparison of Smartphone-Based Headache Diaries and Clinical Interviews". In: *Neurology International* 17.3 (2025)

11. Mitchel Kappen*, **Jonas Van Der Donckt***, Gert Vanhollebeke, Jens Allaert, Vic Degraeve, Nilesh Madhu, Sofie Van Hoecke, and Marie-Anne Vanderhasselt. "Acoustic speech features in social comparison: how stress impacts the way you sound". In: *Scientific Reports* 12.1 (2022), p. 22022

12. Mitchel Kappen, Gert Vanhollebeke, **Jonas Van Der Donckt**, Sofie Van Hoecke, and Marie-Anne Vanderhasselt. "Acoustic and prosodic speech features reflect physiological stress but not isolated negative affect: a multi-paradigm study on psychosocial stressors". In: *Scientific Reports* 14.1 (2024), p. 5515

13. Gert Vanhollebeke, Mitchel Kappen, **Jonas Van Der Donckt**, Ingemarie Coquyt, Sofie Van Hoecke, Rudi De Raedt, Chris Baeken, Pieter van Mierlo, and Marie-Anne Vanderhasselt. "Similar and Dissimilar Neural Activity Along Dimensions of Psychosocial Stress: An EEG Source Imaging Study". In: *Cognitive, Affective and Behavioral Neuroscience (CABN)* (2025)
(in submission)

14. **Jonas Van Der Donckt***, Mitchel Kappen*, Vic Degraeve, Kris Demuynck, Marie-Anne Vanderhasselt, and Sofie Van Hoecke. "Ecologically valid speech collection in behavioral research: The Ghent Semi-spontaneous Speech Paradigm (GSSP)". in: *Behavior Research Methods* 56.6 (2024), pp. 5693–5708

15. **Jonas Van Der Donckt***, Jeroen Van Der Donckt*, Michael Rademaker, and Sofie Van Hoecke. "Shape-Preserving Subsampling for Scalable Line Chart Visualization: a Methodological Assessment". In: *Machine Learning and Knowledge Extraction* (2025)
(in submission)

1.8.2 Publications in International Conference Proceedings

1. Mathias De Brouwer, Nicolas Vandenbussche, Bram Steenwinckel, Marija Stojchevska, **Jonas Van Der Donckt**, Vic Degraeve, Filip De Turck, Koen Paemeleire, Sofie Van Hoecke, and Femke Ongenae. "Towards knowledge-driven symptom monitoring & trigger detection of primary headache disorders". In: *Companion Proceedings of the Web Conference 2022*. 2022, pp. 264–268

2. **Jonas Van Der Donckt***, Mathias De Brouwer*, Pieter Moens, Marija Stojchevska, Bram Steenwinckel, Stef Pletinck, Nicolas Vandenbussche, Annelis Goris, Koen Paemeleire, Femke Ongenae, and Sofie Van Hoecke. "From self-reporting to monitoring for improved migraine management". In: *Engineer meets Physician (EmP)*. 2022

3. **Jonas Van Der Donckt***, Jeroen Van Der Donckt*, Emiel Deprost, and Sofie Van Hoecke. "Plotly-resampler: Effective visual analytics for large time series". In: *2022 IEEE Visualization and Visual Analytics (VIS)*. IEEE. Oklahoma City, USA, 2022, pp. 21–25

4. Jeroen Van Der Donckt*, **Jonas Van Der Donckt***, Michael Rademaker, and Sofie Van Hoecke. "MinMaxLTTB: Leveraging MinMax-Preselection to Scale LTTB". in: *2023 IEEE Visualization and Visual Analytics (VIS)*. IEEE. Melbourne, Australia, 2023

5. **Jonas Van Der Donckt***, Jeroen Van Der Donckt*, and Sofie Van Hoecke. "Left-Right Swapping and Upper-Lower Limb Pairing for Robust Multi-Wearable Workout Activity Detection". In: *ACM International Joint Conference on Pervasive and Ubiquitous Computing*. UbiComp '24. Melbourne VIC, Australia, 2024, pp. 545–550

6. Jeroen Van Der Donckt*, **Jonas Van Der Donckt***, and Sofie Van Hoecke. "Magnitude and Rotation Invariant Detection of Transportation Modes with Missing Data Modalities". In: *ACM International Joint Conference on Pervasive*

and Ubiquitous Computing. UbiComp '24. Melbourne VIC, Australia, 2024, pp. 597–602

1.8.3 Open-Source Libraries

1. `plotly-resampler`: scalable time series visualization with plotly.py
   Git: https://github.com/predict-idlab/plotly-resampler
   Creators: **Jonas Van Der Donckt***, Jeroen Van Der Donckt*, Emiel Deprost
   Stars: 1097, Downloads: 10.8M

2. `tsflex`: flexible time series feature extraction & processing
   Git: https://github.com/predict-idlab/tsflex
   Creators: **Jonas Van Der Donckt***, Jeroen Van Der Donckt*, Emiel Deprost
   Stars: 407, Downloads: 68k

References

[1] Laurent Oudre. *Machine Learning for Time Series Lecture 2: Feature Extraction and Selection*. Feb. 2023. URL: https://www.laurentoudre.fr/ast/AST2.pdf.

[2] Philip Schmidt, Attila Reiss, Robert Dürichen, and Kristof Van Laerhoven. "Wearable-based affect recognition—A review". In: *Sensors* 19.19 (2019), p. 4079.

[3] Wolfgang Aigner, Silvia Miksch, Wolfgang Müller, Heidrun Schumann, and Christian Tominski. "Visualizing time-oriented data—a systematic view". In: *Computers & Graphics* 31.3 (2007), pp. 401–409.

[4] Farhad Balali, Jessie Nouri, Adel Nasiri, Tian Zhao, Farhad Balali, Jessie Nouri, Adel Nasiri, and Tian Zhao. "Data analytics". In: *Data Intensive Industrial Asset Management: IoT-based Algorithms and Implementation* (2020), pp. 105–113.

[5] Sakinah SJ Alhadad. "Visualizing data to support judgement, inference, and decision making in learning analytics: Insights from cognitive psychology and visualization science". In: *Journal of Learning Analytics* 5.2 (2018), pp. 60–85.

[6] Jessica Lin, Eamonn Keogh, and Stefano Lonardi. "Visualizing and discovering non-trivial patterns in large time series databases". In: *Information visualization* 4.2 (2005), pp. 61–82.

[7] Steven Finlay. *Predictive analytics, data mining and big data: Myths, misconceptions and methods*. Springer, 2014.

[8] Jessica Lin, Eamonn Keogh, and Stefano Lonardi. "Visualizing and Discovering Non-Trivial Patterns in Large Time Series Databases". en. In: *Information Visualization* 4.2 (June 2005). ZSCC: 0000178, pp. 61–82. ISSN: 1473-8716, 1473-8724. DOI: 10.1057/palgrave.ivs.9500089. URL: http://journals.sagepub.com/doi/10.1057/palgrave.ivs.9500089 (visited on 03/11/2022).

[9] Ben Shneiderman. "The eyes have it: A task by data type taxonomy for information visualizations". In: *The craft of information visualization*. Elsevier, 2003, pp. 364–371.

[10] Rajeev Agrawal, Anirudh Kadadi, Xiangfeng Dai, and Frederic Andres. "Challenges and opportunities with big data visualization". In: *Proceedings of the 7th International Conference on Management of computational and collective intElligence in Digital EcoSystems*. 2015, pp. 169–173.

[11] Edward Angel and Dave Shreiner. *Interactive Computer Graphics with WebGL*. 7th. Addison-Wesley Professional, 2014. ISBN: 0133574849.

[12] Syed Mohd Ali, Noopur Gupta, Gopal Krishna Nayak, and Rakesh Kumar Lenka. "Big data visualization: Tools and challenges". In: *2016 2nd International conference on contemporary computing and informatics (IC3I)*. IEEE. 2016, pp. 656–660.

[13] Sveinn Steinarsson. "Downsampling Time Series for Visual Representation". en. MA thesis. University of Iceland, 2013. DOI: http://hdl.handle.net/1946/15343. URL: http://hdl.handle.net/1946/15343.

[14] Zhicheng Liu and Jeffrey Heer. "The effects of interactive latency on exploratory visual analysis". In: *IEEE transactions on visualization and computer graphics* 20.12 (2014), pp. 2122–2131.

[15] Holoviz-community. *Datashader, quickly and accurately render even the largest data*. https://github.com/holoviz/datashader.

[16] Jean-Luc R Stevens, Philipp Rudiger, and James A Bednar. "HoloViews: building complex visualizations easily for reproducible science". In: *Proceedings of the 14th Python in Science Conference*. Citeseer. 2015, pp. 61–69.

[17] M. A. Breddels. "Interactive (statistical) visualisation and exploration of a billion objects with vaex". en. In: *Proceedings of the International Astronomical Union* 12.S325 (Oct. 2016), pp. 299–304. ISSN: 1743-9213, 1743-9221. DOI: 10.1017/S1743921316012795. URL: https://www.cambridge.org/core/product/identifier/S1743921316012795/type/journal_article (visited on 03/28/2022).

[18] Kexin Rong and Peter Bailis. "ASAP: prioritizing attention via time series smoothing". In: *arXiv preprint arXiv:1703.00983* (2017).

[19] Uwe Jugel, Zbigniew Jerzak, Gregor Hackenbroich, and Volker Markl. "VDDA: automatic visualization-driven data aggregation in relational databases". en. In: *The VLDB Journal* 25 (Feb. 2016), pp. 53–77. ISSN: 1066-8888, 0949-877X. DOI: 10.1007/s00778-015-0396-z. URL: http://link.springer.com/10.1007/s00778-015-0396-z (visited on 03/03/2023).

[20] Uwe Jugel, Zbigniew Jerzak, Gregor Hackenbroich, and Volker Markl. "M4: a visualization-oriented time series data aggregation". In: *Proceedings of the VLDB Endowment* 7.10 (2014), pp. 797–808.

[21] David H Douglas and Thomas K Peucker. "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature". In: *Cartographica: the international journal for geographic information and geovisualization* 10.2 (1973), pp. 112–122.

[22] Maheswari Visvalingam and James D Whyatt. "Line generalisation by repeated elimination of points". In: *The cartographic journal* 30.1 (1993), pp. 46–51. DOI: 10.1179/000870493786962263.

[23] Amaia Gil, Marco Quartulli, Igor G Olaizola, and Basilio Sierra. "Towards smart data selection from time series using statistical methods". In: *IEEE Access* 9 (2021), pp. 44390–44401. DOI: 10.1109/ACCESS.2021.3066686.

[24] Paul Rosen, Ashley Suh, Christopher Salgado, and Mustafa Hajij. "TopoLines: Topological Smoothing for Line Charts". In: *arXiv preprint arXiv:1906.09457* (2019).

[25] Tak-chung Fu, Ying-kit Hung, and Fu-lai Chung. "Improvement algorithms of perceptually important point identification for time series data mining". In: *2017 IEEE 4th International Conference on Soft Computing & Machine Intelligence (ISCMI)*. IEEE. 2017, pp. 11–15.

[26] Jônatas Davi Paganini. *Downsampling in the database: How data locality can improve data analysis*. www.timescale.com/blog. Feb. 2023. URL: https://www.timescale.com/blog/downsampling-in-the-database-how-data-locality-can-improve-data-analysis/.

[27] Carl H Lubba, Sarab S Sethi, Philip Knaute, Simon R Schultz, Ben D Fulcher, and Nick S Jones. "catch22: CAnonical Time-series CHaracteristics: Selected through highly comparative time-series analysis". In: *Data Mining and Knowledge Discovery* 33.6 (2019), pp. 1821–1852.

[28] Maximilian Christ, Nils Braun, Julius Neuffer, and Andreas W Kempa-Liehr. "Time series feature extraction on basis of scalable hypothesis tests (tsfresh–a python package)". In: *Neurocomputing* 307 (2018), pp. 72–77.

[29] Aleksandr Ometov, Viktoriia Shubina, Lucie Klus, Justyna Skibińska, Salwa Saafi, Pavel Pascacio, Laura Flueratoru, Darwin Quezada Gaibor, Nadezhda Chukhno, Olga Chukhno, et al. "A survey on wearable technology: History, state-of-the-art and current challenges". In: *Computer Networks* 193 (2021), p. 108074.

[30] Ruixuan Dai, Thomas Kannampallil, Seunghwan Kim, Vera Thornton, Laura Bierut, and Chenyang Lu. "Detecting mental disorders with wearables: A large cohort study". In: *Proceedings of the 8th ACM/IEEE Conference on Internet of Things Design and Implementation*. 2023, pp. 39–51.

[31] Lydia Wheeler, Vaclav Kremen, Cole Mersereau, Guillermo Ornelas, Taruna Yadav, Devon Cormier, Allyson Derry, Andrea Duque Lopez, Kevin McQuown, Vladimir Sladky, et al. "Automatic responsiveness testing in epilepsy with wearable technology: The ARTiE Watch". In: *Epilepsia* 66.1 (2025), pp. 104–116.

[32] Neil S Zheng, Jeffrey Annis, Hiral Master, Lide Han, Karla Gleichauf, Jack H Ching, Melody Nasser, Peyton Coleman, Stacy Desine, Douglas M Ruderfer, et al. "Sleep patterns and risk of chronic disease as measured by long-term monitoring with commercial wearable devices in the All of Us Research Program". In: *Nature Medicine* 30.9 (2024), pp. 2648–2656.

[33] Joanna Smith and Helen Noble. "Bias in research". In: *Evidence-based nursing* 17.4 (2014), pp. 100–101.

[34] Ann M Berger, Kimberly K Wielgus, Stacey Young-McCaughan, Patricia Fischer, Lynne Farr, and Kathryn A Lee. "Methodological challenges when using actigraphy in research". In: *Journal of pain and symptom management* 36.2 (2008), pp. 191–199.

[35] Michael Haenlein and Andreas Kaplan. "A brief history of artificial intelligence: On the past, present, and future of artificial intelligence". In: *California management review* 61.4 (2019), pp. 5–14.

[36] Batta Mahesh. "Machine learning algorithms-a review". In: *International Journal of Science and Research (IJSR).[Internet]* 9.1 (2020), pp. 381–386.

[37] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[38] Mohd Javaid, Abid Haleem, Ravi Pratap Singh, Rajiv Suman, and Shanay Rab. "Significance of machine learning in healthcare: Features, pillars and applications". In: *International Journal of Intelligent Networks* 3 (2022), pp. 58–73.

[39] Oscar D Lara and Miguel A Labrador. "A survey on human activity recognition using wearable sensors". In: *IEEE communications surveys & tutorials* 15.3 (2012), pp. 1192–1209.

[40] Julia Amann, Alessandro Blasimme, Effy Vayena, Dietmar Frey, and Vince I Madai. "Explainability for artificial intelligence in healthcare: a multidisciplinary perspective". In: *BMC medical informatics and decision making* 20.1 (2020), pp. 1–9.

[41] Ravid Shwartz-Ziv and Amitai Armon. "Tabular data: Deep learning is not all you need". In: *Information Fusion* 81 (2022), pp. 84–90.

[42] Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. "Why do tree-based models still outperform deep learning on typical tabular data?" In: *Advances in neural information processing systems* 35 (2022), pp. 507–520.

[43] Alex A Freitas. "Comprehensible classification models: a position paper". In: *ACM SIGKDD explorations newsletter* 15.1 (2014), pp. 1–10.

[44] Yoshua Bengio, Aaron Courville, and Pascal Vincent. "Representation learning: A review and new perspectives". In: *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2013), pp. 1798–1828.

[45] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[46] Sinno Jialin Pan and Qiang Yang. "A survey on transfer learning". In: *IEEE Transactions on knowledge and data engineering* 22.10 (2009), pp. 1345–1359.

[47] Pedro Domingos. "A Few Useful Things to Know about Machine Learning". In: *Commun. ACM* 55.10 (Oct. 2012), pp. 78–87. ISSN: 0001-0782. DOI: 10.1145/2347736.2347755. URL: https://doi.org/10.1145/2347736.2347755.

[48] Duncan McElfresh, Sujay Khandagale, Jonathan Valverde, Vishak Prasad C, Ganesh Ramakrishnan, Micah Goldblum, and Colin White. "When do neural nets outperform boosted trees on tabular data?" In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 76336–76369.

[49] Yue Lu, Renjie Wu, Abdullah Mueen, Maria A Zuluaga, and Eamonn Keogh. "Matrix profile XXIV: scaling time series anomaly detection to trillions of datapoints and ultra-fast arriving data streams". In: *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*. 2022, pp. 1173–1182.

[50] Jeroen Van Der Donckt, Jonas Van Der Donckt, Emiel Deprost, Nicolas Vandenbussche, Michael Rademaker, Gilles Vandewiele, and Sofie Van Hoecke. "Do not sleep on traditional machine learning: Simple and interpretable techniques are competitive to deep learning for sleep scoring". In: *Biomedical Signal Processing and Control* 81 (2023), p. 104429.

[51] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. "M5 accuracy competition: Results, findings, and conclusions". In: *International Journal of Forecasting* 38.4 (2022), pp. 1346–1364.

[52] LJ Stovner, K Hagen, Rf Jensen, Z Katsarava, RB Lipton, AI Scher, TJ Steiner, and JA Zwart. "The global burden of headache: a documentation of headache prevalence and disability worldwide". In: *Cephalalgia* 27.3 (2007), pp. 193–210.

[53] "Headache Classification Committee of the International Headache Society (IHS) The International Classification of Headache Disorders, 3rd edition". en. In: *Cephalalgia* 38.1 (Jan. 2018), pp. 1–211. ISSN: 0333-1024, 1468-2982. DOI: 10.1177/0333102417738202. URL: http://journals.sagepub.com/doi/10.1177/0333102417738202 (visited on 01/31/2024).

[54] Fayyaz Ahmed. "Headache disorders: differentiating and managing the common subtypes". In: *British journal of pain* 6.3 (2012), pp. 124–132.

[55] Koen Paemeleire, Nicolas Vandenbussche, and Richard Stark. "Migraine without aura". In: *Handbook of Clinical Neurology* 198 (2023), pp. 151–167.

[56] "Headache Classification Committee of the International Headache Society (IHS) The International Classification of Headache Disorders, 3rd edition". en. In: *Cephalalgia* 38.1 (Jan. 2018), pp. 1–211. ISSN: 0333-1024, 1468-2982. DOI: 10.1177/0333102417738202. URL: http://journals.sagepub.com/doi/10.1177/0333102417738202 (visited on 01/31/2024).

[57] Arne May, Todd J Schwedt, Delphine Magis, Patricia Pozo-Rosich, Stefan Evers, and Shuu-Jiun Wang. "Cluster headache". In: *Nature reviews Disease primers* 4.1 (2018), pp. 1–17.

[58] Diana Yi-Ting Wei, Jonathan Jia Yuan Ong, and Peter James Goadsby. "Cluster headache: epidemiology, pathophysiology, clinical features, and diagnosis". In: *Annals of Indian Academy of Neurology* 21.Suppl 1 (2018), S3–S8.

[59] JHM Tulen, DL Stronks, JBJ Bussmann, Lolke Pepplinkhuizen, and Jan Passchier. "Towards an objective quantitative assessment of daily functioning in migraine: a feasibility study". In: *Pain* 86.1 (2000), pp. 139–149.

[60] Daniel G Rogers, Dale S Bond, John P Bentley, and Todd A Smitherman. "Objectively measured physical activity in migraine as a function of headache activity". In: *Headache: The Journal of Head and Face Pain* 60.9 (2020), pp. 1930–1938.

[61] H Kikuchi, K Yoshiuchi, K Ohashi, Y Yamamoto, and A Akabayashi. "Tension-type headache and physical activity: an actigraphic study". In: *Cephalalgia* 27.11 (2007), pp. 1236–1243.

[62] Mathias De Brouwer, Nicolas Vandenbussche, Bram Steenwinckel, Marija Stojchevska, Jonas Van Der Donckt, Vic Degraeve, Jasper Vaneessen, Filip De Turck, Bruno Volckaert, Paul Boon, et al. "mBrain: towards the continuous follow-up and headache classification of primary headache disorder patients". In: *BMC Medical Informatics and Decision Making* 22.1 (2022), p. 87.

[63] James Walker, Rita Borgo, and Mark W Jones. "Timenotes: a study on effective chart visualization and interaction techniques for time-series data". In: *IEEE transactions on visualization and computer graphics* 22.1 (2015), pp. 549–558.

[64] James A. Bednar and Julia Signell. *Common plotting pitfalls that get worse with large data*. github.com/holoviz/datashader/examples/user_guide/1_Plotting_Pitfalls.ipynb.

[65] Seungeun Chung, Chi Yoon Jeong, Jeong Mook Lim, Jiyoun Lim, Kyoung Ju Noh, Gague Kim, and Hyuntae Jeong. "Real-world multimodal lifelog dataset for human behavior study". In: *ETRI Journal* 44.3 (2022), pp. 426–437.

2

Plotly-Resampler: Effective Visual Analytics for Large Time Series

Chapter 1 highlighted the critical role of visualization in data analytics, particularly for time series analysis. This chapter addresses **RG1-A** by tackling the challenges of scalable time series line chart visualization, introducing Plotly-Resampler—a production-ready tool designed for efficient and interactive exploration of large datasets. Additionally, the flexible design and interface of Plotly-Resampler establishes it as a research platform for investigating time series data aggregation methods. With over 10 million installations to date, Plotly-Resampler has proven its effectiveness across a wide range of research and industry applications.

My contributions can be summarized as follows:

- Identifying key requirements for effective time series visual analytics.
- Mapping existing Python visualization libraries to these requirements.
- Creation of Plotly-Resampler to meet these requirements through:
  - Iterative prototyping.
  - Software design & development.
  - Packaging, testing, and documenting the code.
  - Integrations with other tools (e.g., vscode, Jupyter notebooks, Google Colab).

  - Ongoing maintenance, updates, and bug fixes.

- Benchmarking Plotly-Resampler's scalability/performance to ensure RG1-A is achieved.

Plotly-Resampler: Effective Visual Analytics for Large Time Series

Jonas Van Der Donckt [1], Jeroen Van Der Donckt [1], Emiel Deprost, and Sofie Van Hoecke

Published in proceedings of the "2022 IEEE Conference of Visualization and Visual Analytics. Oklahoma City (USA), 2022, pp. 21-25"

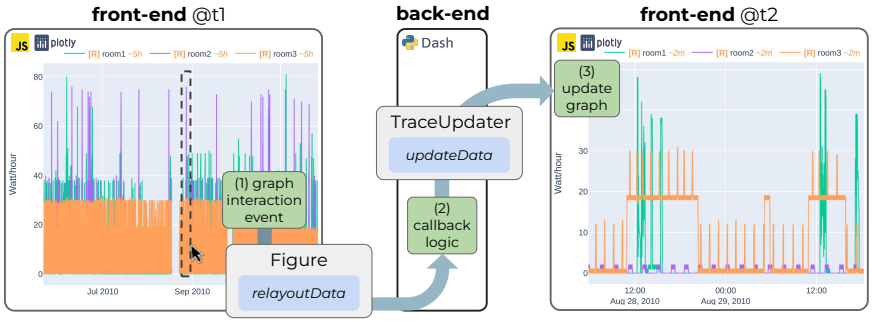


Figure 2.1: High-level overview of Plotly-Resampler's dynamic aggregation functionality (see gif). A front-end graph interaction event (i.e, zoom event) (1) triggers a callback, sending the layout-change (i.e., *relayoutData*) to the back-end. In the Dash back-end (2), the *relayoutData* is processed to perform data aggregation for the new regions of interest. Finally, only the to-be-updated data (i.e., *updateData*) is sent to the front-end, on its end triggering a graph update (3). The presence of an [R] legend prefix indicates that an aggregation of the data is shown. The $\sim$<*time*> suffix in the legend represents an estimate of the aggregation bin size.

Abstract Visual analytics is arguably the most important step in getting acquainted with your data. This is especially the case for time series, as this data type is hard to describe and cannot be fully understood when using for example summary statistics. To realize effective time series visualization, four requirements have to be met: a tool should be (1) interactive, (2) scalable to millions of data points, (3) integrable in conventional data science environments, and (4) highly configurable. We observe that open-source Python visualization toolkits empower data scientists in most visual analytics tasks, but lack the combination of scalability and interactivity to realize effective time series visualization. As a means to facilitate these requirements, we created Plotly-Resampler, an open-source Python library. Plotly-Resampler is an add-on for Plotly's Python bindings, enhancing line chart scalability on top of an interactive toolkit by aggregating the underlying data depending on the current graph view. Plotly-Resampler is built to be snappy, as the reactivity of a tool qualitatively affects how analysts visually explore and analyze data. A benchmark task highlights how our

[1] Contributed equally

toolkit scales better than alternatives in terms of number of samples and time series. Additionally, Plotly-Resampler's flexible data aggregation functionality paves the path towards researching novel aggregation techniques. Plotly-Resampler's integrability, together with its configurability, convenience, and high scalability, allows to effectively analyze high-frequency data in your day-to-day Python environment.

2.1 Introduction

Data wrangling is the process of iterative data exploration and transformation, enabling downstream tasks such as modeling and data analysis. In the data wrangling process, and even in the whole data science pipeline, visual analytics has proven to be a crucial component. For example, visualizations can be utilized to assess the quality of your processing, validate the alignment of your data, obtain insights, and even analyze model predictions. After all, the human eye has frequently been advocated as the ultimate data mining tool [1].

However, effective visualization of time series remains challenging as this data type is less intuitive to grasp compared to for example images or text. Both the temporal and multivariate aspects of time series data should be captured in meaningful visualizations. It has been shown that for most time series analysis tasks, simple visualizations (e.g., line-based charts) are sufficient, compared to more complex or use-case specific approaches [2].

Due to decreasing costs in both storage and sensors, large time series data is becoming more common. We observe that the current Python visualization landscape struggles to effectively handle such large datasets [3]. More specifically, some tools lack graph interactivity, which is necessary to effectively explore time series, whereas other interactive tools become slow or unresponsive with such large quantities of data. To tackle this problem we created Plotly-Resampler, an add-on for Plotly's Python bindings, enabling interactive visualizations of large sequences. Plotly-Resampler complies with the visual information seeking mantra: *"Overview first, zoom and filter, then details-on-demand"* [4]. Correspondingly, Plotly-Resampler is built to be snappy, as the speed of data retrieval qualitatively affects how analysts visually explore and analyze their data [5].

The contribution of this chapter is threefold:

- We **introduce Plotly-Resampler**, an open-source Python toolkit that enables interactive and scalable time series visualization. This is achieved by performing under-the-hood data aggregation, depending on the current graph view.
- We **demonstrate** the **practical usefulness** of Plotly-Resampler for data wrangling and downstream tasks with a real-world use case and show the minimal code overhead of our package.

- We **position and benchmark** Plotly-Resampler against alternatives and highlight its strengths and limitations.

2.2 Related Work

According to the Kaggle 2021 survey, (IPython-)notebook-based environments are the go-to tools for data scientists [6]. Such a notebook-based format drives exploration, which is crucial in every step of the data science process. Their form of interactive computing enables users to execute code, observe, modify, and repeat in an iterative conversation between analyst and data [7].

Based on the observations of Bikakis [8] and Perkel [7], we list four requirements for an effective time series visualization tool;

1. *Provide interactivity*: To enable effective exploration, and thus comply with the information seeking mantra, visualizations need to support functionalities like zooming, panning, showing information on hover, and (de)selecting specific traces [4, 2].
2. *Scalable to large datasets*: Visualizations should not become slow or unresponsive with large data, where millions of observations from multiple modalities are common.
3. *Integrable in conventional data-science environments*: The toolkit should be easy to integrate with existing tools and workflows, for example IPython-based notebooks running on a remote server.
4. *Configurable and convenient*: Having a tool that is both easy to use and highly configurable enables broad applicability. For example, an effective tool should conveniently allow visualizing multivariate data on a shared time axis.

In the last two decades, a lot of research has been done on time series visualization techniques [9, 10, 1, 11, 12, 13, 14, 15]. This work focused on describing and comparing time series visualization approaches. However, when we position this research with regard to practical applicability, a common trend is observed; these contributions do not seem to focus on code availability nor integration with either visualization packages or data wrangling environments. This severely limits their applicability, and also makes it hard to reproduce results or to assess other aspects such as scalability and interactivity. Only Stopar et al. [14] and Kincaid et al. [9] provided benchmarking information about their methodology, whereas only Morrow et al. [13] made their code publicly available.

Given the lack of accessible research outcomes on the one hand and the Kaggle survey results on the other hand, we limit our comparison to open-source Python toolkits. Table 2.1 shows that none of those selected toolkits meet all the aforementioned requirements. We observe that there is a consistent trade-off between interactivity and

Table 2.1: Requirement assessment for effective visualization of large time series, applied to popular open-source libraries.

		Bokeh	Plotly	Matplotlib	Holoviews
1	Interactivity	+	+	-	+
2	Scalability	-	-	±	±
3	Integrability	+	+	+	+
4	Configurability	+	+	+	+

scalability. Matplotlib [16] is somewhat scalable to larger data sizes, but has limited interactivity. Both Bokeh [17] and Plotly [18] provide extensive interactivity by using a JavaScript front-end. However, loading large quantities of data in this front-end hinders responsiveness, thus restricting the scalability of these two libraries. Finally, HoloViews approaches visualization by providing a set of data structures that pair your data with a small amount of metadata [19]. Those data structures are then rendered by a separate plotting system, e.g., Bokeh or Plotly, to visualize data interactively. Just as Plotly-Resampler does, HoloViews provides data aggregation functionality through its Datashader bindings. However, these are not optimized for multivariate line chart aggregation, limiting the convenience and scalability of HoloViews.

As a result, users tend to settle for suboptimal approaches such as downsampling or selecting smaller parts of the data before creating interactive visualizations. On the one hand, static data aggregation (i.e., downsampling) does not show the full data and can induce artifacts such as aliasing [20]. On the other hand, visualizing smaller parts of the data results in a shattered interaction process, as it is substantially harder to associate the overall data with these sporadic local inspections.

It is our first-hand experience with the above shortcomings that sparked the creation of Plotly-Resampler.

2.3 Plotly-Resampler Toolkit

Visualizations are limited by the size of the canvas (i.e., pixel density of the screen). As such, a common issue when visualizing large quantities of data is over-plotting, where multiple data points are rendered on top of each other [21, 10]. Furthermore, front-end rendering and responsiveness slow down significantly when more data points are portrayed. These problems with visualizing large quantities of data can be tackled, to some extent, by either employing series-wise data aggregation (as Plotly-Resampler does) or using density-wise aggregation (as Datashader [22] does).

Datashader was developed to represent large amounts of data for which a shared density color coding is used, making it the most effective when there is a single, large data modality (e.g., a map, a point cloud). However, when there are multiple distinct

modalities in the data, each requires a distinguishable color density coding, resulting in (again) an over-plotting issue. Unfortunately, time series often consists of distinct modalities, rendering this technique unsuitable.

Alternatively, series-wise data aggregation controls over-plotting by reducing the number of rendered data points. This makes multivariate visualizations scalable at the cost of possibly inducing information loss [23]. However, a key insight into visualizing such large amounts of data is that not all data points are equally important. The largest triangle three buckets (LTTB) aggregation technique exploits this insight to effectively select data points [24].

From these observations, it is clear that series-wise data aggregation is most suitable for effective time series visualization. Next to this, in contrast with a custom plotting language (e.g., HoloViews), we opted to build upon Plotly. By complying with the *"Don't reimplement the wheel"*-mantra, our toolkit leverages all of Plotly's functionality, i.e., convenience, integrability, interactivity & high configurability, while at the same time resulting in a smaller, more manageable codebase. The realization of Plotly-Resampler's goal, i.e., scalable data aggregation, is shown in Figure 2.1.

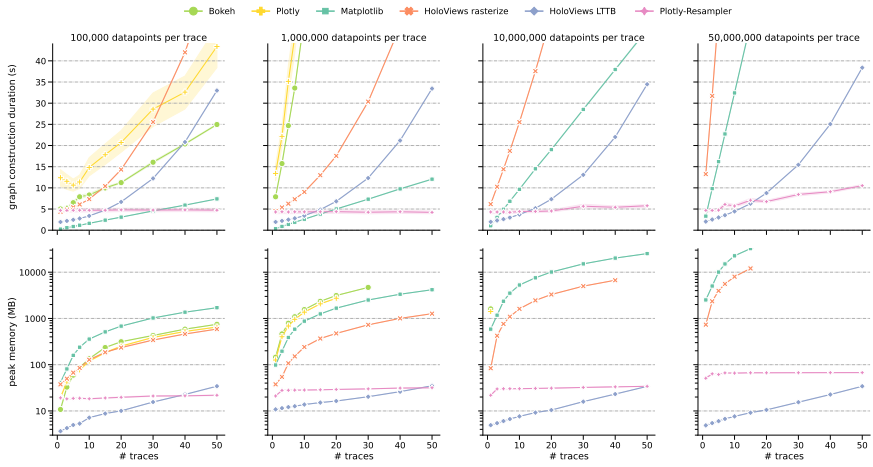


Figure 2.2: Benchmark results for a line-graph visualization task (code [25]). The top charts display the average duration of constructing and rendering the graph. The bottom charts indicate the peak memory usage. The columns indicate the data size per signal, thus showing a trend when scaling to larger datasets. In each chart, the number of visualized traces (i.e., lines) is represented on the x-axis. Each toolkit configuration was benchmarked until the duration exceeded 2 minutes. Dynamic aggregation is realized in `HoloViews LTTB` by using Plotly-Resampler's LTTB aggregation functionality, whereas `HoloViews rasterize` uses the built-in Datashader-rasterize function.

2.3.1 Features

With Plotly-Resampler we aim to contribute a qualitative Python package to the data visualization community for effective time series analysis. For that reason, Plotly-Resampler is listed on PyPi, and can thus be installed via `pip install plotly-resampler`. Plotly-Resampler's source code is available on GitHub [25], encouraging input from the community (e.g., contributions via pull requests, issues for discovered bugs, or feature requests). For detailed information, users can consult the documentation along with several code examples demonstrating how to apply Plotly-Resampler for various visualization tasks. To assure Plotly-Resampler's quality, the functionality is validated with a CI/CD testing pipeline, which tests the toolkit's functionality as a web-app with Selenium [26] and validates the content of the transferred packets (triggered by graph callbacks) [27].

The scalability of Plotly-Resampler is realized by optimizing several aspects. To perform data updates, the default Dash graph component requires re-sending all graph data to the front-end, resulting in unnecessary data and computation overhead. This behavior causes an increased callback latency, thus impacting responsiveness. To overcome this problem, we created TraceUpdater [28], a Dash component that minimizes callback latency by only sending to-be-updated trace data to the front-end. Additionally, the back-end data access latency is negligible as the back-end data is stored in memory. Finally, code profiling allowed us to look for bottlenecks and validate that view-based operations (i.e. pass by reference) were applied to the data where possible, as such operations do not induce large memory and runtime overheads.

Other features of the Plotly-Resampler toolkit are a direct consequence of its flexibility. For example, Plotly-Resampler is not limited to numeric data types, but also supports categorical and boolean series. Furthermore, there is a high configurability of the functionality, e.g., the data aggregation technique, time series gap detection, or how the web-app should be served. As Plotly-Resampler is built in the Plotly-Dash ecosystem, it also integrates seamlessly with Dash web applications.

2.3.2 Benchmarks

Given the prevalence of large and high-frequency time series, the scalability capabilities of eligible toolkits were assessed with a line-graph visualization benchmark. The profiling is realized using the `VizTracer` [29] package with the `VizPlugins` add-on. The benchmarking code is available on GitHub [25], encouraging reproducibility. We further refer to this README [30] for detailed information regarding the benchmarking procedure.

Figure 2.2 depicts the profiling results. The top charts represent the total time to construct and render the visualization. As expected, the graph construction time increases with the number of data points (i.e., columns) and number of traces (i.e., x-axis), however not all at the same pace. `Plotly-Resampler` clearly scales better

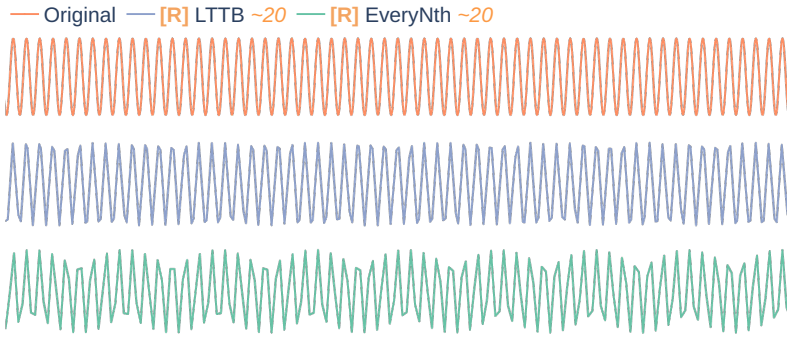


Figure 2.3: Influence of data aggregation method on aliasing. The middle graph uses the LTTB aggregation algorithm, whereas the bottom graph employs the naive `EveryNth` algorithm.

in terms of data points and number of traces. `Bokeh` and `Plotly` (without any aggregation functionality) scale the worst, as they are heavily affected by the dataset size. The visualization time of HoloViews-based approaches scales exponentially in terms of the number of visualized traces (i.e., x-axis), rendering them unsuitable for large multivariate visualizations.

The bottom charts indicate the peak memory usage. We observe that both `Plotly-Resampler` and `HoloViews LTTB` scale better. These two approaches use less than 100 MB for the largest configuration (right column), whereas `Matplotlib` and `HoloViews rasterize` exceed 10 GB.

When dealing with more than 10,000,000 samples per series and more than 15 traces, `Plotly-Resampler` emerges as the most viable toolkit regarding graph construction time and memory usage. Such scenarios frequently arise when handling high-resolution data from diverse modalities, such as clinical data.

2.3.3 Limitations

Data aggregation is a form of resampling, rendering it susceptible to downsampling artifacts, in particular, aliasing [20]. Aliasing is more prevalent when a naive aggregation algorithm, such as `EveryNth` is used (where data points are aggregated by sampling with a fixed interval N). More advanced aggregation methods seem to be less prone to such artifacts. For example, LTTB samples data points in bins with the goal of maximizing the triangular surface of surrounding bins [24]. As an illustration, Figure 2.3 highlights the influence of the aggregation method on aliasing artifacts. We observe that for the shown region of interest, both methods display artifacts. As expected, `EveryNth` aggregation (bottom graph) results in a more distorted view than `lttb` (middle graph). Plotly-Resampler attempts to minimize the aliasing issue by employing an efficient heuristic of LTTB as the default aggregation algorithm and by indicating when data aggregation is performed. This indication is realized by displaying the [R] legend prefix

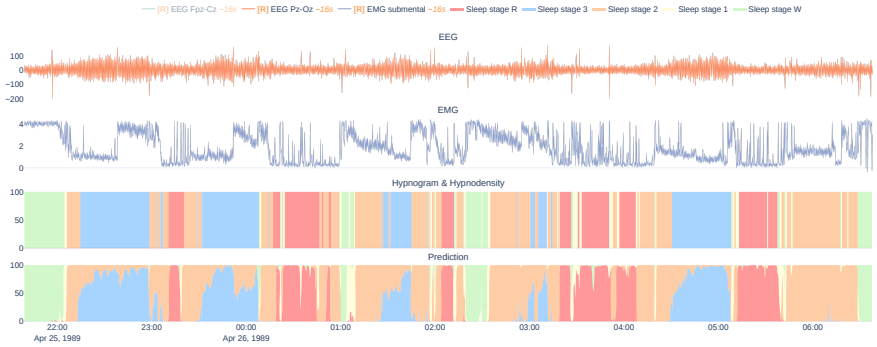


Figure 2.4: Sleep stage classification example, demonstrating the joined visualization of raw sensor data with probabilistic predictions. The first two rows show the raw data, i.e., EEG signals which are sampled at 100Hz, and an EMG signal which is sampled at 1Hz, respectively. The third row shows the sleep stages, also called a hypnogram on the right y-axis (black line), and the hypnodensity, i.e., a probabilistic hypnogram, on the left y-axis. The final row shows probabilistic hypnodensity predictions of a machine learning pipeline, empowered by `tsflex` [31].

as well as an estimate of the aggregation bin-size as legend suffix.

Plotly-Resampler supports various data types, such as numeric, categorical, and boolean time series data. However, it cannot consider all aspects involved when visualizing time-oriented data, e.g., time-bound text data. Aigner et al. [2] indicated that such different types of time-oriented data can only be visualized with dedicated methods. Hence, we argue that supporting such visualizations is out of scope for this toolkit.

Plotly-Resampler utilizes the `pandas.Series` [32] data container for storing high-frequency time series. This induces the limitation that all visualized data must fit in memory.

2.4 Toolkit Usage and Example Use Cases

Using Plotly-Resampler requires minimal code overhead. To add aggregation functionality to a Plotly figure, end-users only need to wrap their Plotly figure with the `FigureResampler` decorator and call `show_dash()` on that object. A minimal code example as a GitHub gist illustrates this usage [33].

2.4.1 Use Case: Sleep Scoring Model Analysis

Figure 2.4 highlights how Plotly-Resampler can be used for sleep polysomnography (PSG) data analysis [34]. Plotly-Resampler allows us to visualize a full night recording of electroencephalography (EEG), and electromyography (EMG) data with a machine learning model its predictions in one graph. From Figure 2.2 we observe that no other

tool is capable of serving the scalability to interactively visualize such large quantities of multivariate data. Plotly-Resampler gives a global overview in which we can see that the model probabilities are well-adjusted. Furthermore, we can obtain insights by zooming in on regions of interest, e.g., mispredictions, sudden changes in the raw data.

2.5 Conclusion

In this chapter, we present Plotly-Resampler, an open-source Python package enabling effective time series visualization. An effective visualization toolkit should be (1) interactive, (2) scalable, (3) integrable, and (4) configurable. We observe that no existing Python package meets these four requirements, as there is a consistent trade-off between interactivity and scalability. Plotly-Resampler tackles this trade-off by adding scalability to a toolkit that already meets requirements 1, 3, and 4 (namely Plotly). To achieve this scalability, Plotly-Resampler separates the visualization into a back-end and front-end. The back-end stores all the data, whereas the front-end shows an aggregated view of this data. The interactivity, i.e., dynamic data aggregation, is realized through optimized callbacks between the front-end and back-end.

Benchmarks indicate that Plotly-Resampler significantly outperforms existing alternatives when scaling to large, multivariate datasets. Moreover, we also highlight the applicability and convenience of Plotly-Resampler with a real-world use case.

Plotly-Resampler's flexible architecture enables it to serve as a platform for researching time series visualization approaches. In particular, it can be used to compare and research data aggregation techniques.

With Plotly-Resampler we aim to contribute a qualitative Python package to the data visualization community. Our goal is to enable effective time series visualization in your day-to-day Python environment.

References

[1] Jessica Lin, Eamonn Keogh, and Stefano Lonardi. "Visualizing and Discovering Non-Trivial Patterns in Large Time Series Databases". en. In: *Information Visualization* 4.2 (June 2005). ZSCC: 0000178, pp. 61–82. ISSN: 1473-8716, 1473-8724. DOI: 10.1057/palgrave.ivs.9500089. URL: http://journals.sagepub.com/doi/10.1057/palgrave.ivs.9500089 (visited on 03/11/2022).

[2] Wolfgang Aigner, Silvia Miksch, Wolfgang Müller, Heidrun Schumann, and Christian Tominski. "Visualizing time-oriented data—a systematic view". In: *Computers & Graphics* 31.3 (2007), pp. 401–409.

[3] Jovan Veljanoski. *Interactive and scalable dashboards with Vaex and Dash*. en. June 2020. URL: https://medium.com/plotly/interactive-and-scalable-dashboards-with-vaex-and-dash-9b104b2dc9f0 (visited on 04/26/2022).

[4] Ben Shneiderman. "The eyes have it: A task by data type taxonomy for information visualizations". In: *The craft of information visualization*. Elsevier, 2003, pp. 364–371.

[5] Sye-Min Chan, Ling Xiao, John Gerth, and Pat Hanrahan. "Maintaining interactivity while exploring massive time series". en. In: *2008 IEEE Symposium on Visual Analytics Science and Technology*. ZSCC: 0000087. Columbus, OH, USA: IEEE, Oct. 2008, pp. 59–66. ISBN: 978-1-4244-2935-6. DOI: 10.1109/VAST.2008.4677357. URL: http://ieeexplore.ieee.org/document/4677357/ (visited on 11/17/2021).

[6] Kaggle-inc. *Kaggle's State of Machine Learning and Data Science 2021.pdf*. English. survey. Oct. 2021. URL: https://www.kaggle.com/kaggle-survey-2021.

[7] Jeffrey M Perkel. "Why Jupyter is data scientists' computational notebook of choice". In: *Nature* 563.7732 (2018), pp. 145–147.

[8] Nikos Bikakis. "Big Data Visualization Tools". In: *Encyclopedia of Big Data Technologies*. Cham: Springer International Publishing, 2018, pp. 1–6. ISBN: 978-3-319-63962-8. DOI: 10.1007/978-3-319-63962-8_109-1. URL: https://doi.org/10.1007/978-3-319-63962-8_109-1.

[9] Robert Kincaid. "Signallens: Focus+ context applied to electronic time series". In: *IEEE Transactions on Visualization and Computer Graphics* 16.6 (2010), pp. 900–907.

[10] James Walker, Rita Borgo, and Mark W Jones. "Timenotes: a study on effective chart visualization and interaction techniques for time-series data". In: *IEEE transactions on visualization and computer graphics* 22.1 (2015), pp. 549–558.

[11] Jian Zhao, Fanny Chevalier, Emmanuel Pietriga, and Ravin Balakrishnan. "Exploratory analysis of time-series with chronolenses". In: *IEEE Transactions on Visualization and Computer Graphics* 17.12 (2011), pp. 2422–2431.

[12] Myoungsu Cho, Bohyoung Kim, Hee-Joon Bae, and Jinwook Seo. "Stroscope: Multi-scale visualization of irregularly measured time-series data". In: *IEEE transactions on visualization and computer graphics* 20.5 (2014), pp. 808–821.

[13] Bryce Morrow, Trevor Manz, Arlene E Chung, Nils Gehlenborg, and David Gotz. "Periphery plots for contextualizing heterogeneous time-based charts". In: *2019 IEEE visualization conference (VIS)*. IEEE. 2019, pp. 1–5.

[14] Luka Stopar, Primoz Skraba, Marko Grobelnik, and Dunja Mladenic. "Streamstory: Exploring multivariate time series on multiple scales". In: *IEEE transactions on visualization and computer graphics* 25.4 (2018), pp. 1788–1802.

[15] Jessica Lin, Eamonn Keogh, Stefano Lonardi, Jeffrey P Lankford, and Daonna M Nystrom. "Viztree: a tool for visually mining and monitoring massive time series databases". In: *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*. 2004, pp. 1269–1272.

[16] J. D. Hunter. "Matplotlib: A 2D graphics environment". In: *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. DOI: 10.1109/MCSE.2007.55.

[17] Holoviz-community. *Bokeh, Interactive Data Visualization in the browser, from Python*. https://github.com/bokeh/bokeh.

[18] Plotly-community. *Plotly, the interactive graphing library for Python*. https://github.com/plotly/plotly.py.

[19] Jean-Luc R Stevens, Philipp Rudiger, and James A Bednar. "HoloViews: building complex visualizations easily for reproducible science". In: *Proceedings of the 14th Python in Science Conference*. Citeseer. 2015, pp. 61–69.

[20] Moshe Mishali and Yonina C Eldar. "Sub-nyquist sampling". In: *IEEE Signal Processing Magazine* 28.6 (2011), pp. 98–124.

[21] James A. Bednar and Julia Signell. *Common plotting pitfalls that get worse with large data*. github.com/holoviz/datashader/examples/user_guide/1_Plotting_Pitfalls.ipynb.

[22] Holoviz-community. *Datashader, quickly and accurately render even the largest data*. https://github.com/holoviz/datashader.

[23] Rajeev Agrawal, Anirudh Kadadi, Xiangfeng Dai, and Frederic Andres. "Challenges and opportunities with big data visualization". en. In: *Proceedings of the 7th International Conference on Management of computational and collective intElligence in Digital EcoSystems*. Caraguatatuba Brazil: ACM, Oct. 2015, pp. 169–173. ISBN: 978-1-4503-3480-8. DOI: 10.1145/2857218.2857256. URL: https://dl.acm.org/doi/10.1145/2857218.2857256 (visited on 03/27/2022).

[24] Sveinn Steinarsson. "Downsampling Time Series for Visual Representation". MA thesis. University of Iceland, 2013.

[25] Jonas Van Der Donckt and Jeroen Van Der Donckt. *Plotly-Resampler benchmark code*. https://github.com/predict-idlab/plotly-resampler-benchmarks.

[26] Antawan Holmes and Marc Kellogg. "Automating functional tests using selenium". In: *AGILE 2006 (AGILE'06)*. IEEE. 2006, 6–pp.

[27] Will Keeling. *Selenium wire: Extends Selenium's Python bindings, enabling to inspect requests made by the browser*. https://github.com/wkeeling/selenium-wire.

[28] Jonas Van Der Donckt. *TraceUpdater: Dash component to update a dcc.Graph its traces via callbacks*. https://github.com/predict-idlab/trace-updater.

[29] Gao Tian. *VizTracer: a low-overhead logging, debugging, and profiling tool to trace and visualize Python code execution.* https://github.com/wkeeling/selenium-wire. 2020.

[30] Jonas Van Der Donckt. *Plotly-Resampler benchmark procedure and additional results*. https://github.com/predict-idlab/plotly-resampler-benchmarks/tree/6fda4b25f42bb034168f345a97a785562350e2cb/reports.

[31] Jonas Van Der Donckt, Jeroen Van Der Donckt, Emiel Deprost, and Sofie Van Hoecke. "tsflex: Flexible time series processing & feature extraction". In: *SoftwareX* 17 (2022), p. 100971.

[32] The pandas development team. *pandas-dev/pandas: Pandas*. Version latest. Feb. 2020. DOI: 10.5281/zenodo.3509134. URL: https://doi.org/10.5281/zenodo.3509134.

[33] Jonas Van Der Donckt. *Plotly-Resampler usage example*. https://gist.github.com/jonasvdd/828ea56dbfef21e9999392234c38a8af. 2022.

[34] Richard B Berry, Rohit Budhiraja, Daniel J Gottlieb, David Gozal, Conrad Iber, Vishesh K Kapur, Carole L Marcus, Reena Mehra, Sairam Parthasarathy, Stuart F Quan, et al. "Rules for scoring respiratory events in sleep: update of the 2007 AASM manual for the scoring of sleep and associated events: deliberations of the sleep apnea definitions task force of the American Academy of Sleep Medicine". In: *Journal of clinical sleep medicine* 8.5 (2012), pp. 597–619.

3

Shape-Preserving Subsampling for Scalable Line Chart Visualization: a Methodological Assessment

Chapter 2 introduced Plotly-Resampler as a scalable visualization tool by integrating shape-preserving subsampling algorithms into the widely used Plotly visualization framework. However, during its design and development, we identified a gap in literature regarding detailed comparisons of these subsampling algorithms and their effectiveness. This chapter bridges this gap by introducing an extensive open-source framework for evaluating the visual representativity and visual stability of shape-preserving subsampling algorithms. The findings result in an improved default configuration for Plotly-Resampler and establish a foundation for future algorithm development, thereby achieving **RG1-B**.

My contributions can be summarized as follows:

- Identifying the most prominent shape-preserving subsampling algorithms.
- Designing a reproducible framework to evaluate visual representativity by:
  - Defining parameters (e.g., rasterization back-end, canvas size, line widths, visualization toolkits, n_{out} range).
  - Identifying metrics to assess visual representativity.
- Introducing and evaluating the concept of "visual stability".

- Conducting reproducible, open-source experiments and interpreting results.
- Providing practical guidelines for end-users on subsampling algorithms.

Shape-Preserving Subsampling for Scalable Line Chart Visualization: a Methodological Assessment

Jonas Van Der Donckt [1], Jeroen Van Der Donckt [1], Michael Rademaker, and Sofie Van Hoecke

Submitted to "Machine Learning and Knowledge Extraction"

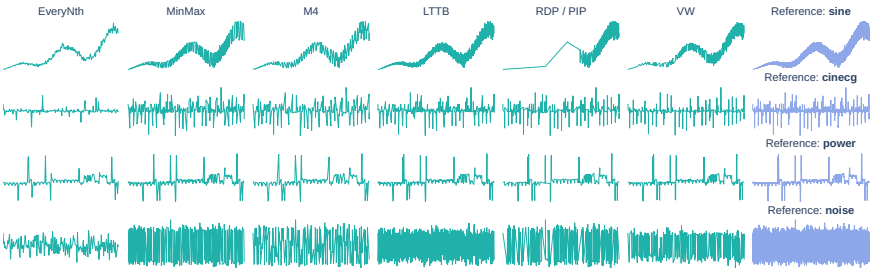


Figure 3.1: Visual comparison of six shape-preserving subsampling algorithms' ability to visually represent the original data. Each row corresponds to a unique template, with the rightmost column presenting a reference image, constructed from 50k data points. The six adjacent columns demonstrate the visual representations generated by different subsampling algorithms, all employing an equal number of output data points ($n_{out} = 200$). Visualization parameters used include: Plotly as toolkit, line-width set to 1 pixel, and anti-aliasing enabled. A GIF further demonstrates the above visualization with varying values for n_{out}.

Abstract Time series visualization plays a crucial role in identifying patterns and extracting insights across various domains, with line charts proving particularly effective for most tasks. However, as datasets continue to grow in size, visualizing them efficiently becomes challenging, necessitating scalable solutions. Downsampling, which reduces the amount of data points via aggregation or selection, is a well-established approach that aims to overcome this challenge. This work particularly focuses on shape-preserving subsampling, a collection of downsampling algorithms that select data points (or samples) from the original time series, aiming to preserve the visual characteristics of the original data. Despite the widespread adoption of subsampling in visualization platforms and time series databases, there is limited literature on the evaluation of these techniques. Therefore, we present an extensive, metrics-based evaluation methodology to assess and compare different subsampling algorithms for line chart visualization. Specifically, our methodology quantifies *visual representativeness* by assessing how well a subsampled time series line chart visually approximates the original data. Moreover, we introduce *visual stability* in a line chart context, analyzing the visual consistency of subsampling algorithms during updates (streaming) or interactions (panning and

[1] Contributed equally

zooming). We evaluated six subsampling algorithms across three open-source visualization toolkits, considering figure-drawing aspects such as line width. Following the analysis of our findings, we formulate a set of evidence-based guidelines for scalable line chart visualization with subsampling. The proposed evaluation methodology, along with the results and obtained insights from this study, are made openly accessible and establishes the necessary foundation for future research in this domain.

3.1 Introduction

Time series are ubiquitous, being commonly found in various fields, such as healthcare, finance, and engineering. This type of data often comprises complex information that cannot be fully understood through summary statistics alone, making visualization an essential tool for identifying patterns and extracting insights [1]. After all, the human eye has been frequently advocated as the ultimate data analytics tool [2]. Among the various visualization methods, line charts have proven to be particularly effective for analyzing time series in diverse applications [3].

The size of time series data has been growing continuously, often comprising millions to billions of data points [4]. However, handling and visualizing such large datasets poses significant challenges regarding storage, data transmission, and analysis (including visualization), necessitating scalable solutions [5, 6, 7].

Data aggregation emerged as the most established technique to efficiently visualize large data by either aggregating or selecting a subset of data points [8, 9, 3]. Particularly, shape-preserving subsampling algorithms, which aim to reduce the number of data points while preserving the overall shape of the original time series line chart, have become integral in many scalable time series visualization pipelines and database solutions [10].

However, despite the increasing prominence and integration of these shape-preserving subsampling techniques in database solutions and visualization platforms [7, 11, 10], there exists no formal analysis nor guidelines regarding these algorithms' visual representativeness relative to (i) different time series characteristics, such as variability, stationarity, and spikiness and (ii) line chart visualization parameters such as line width, line shape, or toolkit-specific rasterization. We believe that the profound evaluation of these algorithms has been under-studied due to the absence of a single universal metric as well as the interplay of visualization parameters with rendering.

As such, this work specifically focuses on evaluating shape-preserving subsampling algorithms in existing toolkits, focusing on their default configurations. Focusing on scalability, we pinpointed the most prominent algorithms, given their adoption in industry [12, 13]: EveryNth (selecting every n^{th} data point), MinMax, largest triangle three buckets (LTTB) [14], M4 [15], Ramer–Douglas–Peucker (RDP) [16]—also known as perceptually important points (PIP) [17], and Visvalingam–Whyatt (VW) [18].

To facilitate a formal analysis, we propose an extensive metrics-based method-

ology for evaluating the visual quality of shape-preserving subsampling algorithms, focusing on two key aspects: visual representativeness and visual stability. Visual representativeness assesses the extent to which a subsampled time series line chart visually approximates the rendered original data by employing image-space metrics, as such capturing both toolkit and line drawing characteristics. Visual stability quantifies the visual consistency between two slightly different aggregations by employing a data-space metric. This is particularly relevant when updating visualizations gradually, such as during data streaming, panning, or small zooms. These interactions have been demonstrated to be the most effective means for visually exploring time series data [19, 7, 20], making visual stability an essential aspect of time series subsampling for visualization.

To summarize, the contributions of this chapter are threefold. First, we present a robust and extensive metrics-based evaluation methodology for shape-preserving subsampling algorithms, quantifying visual representativeness and visual stability. Second, employing the proposed methodology, we assess the visual quality of six prominent downsampling algorithms, across three visualization toolkits, considering figure drawing parameters such as line width. The insights gained from this analysis are summarized into a set of guidelines for scalable time series visualization with subsampling. Lastly, to ensure reproducibility and facilitate the analysis and improvement of novel data aggregation algorithms, we provide a high-quality open-source implementation of the proposed framework along with the obtained results, available at github.com/predict-idlab/ts-datapoint-selection-vis.

3.2 Related Work

In this section, we first examine scalable time series visualization techniques, discussing high-level methods for data aggregation. Specifically, we focus on shape-preserving subsampling algorithms, considering both their practical applicability and prior research in this area.

3.2.1 Time Series Visualization

Visualization plays a crucial role in understanding time series data, especially given the non-intuitive nature of this data type. Particularly, simple visualizations, such as line-based charts, have been found to be suitable for most time series analysis tasks, as opposed to more complex or specific approaches [3].

Interactive visualization techniques are essential for exploring large amounts of time series data. Shneiderman et al. proposed the visual information-seeking mantra, which emphasizes the importance of exploring an overview of the data, followed by zooming and filtering before accessing details-on-demand [19]. Moreover, Walker et al. stress that interactive line chart visualizations improve the efficiency of performing visualization-oriented tasks, such as trend analysis and outlier detection [20].

However, handling large datasets poses significant storage, data transmission, and visualization challenges, necessitating scalable solutions [5, 6, 7]. As a result, over the last decade, there has been a surge in the development of time series databases, such as InfluxDB and TimescaleDB, designed to handle and persist large-scale time series data [21]. Concerning visualization, advances in progressive streaming and rendering techniques focused on improving data transmission and rendering latency [22, 23]. Yet, data aggregation proves the most established technique to efficiently visualize large data by either aggregating or selecting a subset of data points [8, 9, 3].

3.2.2 Data Aggregation for Line Chart Visualization

Data aggregation techniques play a pivotal role in overcoming the challenges posed by big data and visual analytics [8, 24, 25]. In particular, front-end rendering and responsiveness slow down substantially when more data points are transferred and portrayed, making data aggregation an effective approach to facilitate scalability [23]. Their ability to integrate smoothly with existing visualization systems, due to minimal interdependence, has contributed to their widespread adoption [7, 10]. Remark that a common side effect of visualizing large amounts of data is overplotting, where multiple data points overlap due to limited visualization canvas size [26, 20]. Data aggregation can leverage this effect by potentially omitting some of the overplotted data points.

There are two different approaches to performing data aggregation for visualization: downsampling and density-wise aggregation. Downsampling outputs a lower-dimensional representative dataset, while density-wise aggregation employs shared density color coding to render an image. Downsampling can be further divided into subsampling and characteristic data aggregation. Below, we discuss density-wise and characteristic data aggregation, while Section 3.2.3 delves into (shape-preserving) subsampling.

3.2.2.1 Density-Wise Aggregation

Density-wise data aggregation involves using a shared density color coding to render an image of the data [27]. While this technique is effective for visualizing single, large data modalities such as maps and point clouds, it poses challenges when there are multiple distinct modalities in the data. Each modality would then require a distinguishable color density coding, often leading to overplotting. However, time series data typically consists of distinct modalities, making this technique less preferable. Moreover, converting data into images restricts differentiation and interactivity, such as toggling time series traces, further reducing its suitability for time series visualization. *Datashader* is the only known implementation of density-wise data aggregation that works for time series data (i.e., line charts) [28].

3.2.2.2 Characteristic Aggregation

Characteristic data aggregation reduces data by highlighting properties or trends, using methods like mean, median, and smoothing. These approaches are frequently tailored to optimize a specific visual information retrieval task [29], necessitating user studies, as these tasks are hard to quantify objectively [8].

In the context of line chart visualization, Rong et al. introduced *ASAP* (Automatic Smoothing for Attention Prioritization) [30], which smooths time series by adaptively optimizing noise reduction and trend retention, directing users' attention towards significant deviations. Rosen et al. proposed *topolines*, a topological-based line chart smoothing algorithm through subsampling. Topolines aims to preserve high amplitude extrema while flattening low amplitude ones, maintaining minimal residual error [31]. The authors compared their technique to other smoothing algorithms, using data-space metrics that compare the (interpolated) smoothed data against the original data for various smoothing levels.

3.2.3 Shape-Preserving Subsampling

Subsampling—also known as data point selection or value-preserving aggregation—performs data aggregation by selecting data points from the original time series [10]. Shape-preserving subsampling is concerned with preserving the overall shape of the data [15]. To do so, such techniques leverage linear interpolation in the line chart visualization between adjacent selected data points.

Unlike characteristic and density-wise aggregation techniques, the objective of shape-preserving aggregation is to (visually) approximate the ground truth image. The ground truth image, also referred to as the template or reference image, is the visualization that includes all the data points [15, 14]. Thus, the visual representativeness of value-preserving methods can be evaluated using image quality assessment metrics [32]. Note that both aggregated and ground truth images may suffer from overplotting. Yet, these cases are still relevant in the context of interactive visualization systems like zoom-plot [20], where zooming and panning interactions facilitate effective exploration of large time series data [7].

3.2.3.1 Scalability

For real-world, large-scale time series data, downsampling algorithms should meet specific computational requirements. These include: (i) at most $O(N)$ runtime complexity, i.e., a single iteration over the data; (ii) avoidance of additional data structures scaling with data size, as maintaining and updating such structures is computationally infeasible; and (iii) parallelizable to alleviate linear scaling by distributing data partitions across multiple threads or cores.

Table 3.1 summarizes the prominent shape-preserving algorithms for time series

Table 3.1: Overview of shape-preserving subsampling algorithms, where N denotes the time series length and n_{out} represents the aggregation output size. Techniques highlighted in bold are considered for analysis in this work.

	(i) Time	(ii) Memory	(iii) Parallelizable	Data points per bucket
EveryNth	$\mathcal{O}(n_{out})$	$\mathcal{O}(n_{out})$	✓	1
MinMax	$\mathcal{O}(N)$	$\mathcal{O}(n_{out})$	✓	2
M4 [15]	$\mathcal{O}(N)$	$\mathcal{O}(n_{out})$	✓	± 4
LTOB [14]	$\mathcal{O}(N)$	$\mathcal{O}(n_{out})$	✓	1
LTTB [14]	$\mathcal{O}(N)$	$\mathcal{O}(n_{out})$	×	1
RDP / PIP [16, 17, 33]	$\mathcal{O}(N * n_{out})$	$\mathcal{O}(n_{out})$	×	/
VW [18]	$\mathcal{O}\left(N\left(N - n_{out}\right)\right)$	$\mathcal{O}(N)$	×	/
MinStdErrB [14]	$\mathcal{O}(N^2)$	$O(N^2)$	×	1
LLB [14]	$\mathcal{O}(N^2)$	$\mathcal{O}(N^2)$	×	1

visualization, aligning them with the three outlined scalability requirements. Only the four algorithms above the first horizontal line satisfy all proposed requirements. The algorithms listed below this horizontal line fail to meet the parallelization requirements. Since largest triangle one bucket (LTOB) is a shortsighted variant of LTTB (considering only its two neighboring data points) [14] and Gil et al. [34] demonstrated LTTB's superiority over LTOB, we excluded LTOB from this study. Both RDP and VW are not optimal for large-scale time series visualization due to their more expensive computational runtime. Finally, the table lists two other algorithms introduced by Steinarsson [14] that are deemed unsuitable for similar reasons, with both Min-Std-Error-Bucket and longest line bucket (LLB) having an $O(N^2)$ runtime and memory complexity. Remark that most algorithms in Table 3.1 are considered uniform, since they rely on binning, i.e., divide the x-range into buckets for which they select one or multiple data points. Conversely, RDP and VW operate on the full data range without binning, making them non-uniform algorithms.

3.2.3.2 EveryNth

The EveryNth algorithm, also known as uniform subsampling or decimation, is a highly efficient but rather naive method that selects data points at regular intervals; i.e., every n^{th} data point from the input data array, with $n = \frac{N}{n_{out}}$. Notably, this is the only algorithm among those listed in Table 3.1 that scales linearly with n_{out} instead of N.

3.2.3.3 MinMax

MinMax aims to preserve the shape of time series data by selecting vertical extrema for each bucket (i.e., bin). The algorithm selects both the minimum and maximum value within each bin, but does not enforce extrema alternation. This can result in repeated neighboring minima (or maxima), potentially failing to correctly display alternations

occurring in the original time series. MinMax is easily parallelizable over the bins, enabling efficient execution on multicore processors.

3.2.3.4 M4

M4, proposed by Jugel et al. [15], is a subsampling method that can achieve pixel-perfect data aggregation, implying that no pixel differences can be observed. For each bin, M4 selects two horizontal (first, last) and two vertical (min, max) extrema. The algorithm is also easily parallelizable over the buckets. Remark that M4 selects more data points per bin than the other algorithms. As such, Bae et al. introduced inter-pixel gradient-based M4 (IGM4) [35], which removes redundant data points as a post-processing step, reducing the selected data points by an average of 20%.

3.2.3.5 Largest-Triangle-Three-Buckets (LTTB)

Steinarsson proposed several shape-preserving subsampling algorithms [14]. This work focuses on the LTTB algorithm, as the other proposed algorithms either have unfavorable time complexity or have poorer visual representativeness compared to LTTB [34]. LTTB is based on the concept of effective triangular area, which is often employed in line simplification algorithms. For each bucket, it selects the data point that forms the largest triangular surface with the previously selected data point and the next bucket's average value. Although its runtime scales linearly with data size, it has a steeper slope compared to MinMax and M4 due to the higher cost of area calculations. LTTB cannot be parallelized, as it requires a sequential pass over the data (since the previously selected data point is part of the local surface calculation).

3.2.3.6 Ramer-Douglas-Peucker (RDP)

RDP is a non-uniform subsampling technique that optimizes the l^{∞}-norm of the residual error [31, 16]. RDP starts with selecting the inputs' boundary points and connects them with linear interpolation. Points are then iteratively added by inserting the point with the greatest (orthogonal) distance to the interpolated line between already selected points. This continues until reaching either the desired number of output points or a specified distance threshold (ϵ). RDP excels in displaying spikes or large changes due to this interpolation-based distance selection. The iterative selection process is highly related to importance ranking, which is also known as perceptually important points (PIP) [17]. Remark that the iterative selection process, along with the orthogonal distance computation, results in higher time complexity than the above outlined techniques [33].

3.2.3.7 Visvalingam-Whyatt (VW)

Similar to RDP, the VW algorithm is a non-uniform line simplification method [18]. However, VW performs an elimination instead of a selection, iteratively removing the data point with the smallest effective area to its two adjacent (non-eliminated) points. This iterative process of recalculating and removing points results in the high time and memory complexity of VW. VW's effective area computation results in a more uniform data sampling than RDP, as the latter is solely concerned with orthogonal distances to a line segment, apparent in Fig. 3.1. As a result, VW may eliminate spikes in favor of covering larger horizontal spaces (rather than vertical ones), using the effective area.

3.2.4 Evaluating Time Series Data Aggregation

Several works have made steps toward assessing the visual representativeness of data aggregation for time series visualization.

Shi et al. [36] proposed two data-based metrics for evaluating cartographic line simplification: displacement measure (the maximal perpendicular distance between the simplified and original line) and shape distortion measure (the average difference in inclination angles between the simplified and original line).

Building on their prior research, Rosen et al. extended the topolines validation framework [31] (see section 3.2.2.2) to assess the effectiveness of multiple line chart smoothing algorithms [37]. This was performed by linking data-space metrics, such as largest extrema variation, to specific user visualization tasks, like extrema value retrieval. The effectiveness of these smoothing algorithms was then determined by examining how these metrics performed across different time series signals at various smoothing levels. The visual complexities of these smoothing levels were assessed using the approximate entropy (ApEn) [38] measure on the interpolated smoothed data. While this approach yields valuable insights into the effectiveness of line chart smoothing algorithms for specific user tasks at different smoothing levels, it does not directly correlate with visual representativeness and data efficiency of shape-preserving algorithms. Specifically, relying on ApEn for visual complexity assessment could potentially bias the results [2].

Jugel et al. [15] used the structured similarity index measure (SSIM) as an image-based metric to evaluate and compare their M4 data aggregation technique. However, the authors investigated data efficiency using the number of bins rather than the number of output data points as the independent variable, overlooking the variability in the number of selected data points per bin.

Gil et al. [34] contributed a method for adaptively identifying optimal data point selection algorithms for time series compression, using multiple value-preserving algorithms and a data-based error score. However, the approach did not include MinMax

[2] This README demonstrates that there are large differences in n_{out} for similar ApEn values of RDP and EveryNth, demonstrating that RDP has on average 3-10 fewer data points than EveryNth (largely attributable to the non-uniformity of RDP sample selection, contributing to higher entropy values)

and only evaluated mean absolute error (MAE). Furthermore, the scalability of the proposed approach is limited as it requires interpolating the downsampled signal to the original signal, resulting in $O(N)$ memory complexity. Additionally, their results suggest that using LTTB or M4 is only marginally worse than the optimal strategy, questioning the added value of their approach.

Despite numerous studies examining various data aggregation algorithms, there remains a lack of thorough analysis concerning the visual representativeness of shape-preserving algorithms and their data efficiency. Moreover, to the best of our knowledge, no prior research has investigated the stability of these subsampling methods in a line chart context when updating (e.g., streaming) or interacting with the visualization (panning and zooming).

3.3 Methodology

The purpose of this work is to evaluate and compare shape-preserving subsampling algorithms for scalable line chart visualization. Specifically, we focus on two aspects:

1. **Visual representativeness** refers to the degree to which a line chart visualization of subsampled data visually reflects the original rendered data. To assess this, we use two image-space metrics that are each aggregated, i.e., scaled, using an OR-convolution mask of the joined image.

2. **Visual stability** aims to assess the visual changes that occur when zooming, panning, or streaming, i.e., when re-rendering subsampled data with a large overlapping region. To assess visual stability, we use a data-space metric that focuses on residual errors at peaks.

3.3.1 Study Design

A key aspect of our study design involves utilizing a fixed canvas size of 800 by 250 pixels for all visualizations, in accordance with Jugel et al. [15]. This allows for reliable image-space comparisons when varying n_{out}, facilitating a fair analysis regarding data efficiency. Furthermore, since our study focuses solely on univariate time series visualization, resulting in a single line (per) chart, there is only one color dimension (grayscale) needed. Hence, the images of the resulting visualization are 2D matrices, with each pixel coordinate containing a single illumination value ($v \in [0, 255]$).

3.3.1.1 Time Series Templates

Multiple time series templates are utilized to cover a wide range of time series properties, including periodicity, (non-)stationarity, noisiness, spikiness, and spanning surface of time series on the visual canvas. These templates, also referred to as image templates, represent signals of a specific size N. The extent of N-variation in these templates provides a comprehensive representation of various zoom levels, which is a prominent way

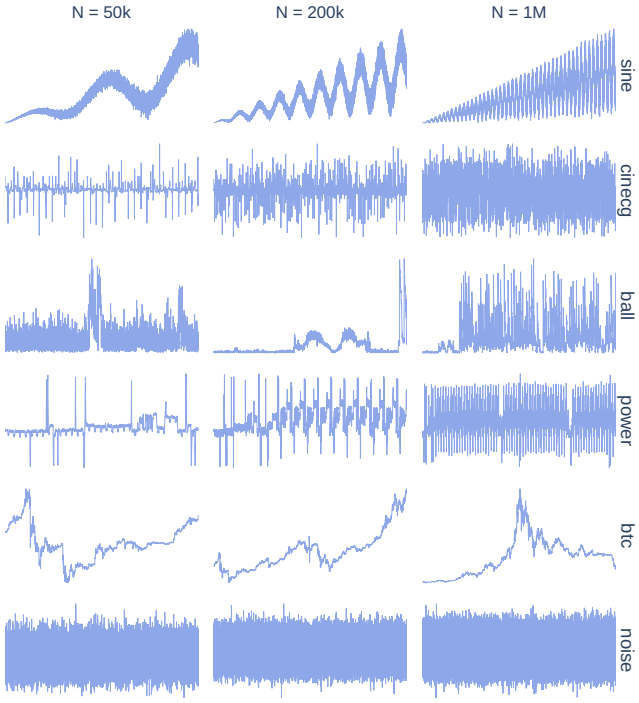


Figure 3.2: Grid of utilized time series templates. Each row represents a distinct time series, and the columns indicate the template sizes.

of interacting with time series data [3]. We selected the sizes $N \in \{50K, 200K, 1M\}$ through empirical analysis. These sizes were deemed practical for visualization purposes (i.e., capable of rendering the reference image without crashing).

Our study utilized six distinct time series signals that can be observed in Fig. 3.2. These include: (1) a noisy increasing sine, representing a periodic, non-stationary signal, with noise that scales with signal amplitude; (2) an ECG signal with prominent peaks and periodicity from the UCR time series archive [39], used by Franco et al. [40]; (3) a positively peaked ball speed signal from the DEBS 2013 dataset [41], used by Jugel et al. [15, 10]; (4) a power consumption signal from the DEBS 2012 dataset [42] exhibiting periodicity and prominent peaks, used by Jugel et al. [15, 10]; (5) a non-stationary signal constructed of 1-minute intervals of historical bitcoin data [43], in accordance with the stock data used by Steinarsson [14]; and (6), generated Gaussian noise, representing stationary noise, spanning a large surface of the canvas. Fig. 3.2 shows that as the template size N increases for both the ECG and power signals, more peaks and periods become visible, increasing the relative frequency. Remark that the ball time series signal displays completely different patterns as N changes.

3.3.1.2 Data Efficiency

When assessing the effectiveness of downsampling algorithms, it is important to compare them based on the number of output data points n_{out}. This number corresponds to the amount of data that is transmitted to and rendered in the front-end, which directly affects visualization performance. Using n_{out}, instead of number of bins, overcomes the variability within uniform (i.e., bin-based) subsampling and accommodates non-uniform subsampling approaches that do not have the concept of buckets, see Table 3.1. Subsampling algorithms that yield (relatively) high performance at rather low n_{out} values are considered data efficient.

3.3.2 Visual Representativeness

Visual representativeness measures how well a subsampled time series graph visually approximates the original data's visualization. This involves comparing the aggregated time series visualization to the reference visualization, which corresponds to comparing images. We utilize two well-established image-space metrics that capture various aspects of this comparison: peak signal to noise ratio (PSNR) and structured similarity index measure (SSIM). These metrics, considered full-reference image quality assessment measures within the spatial domain [32], aim to quantify both absolute errors via PSNR and perceived quality with SSIM. While PSNR is the most commonly used measure for image quality assessment, it has been criticized for overlooking certain image signal characteristics and poor correlation with human perception [44]. Therefore, we also use SSIM, which is frequently utilized to complement the PSNR [15, 45]. Since these two metrics operate in image space, we can investigate the impact of the various parameters that affect the visualization. These include, but are not limited to, the rasterization backend, plotting toolkit, line-drawing style, line width, and anti-aliasing. Remark that the reference and aggregated visualization, on which the metrics are computed, should be constructed using the same toolkit configuration. We are particularly interested in evaluating the visual representativity of visualization toolkits' default configurations, i.e., using line charts in a Cartesian coordinate system with linear interpolation between data points.

Notably, data-space metrics, instead of image-space metrics, cannot adequately capture the "visual" representativeness of subsampled data. In the context of shape-preserving subsampling techniques for scalable visualization, large data reductions are commonly performed, removing several *data* properties, e.g., frequency information. Data-space metrics are inclined to impose penalties for the exclusion of these characteristics. Yet, despite the loss of certain data properties through subsampling, the resulting visualization may still closely approximate the original data's visualization. For instance, the subsampled data might not contain (high-)frequency components or lower narrow peaks present in the original data, which could be obscured in the visualization due to phenomena such as overplotting.

3.3.2.1 Peak Signal-to-Noise Ratio (PSNR)

PSNR transforms the mean squared error (MSE) in a logarithmic scale, improving the interpretability of the reconstruction error. For each pixel, the squared error is calculated, which is then aggregated into an MSE by averaging over all pixels within the OR-conv mask. MSE measures absolute pixel differences but, unlike MAE, it imposes smaller penalties on minor shading variations caused by anti-aliasing.

$$PSNR = 10 \log_{10} \left(MAX^2 / MSE \right); \quad MAX = 255$$

3.3.2.2 Structural Similarity (SSIM)

SSIM is a commonly used metric for assessing perceived quality [46], producing a similarity score ranging from -1 to 1. A score of 1 indicates perfect similarity, 0 indicates no similarity, and -1 indicates perfect anti-correlation. Just as for the PSNR, SSIM outputs a matrix that contains a value for each pixel. These per-pixel SSIM values are then mean-aggregated within the OR-conv mask, see Section 3.3.2.3.

3.3.2.3 OR-conv Masking

We introduce a technique called "OR-conv masking" to aggregate per-pixel metrics in a more targeted manner. Typically, full-reference image quality assessment measures are aggregated over the entire image using a global mean, as performed by Jugel et al. [15, 10]. However, line charts often occupy only a small part of the canvas, and aggregating them using a global mean would result in smoothened-out metrics influenced by the canvas's filling level. To address this issue and facilitate cross-template comparisons, we introduce OR-conv masking. This technique aggregates only the per-pixel values (in the neighborhood) of rendered data points in the visualization, which is (to some extent) in line with the weighted aggregation proposed by Li et al. [47]. For further details, we refer to a description containing examples and implementation details in the GitHub repository.

3.3.3 Visual Stability

Visual stability evaluates visual changes that stem from updating the current view of the data by subsampling a limited new data range, reflecting user interactions such as panning and zooming. The stability of subsampling algorithms is a crucial aspect to consider, given that interactivity is essential for effective exploration of time series data [7, 19].

Visual stability assessment consists of two primary cases: panning and zooming. Panning involves shifting the displayed data range by an offset $\Delta \neq 0$, resulting in a range of $[x_{start} + \Delta, x_{end} + \Delta]$. Zooming changes the data range to $[x_{start} + \Delta_l, x_{end} + \Delta_r]$ with at least one $\Delta \neq 0$ and $\Delta_l \neq \Delta_r$ (as $\Delta_l = \Delta_r$

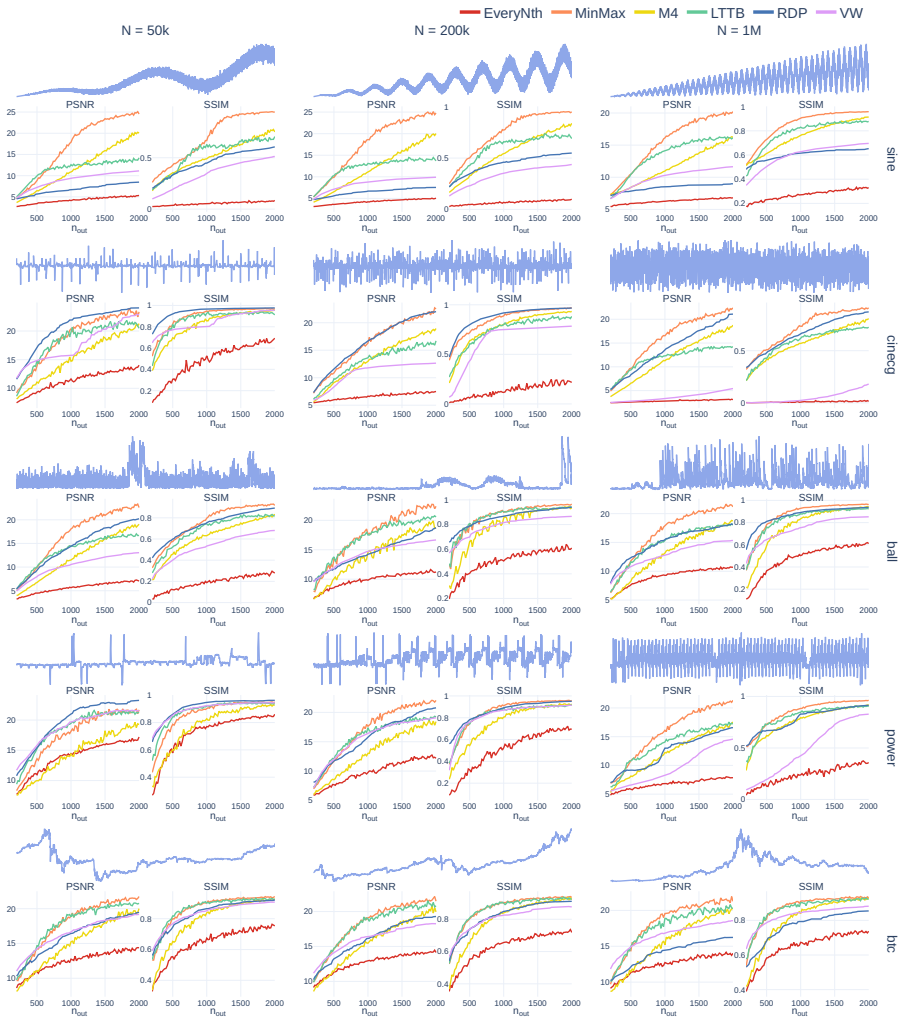


Figure 3.3: Assessing visual representativeness of shape-preserving subsampling algorithms for various image templates. Each row displays a distinct time series dataset, with columns indicating the template size. All image templates were generated using Plotly's default settings (linear interpolation, 2-pixel line width). The metric subplots reveal performance trends of the aggregators as n_{out} (x-axis) increases (range [200, 2000]). Each metric is scaled via an OR-conv mask, with PSNR indicating the approximation quality and SSIM assessing the structural similarity. This gif demonstrates the consistency in the metric-based rankings of the algorithms across the default configurations of the other toolkits.

is panning). Remark that streaming corresponds to zooming out with $\Delta_l = 0$ and $\Delta_r > 0$. Additionally, it is worth noting that visual stability is complementary to visual representativeness, as the latter addresses large zooms (due to varying template

sizes N), while the former concentrates on smaller updates.

Visual stability is primarily influenced by the chosen downsampling algorithm rather than the visualization toolkit's specific configuration. Although an image-space approach could assess stability, it faces challenges such as the need for perfectly overlapping pixels and the inability to change the y-range when new global extrema are added during an update. To avoid these limitations and effectively quantify stability, we utilize a data-space metric. Our approach involves first interpolating the updated subsampled data to the initial subsampled data at their intersection, enabling data-space comparisons at the x-values of the initial data. We then calculate our stability metric: the mean absolute error at the peaks (MAEP). Note that this measure assesses the stability of the downsampling algorithm and thus does not consider the visual representativeness of the downsampled time series, which is measured by the visual representativeness metrics.

3.3.3.1 Mean Absolute Error at Peaks (MAEP)

The MAEP quantifies the average difference at the peaks of the initial subsampled data and the interpolation of the updated data. Peaks are defined as the local maxima or minima in the initial aggregated data that have a relative prominence of $\geq 10\%$, making these regions more visually prominent than other parts of the time series [48]. This metric is partly in accordance with the max perpendicular distance proposed by Shi et al. [36], which aims to capture the largest deviation, and the peak variation 1-Wasserstein distance of Rosen et al. [37]. This makes MAEP a sensitive measure of visual stability, as it focuses on the most visually prominent parts of the data.

3.4 Results

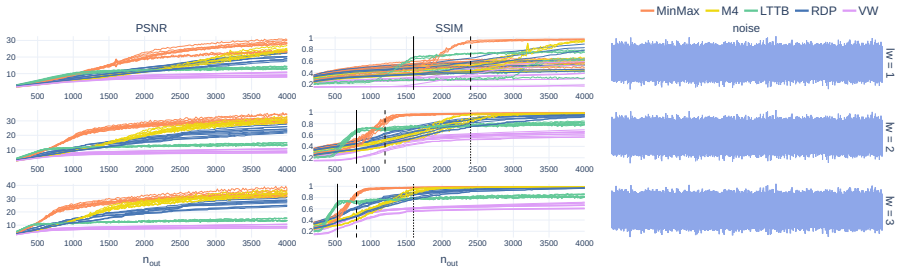


Figure 3.4: Visual representativeness of noise for varying line widths (1, 2, 3), data sizes (50k, 200k, 1M), and toolkits (plotly, bokeh, matplotlib), with n_{out} 200-4000. The aggregation images are compared to reference images of the same line width. The EveryNth algorithm was omitted from this visualization to reduce visual clutter.

3.4.1 Visual Representativeness

We assess visual representativeness for three widely used Python visualization toolkits: Matplotlib, Plotly, and Bokeh. These toolkits were chosen as they are the most prominent visualization tools in Python, which is the leading language for interoperating with large data [49].

The visual representativeness of the subsampling algorithms is examined through two experiments. First, we maintain a constant line-chart configuration while altering image templates, focusing on the toolkits' default settings. This experimental design mirrors the typical user behavior of not adjusting these settings. Through this analysis, we can identify trends in downsampling algorithms concerning data efficiency across various templates, without the interference of varying configurations. Second, we utilize a stationary noise signal as a fixed time series template to investigate the effects of different visualization libraries and line widths on the visual representativeness of shape-preserving subsampling algorithms. Finally, we examine the specific requirements for achieving pixel-perfect M4 in existing toolkits.

3.4.1.1 Data Efficiency & General Trends

Fig. 3.3 presents the visual representativeness metrics for Plotly's default configuration (line width of 2 pixels, linear interpolation, Cartesian coordinates) for the six shape-preserving subsampling techniques. By visualizing the metrics over n_{out} as the independent variable, we can visually compare data efficiency. A similar visualization using the default configuration of the two other toolkits can be found in the code repository [3].

As a first, general observation, we notice that the trend and ranking of the subsampling algorithms remain largely consistent across the two metrics for each template subplot. Next, corresponding to intuition, we note that increasing n_{out} improves visual representativeness. Regarding the ranking of the subsampling algorithms, we observe that MinMax, LTTB, and RDP generally perform the best across the whole n_{out} range, followed by M4, then VW, and EveryNth performing the worst. These observations also hold true for the default configuration of the other two toolkits, see footnote [3].

In terms of data efficiency, MinMax and LTTB stand out as the most data-efficient aggregators. This is consistent with the findings reported by Gil et al. [34], whose results identified LTTB as a data-efficient algorithm. As n_{out} increases, we notice that MinMax converges to higher scores than LTTB. MinMax excels with rough templates such as all sine templates, cinecg-1M, ball-50k, ball-1M, and power-1M. For low-roughness templates, such as power-50k and all btc templates, LTTB and MinMax demonstrate similar performance. Although LTTB and MinMax have comparable PSNR scores for low n_{out}, e.g., all sine templates, we observe that the SSIM metric favors MinMax over LTTB. This discrepancy between both metrics is only apparent

[3] github:ts-datapoint-selection-vis/details/visual-representativity

in rough templates and stems from SSIM more strongly penalizing the envelopes of these rough curves, where LTTB misses some local extrema due to its bin-wise extrema alternation.

RDP ranks overall third, outperforming other techniques for certain templates. This is particularly the case for sparse spiked data such as cinecg-50k, cinecg-200k, and power-50k. In contrast, RDP yields the second-worst PSNR performance for all sine templates, attributable to its non-uniform selection of data points. This non-uniform sampling is especially apparent in signals where the amplitude of the noise varies over time, see Fig. 3.1. We further observe that RDP has a higher ranking for the SSIM metric compared to the PSNR metric for these sine templates. This increase, related to the LTTB SSIM drop behavior, is attributable to SSIM's higher sensitivity to envelope errors. Such errors are largely mitigated by RDP due to its behavior of selecting prominent local extrema, as demonstrated in footnote [3].

For most templates, VW performs slightly worse than the above techniques for low n_{out}. However, VW appears to perform poorly for high roughness templates such as sine-50k, cinecg-200k, cinecg-1M, ball-50k, and power-1M. This is caused by VW's tendency to discard narrow peaks due to their low effective area when using adjacent points, which is the criterion during the iterative elimination. Consequently, this phenomenon leads to a drop in VW's SSIM ranking compared to its PSNR ranking for the sine templates, which just like RDP's increase is attributable to SSIM's sensitivity to capturing the outer envelope.

M4 and EveryNth exhibit poorer data efficiency and converge considerably slower than MinMax and LTTB, which aligns with expectations, given EveryNth's naive selection and M4 outputting 4 selected data points per bin, reducing its data efficiency.

3.4.1.2 Data Efficiency & Line Width

Fig. 3.4 presents the visual representativeness metrics for the stationary noise time series template, with each row corresponding to a distinct line width. These visualizations include the default configuration for all three toolkits (except for the line width), and the data size N is varied within the subplot, as for stationary noise, N has minimal impact on the image template characteristics.

A general observation is the consistent trend of the aggregator performance curves across various data sizes and toolkits for each of the metric subplots. Remark that some larger inconsistencies are visible in the SSIM subplot for the line width of 1 pixel, caused by rasterization differences of the toolkits, for which we provide a detailed explanation on the GitHub repository[3].

LTTB emerges as the most data-efficient algorithm (for low n_{out}), which can be attributed to its maximization of triangular surfaces. This algorithm is particularly effective at filling large areas due to its preference for alternating between bin-wise minimum and maximum values. In the teaser Fig. 3.1, this alternation between extrema is clearly visible for the LTTB aggregation of noisy data. Nonetheless, MinMax, and

thus not LTTB, converges to the highest value, which can also be attributed to LTTB's bin-wise extrema alternation, leading to the exclusion of global extrema that are selected by MinMax, M4, and RDP. M4 and RDP also converge, albeit slower, to the same values as MinMax, as can be observed in the $lw = 3$ row. VW proves the least data-efficient technique, even converging to the lowest value. This is attributable to VW its effective area elimination process, only considering adjacent points.

For each of the uniform (i.e., bin-based) aggregators, we notice an elbow in the performance curves, i.e., where the curve converges, for the SSIM metric. We notice that the n_{out} position of this elbow decreases when increasing the line width. This indicates that increasing line width can be a means to improve data efficiency (at the cost of granularity). Upon further investigation, we derived a general formula to identify the specific n_{out} position of the observed elbow as a function of canvas width (cw) and line width (lw):

$$elbow = \frac{cw * \bar{n}_{ga}}{lw}$$

Here, $\bar{n}_{ga}$ represents the number of selected data points necessary to have a *guaranteed alternation*. Since LTTB favors extrema alternation, $\bar{n}_{ga}$ is 2. In contrast, MinMax can require up to 3 data points to achieve alternation, resulting in an $\bar{n}_{ga}$ of 3. M4 can be considered as a variation of MinMax that also selects the first and last data points in a bucket, and therefore requires twice the $\bar{n}_{ga}$ of MinMax, i.e., 6. Fig. 3.4 depicts the elbow positions for $\bar{n}_{ga}$ values of 2 (LTTB), 3 (MinMax), and 6 (M4) using solid, dashed, and dotted vertical lines, respectively. The trend for higher data efficiency with increasing line width is also apparent for the image template grid [4].

3.4.1.3 Pixel-perfect M4

The previous subsections focused on data efficiency for visual representativeness, in which M4 did not excel. Instead, its capability to achieve pixel-perfectness makes it stand out. As noted by Jugel et al., achieving pixel-perfect M4 requires a n_{out} that is four times the pixel-width of the canvas [15]. This subsection explores the practicalities necessary to create a pixel-perfect aggregation with the M4 algorithm.

Our first focus is on anti-aliasing, also known as pixel binarization, which is a common technique employed to enhance the perceptual quality of visualizations [50]. In all three toolkits investigated in this study, anti-aliasing is enabled by default. Moreover, Bokeh and Plotly do not offer the option to render aliased figures (disabling anti-aliasing). A straightforward and effective approach to anti-aliasing, suitable for simple graphics like line charts, is to determine the percentage of pixel occupancy and use that percentage as color shading. Fig. 3.5 uses this approach to illustrate M4's fragility to produce pixel-perfect visualizations when anti-aliasing is employed.

Another aspect of the visualization configuration that influences M4 its pixel-perfectness is line width. Increasing the line width results in larger shaded areas, potentially

[4] This GIF illustrates the effect of line width on the image template grid.

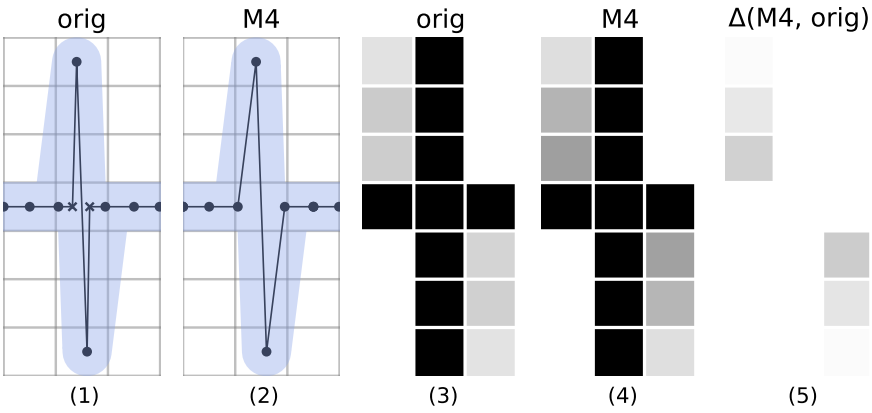


Figure 3.5: Visualization of M4-related pixel errors when anti-aliasing is applied. Subplot (1) presents the original time series featuring a 1-pixel-wide shading. Upon applying M4 in subplot (2), the two data points marked by an $\times$-symbol in (1), are excluded. Subplots (3) and (4) depict the anti-aliased renderings of the original and M4 aggregated time series, respectively, while subplot (5) emphasizes the difference between both.

amplifying pixel shading errors. Bokeh's default line width is set to 1 pixel, Plotly's default is 2 pixels, and Matplotlib defines line width in points, corresponding to $1/72$ of the figure's DPI. Consequently, achieving pixel-perfect M4 without manually adjusting line widths in these toolkits proves to be challenging. Aside from configuring aliasing and line width, we observed that achieving pixel-perfect M4 requires modifying other visualization-related parameters, such as rasterization back-end and x-range binning.

Finally, to establish a bin to pixel-column mapping ($n_{out} = 4 * cw$), one must either control or be informed of the canvas width in advance, which is not always trivial. Graph layout elements, including y-axis ticks, legends, and font sizes, influence the eventual canvas width. Additionally, these elements can change due to updates, such as y-axis ticks changing following a user graph interaction, further complicating canvas width assessment. Moreover, web-based visualization frameworks, such as Plotly and Bokeh can stretch across the entire webpage and are susceptible to browser page resizing and zooming.

In conclusion, achieving pixel-perfect visualization with M4 is difficult or even impossible in most existing visualization toolkits, given the necessity for substantial control over the visualization environment. Jugel et al. mentioned that "in a real implementation, the engineers have to make sure that $n_{out} = 4 * cw$ to achieve the best results" [15], but this statement was only a minor point in their paper. Bae et al. presented a practical implementation of M4 in D3, yet they neither mentioned anti-aliasing nor verified the pixel-perfectness of their implementation [35]. Therefore, we argue that M4 should not be considered merely as another subsampling algorithm, but rather as a (data-efficient) visualization technique, given its tight coupling to the rendering process of the data points.

3.4.2 Visual Stability

As detailed in the methodology section, visual stability is categorized into two primary cases, i.e., panning and zooming, which both can be formally defined using offsets. These offsets, defined as ratios, are calculated based on N and include the following values: $[1/53, 1/27, 1/17, 1/11, 3/23, 2/11]$. Note that these selected offset ratios all have a prime number in the denominator, corresponding more to real-world user interactions instead of aligned offsets. For instance, selecting 0.1 as the offset ratio

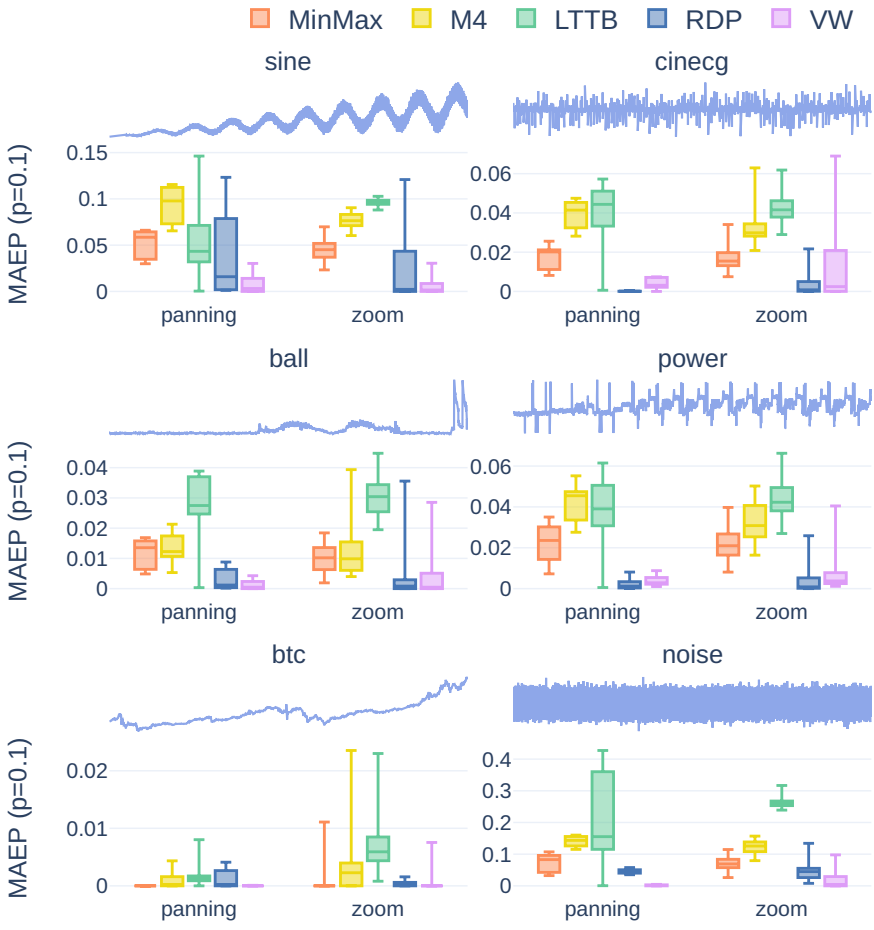


Figure 3.6: Evaluation of visual stability for shape-preserving subsampling algorithms applied to different time series templates with length 200k. Each subplot represents a unique time series template displaying the MAEP (peak prominence $\geq 10\%$) distributions for the examined subsampling algorithms, distinguished by color. In accordance with the line width analysis in Fig. 3.4, we omitted the (naive) EveryNth aggregator to reduce visual clutter. A GIF showing visual stability in practice can be found online for both the zooming and panning actions.

would result in perfectly aligned bins over the intersection, which will (likely) yield a MAEP score of 0.

Fig. 3.6 shows visual stability outcomes for the six datasets proposed in our methodology, utilizing a data size of 200k. We opted for the 200K templates to avoid excessively low or high roughness found in the 50K and 1M templates, see Fig. 3.2. Additionally, including the noise data covers high roughness cases.

We observe a remarkable consistency in the algorithms' MAEP rankings across (i) the panning and zooming interactions for the same time series and (ii) across the different time series templates. VW and RDP exhibit the greatest overall visual stability, characterized by low MAEP values. MinMax has the third-lowest overall MAEP, followed by M4, while LTTB demonstrates the poorest visual stability.

The high visual stability of RDP and VW can be attributed to their non-uniform approach, i.e., these two algorithms do not perform binning. Instead of binning, both algorithms work on the full range of data points, making the selection or reduction process less prone to the precise start and end points of the template. Furthermore, remark how RDP demonstrates a large MAEP spread for the sine template, which is also the template for which RDP has the worst visual representativeness, see Fig. 3.3. This can be attributed to the high sensitivity of RDP's selection procedure to noise with varying amplitude.

VW yields the highest visual stability for the majority of templates. However, for the cinecg template that is characterized by its spikiness, VW exhibits a larger spread for zooming. Notably, the visual representativity outcomes for VW indicate lower performance for such highly spiked templates (see Fig. 3.3, cinecg-200k cinecg-1M).

MinMax ranks overall third, making it the most stable algorithm among those that employ binning. A plausible explanation for MinMax's stability, compared to the other algorithms that use binning, lies in its high *locality*. Locality, in uniform algorithms, refers to the extent that subsampling depends on individual bin values without influence from neighboring bins. High locality results in greater stability and reduces susceptibility to alignment issues. In contrast, non-uniform algorithms, which operate iteratively on the full data rather than relying on binning, are also considered to have high locality.

M4, a combination of MinMax and EveryNth, results in (the EveryNth) half of the chosen data points being susceptible to alignment. In M4, the minimum and maximum values are always between the first and last (bin) points, causing two out of three visually interpolated lines per bin to be sensitive to the offset (only the interpolation between the minimum and maximum values remains largely unaffected). Consequently, we observe that M4 performs worse than MinMax.

Finally, LTTB exhibits the poorest visual stability, which can be attributed to its lower locality. In particular, LLTB's tendency to favor bin-wise extrema alternation amplifies the propagation of alignment issues throughout the data point selection process. In particular, for noise data, the instability of LTTB is most prominent, as

there are many local alternating peaks to choose from.

3.5 Guidelines and Discussion

Based on the evaluation results, we formulate the following evidence-based guidelines for time series visualization at scale with shape-preserving subsampling:

1. **Use MinMax, RDP, or LTTB for data-efficient aggregation**: MinMax and LTTB proved the most data-efficient for low n_{out}. However, for larger n_{out}, MinMax often outperforms RDP's and LTTB's visual representativeness. RDP particularly excels in representing low noisy time series with sparse spikiness. LTTB proves to be most efficient in aggregating noisy data.

2. **Choose RDP or MinMax for superior visual stability**: RDP stands out for its high visual stability, substantially surpassing uniform algorithms. MinMax emerges as the most visually stable uniform algorithm, attributed to its high locality. Although VW generally exhibits the highest visual stability, its lower visual representativeness makes it fall short for recommendation.

3. **Consider pixel-perfect M4 as a visualization technique rather than an aggregation algorithm**: Achieving pixel-perfect M4 requires aliased figures and control over various visualization-related parameters, such as line width, rasterization back-end, x-range binning, and canvas width.

4. **Consider line width when aiming for high data efficiency**: Increasing line width has proven to be an effective means for improving data efficiency at the cost of granularity. We identified a relationship between line width, canvas width, and n_{out}, which defines the elbow of the performance curve for uniform subsampling algorithms.

While MinMax and RDP seem to have received only little attention, for instance, the studies conducted by Gil et al.[34], Bae et al. [35], and Steinarsson [14] did not consider both algorithms, our results nevertheless indicate that MinMax and RDP are highly competitive downsampling algorithms for line chart visualization. In accordance with our results, only Jugel et al. [15], acknowledged the high data efficiency of MinMax and RDP as a (minor) observation in their analysis. While RDP demonstrates high visual representativeness and stability, its non-uniformity may result in visualization artifacts, e.g., long interpolated segments in parts of the line chart with minor (vertical) fluctuations, which can have a noticeable perceptual impact [14].

RDP and VW are two non-uniform subsampling algorithms. Yet, our evaluation shows that their underlying selection/elimination procedure causes a large difference in visual representativeness. Particularly, VW utilizes the effective area of adjacent points, resulting in more uniform sampling, at the cost of possibly omitting local (narrow)

peaks. In contrast, RDP relies on (projected) distances to interpolated lines, making it good at selecting local peaks, at the cost of highly non-uniform sampling, resulting in less visually pleasing outcomes for low n_{out}.

LTTB excels in visual representativeness and data efficiency for noisy data, as shown in Fig. 3.4. Considering that large (stationary) datasets, displayed on constrained canvas sizes, tend to resemble noisy signals, LTTB arguably emerges as the most suitable algorithm. Yet, its major drawback lies in its low locality and bin-wise extrema alternation, compromising visual stability. As such, future studies should explore methods to improve LTTB's limitations.

3.6 Conclusion

This work presents an extensive metrics-based methodology for evaluating shape-preserving subsampling algorithms for line chart visualization, quantifying visual representativeness and visual stability. Our methodology employs various time series datasets, each of three different sizes, accounting for a wide range of time series properties and zoom levels. To measure visual representativeness, we use two well-established image-space metrics, allowing us to investigate the influence of visualization toolkit parameters. Furthermore, we advocate using the number of selected data points as a more fair measure of data efficiency. To measure visual stability, we propose using a data-space metric, overcoming alignment limitations that would present themselves when evaluating in image space.

We evaluated the EveryNth, MinMax, M4, LTTB, RDP, and VW algorithms using our proposed methodology, taking various figure-drawing properties into account. Visual representativeness was evaluated for these six subsampling algorithms for three different toolkits using their default configuration. We observed that the ranking of the algorithms remains consistent across these toolkits. In particular, MinMax, LTTB, and RDP emerged as superior algorithms in terms of visual representativeness. Using our methodology, we are capable of uncovering the intricacies between the two image-space metrics for LTTB and MinMax, and RDP and VW. We also explored the impact of line width on data efficiency and derived a general formula for uniform subsampling algorithms that defines the elbow of the performance curve. Furthermore, we showed that realizing pixel-perfect M4 is nontrivial in practice and even impossible in the presence of anti-aliasing. As for visual stability, VW, RDP, and MinMax were identified as the most stable algorithms, often by a significant margin. We provided an explanation for these results by relating to the locality of downsampling algorithms.

Based on these insights, we derived a set of guidelines for scalable line chart visualization with subsampling. A general conclusion is that both the MinMax and RDP algorithms deserve more attention, given their superior performance while not being considered in most related studies (in favor of LTTB) [34].

The proposed methodology and obtained results are publicly available, enabling

reproducibility and further research in this domain. This study provides a foundation for the design and evaluation of novel advancements in shape-preserving subsampling.

3.6.1 Limitations

Although this study provides numerous insights on subsampling for visualization, some limitations should be acknowledged. Firstly, we scoped this study, in line with other related studies [14, 30, 15, 34], to (nearly) regularly sampled univariate data without any large gaps, which may not be fully representative of real-world time series data.

To assess the visual representativeness and data efficiency across the algorithms, two established image-space metrics were employed. Yet, the effectiveness of how these metrics relate to human perception (of line-charts) is not certain [46, 51]. To partly mitigate this limitation, we also analyzed other metrics in our online materials[3], demonstrating highly similar rankings and trends, increasing our confidence regarding the validity of the two proposed metrics.

Additionally, no user studies were employed, and while metrics can capture visual representativeness and visual stability to some degree, they are proxies for user perception. Therefore, future work should consider employing user studies to (i) validate our findings regarding visual representativeness and visual stability and (ii) possibly identify thresholds for the utilized metrics below which the perception does not improve. Our openly available code base provides a sound foundation for such studies, as two of the investigated toolkits are web-integrable, enabling web-based surveys.

Finally, this study only investigated a limited set of visualization configurations that influence visual representativeness, such as line width and toolkit, with a specific focus on the default parameters of the considered visualization toolkits. Line shape and other configurations, including canvas aspect ratio and axis scale (e.g., logarithmic), were not investigated, as these are less frequently changed. However, our proposed methodology allows for investigating these in future studies.

References

[1] Tak-chung Fu. "A review on time series data mining". In: *Engineering Applications of Artificial Intelligence* 24.1 (2011), pp. 164–181.

[2] Jessica Lin, Eamonn Keogh, and Stefano Lonardi. "Visualizing and Discovering Non-Trivial Patterns in Large Time Series Databases". en. In: *Information Visualization* 4.2 (June 2005), pp. 61–82. ISSN: 1473-8716, 1473-8724. DOI: 10.1057/palgrave.ivs.9500089. URL: http://journals.sagepub.com/doi/10.1057/palgrave.ivs.9500089 (visited on 03/11/2022).

[3] Wolfgang Aigner, Silvia Miksch, Wolfgang Müller, Heidrun Schumann, and Christian Tominski. "Visualizing time-oriented data—A systematic view". en. In: *Computers & Graphics* 31.3 (June 2007), pp. 401–409. ISSN: 00978493. DOI: 10.1016/j.cag.2007.01.030. URL: https://linkinghub.elsevier.com/retrieve/pii/S0097849307000611 (visited on 03/03/2022).

[4] G. Shurkhovetskyy, N. Andrienko, G. Andrienko, and G. Fuchs. "Data Abstraction for Visualizing Large Time Series". In: *Computer Graphics Forum* 37.1 (Feb. 2018), pp. 125–144. ISSN: 0167-7055, 1467-8659. DOI: 10.1111/cgf.13237. URL: https://onlinelibrary.wiley.com/doi/10.1111/cgf.13237 (visited on 03/15/2024).

[5] Jean-Daniel Fekete. "Visual analytics infrastructures: From data management to exploration". In: *Computer* 46.7 (2013), pp. 22–29.

[6] Fan Du, Ben Shneiderman, Catherine Plaisant, Sana Malik, and Adam Perer. "Coping with volume and variety in temporal event sequences: Strategies for sharpening analytic focus". In: *IEEE transactions on visualization and computer graphics* 23.6 (2016), pp. 1636–1649.

[7] Jonas Van Der Donckt, Jeroen Van Der Donckt, Emiel Deprost, and Sofie Van Hoecke. "Plotly-resampler: Effective visual analytics for large time series". In: *2022 IEEE Visualization and Visual Analytics (VIS)*. IEEE. 2022, pp. 21–25. DOI: 10.1109/VIS54862.2022.00013.

[8] Bum Chul Kwon, Janu Verma, Peter J. Haas, and Cagatay Demiralp. "Sampling for Scalable Visual Analytics". en. In: *IEEE Computer Graphics and Applications* 37.1 (Jan. 2017), pp. 100–108. ISSN: 0272-1716. DOI: 10.1109/MCG.2017.6. URL: http://ieeexplore.ieee.org/document/7819391/ (visited on 03/05/2023).

[9] Rajeev Agrawal, Anirudh Kadadi, Xiangfeng Dai, and Frederic Andres. "Challenges and opportunities with big data visualization". en. In: *Proceedings of the 7th International Conference on Management of computational and collective intElligence in Digital EcoSystems*. Caraguatatuba Brazil: ACM, Oct. 2015, pp. 169–173. ISBN: 978-1-4503-3480-8. DOI: 10.1145/2857218.2857256. URL: https://dl.acm.org/doi/10.1145/2857218.2857256 (visited on 03/27/2022).

[10] Uwe Jugel, Zbigniew Jerzak, Gregor Hackenbroich, and Volker Markl. "VDDA: automatic visualization-driven data aggregation in relational databases". en. In: *The VLDB Journal* 25 (Feb. 2016), pp. 53–77. ISSN: 1066-8888, 0949-877X. DOI: 10.1007/s00778-015-0396-z. URL: http://link.springer.com/10.1007/s00778-015-0396-z (visited on 03/03/2023).

[11] Jean-Luc R Stevens, Philipp Rudiger, and James A Bednar. "HoloViews: Building Complex Visualizations Easily for Reproducible Science". en. In: (2015), p. 9.

[12] Jônatas Davi Paganini. *Downsampling in the database: How data locality can improve data analysis*. www.timescale.com/blog. Feb. 2023. URL: https://www.timescale.com/blog/downsampling-in-the-database-how-data-locality-can-improve-data-analysis/.

[13] Benjamin Raskin and Niknunj Aggarwal. *The billion data point challenge: Building a query engine for high cardinality time series data*. uber.com/billion-data-point-challenge. Dec. 2018. URL: https://www.uber.com/en-BE/blog/billion-data-point-challenge/.

[14] Sveinn Steinarsson. "Downsampling Time Series for Visual Representation". en. MA thesis. University of Iceland, 2013. DOI: http://hdl.handle.net/1946/15343. URL: http://hdl.handle.net/1946/15343.

[15] Uwe Jugel, Zbigniew Jerzak, Gregor Hackenbroich, and Volker Markl. "M4: a visualization-oriented time series data aggregation". en. In: *Proceedings of the VLDB Endowment* 7.10 (June 2014), pp. 797–808. ISSN: 2150-8097. DOI: 10.14778/2732951.2732953. URL: https://dl.acm.org/doi/10.14778/2732951.2732953 (visited on 10/14/2022).

[16] David H Douglas and Thomas K Peucker. "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature". In: *Cartographica: the international journal for geographic information and geovisualization* 10.2 (1973), pp. 112–122. DOI: 10.3138/FM57-6770-U75U-7727.

[17] Tak-chung Fu, Fu-lai Chung, Robert Luk, and Chak-man Ng. "Representing financial time series based on data point importance". In: *Engineering Applications of Artificial Intelligence* 21.2 (Mar. 2008), pp. 277–300. ISSN: 09521976. DOI: 10.1016/j.engappai.2007.04.009. URL: https://linkinghub.elsevier.com/retrieve/pii/S0952197607000577 (visited on 02/28/2024).

[18] Mahes Visvalingam and J. Duncan Whyatt. "Line generalisation by repeated elimination of points". In: *Cartographic Journal* 30 (1993), pp. 46–51. URL: https://api.semanticscholar.org/CorpusID:54049050.

[19] Ben Shneiderman. "The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations". en. In: *Proceedings 1996 IEEE symposium on visual languages*. IEEE. 1996, pp. 336–343. DOI: 10.1016/B978-155860915-0/50046-9.

[20] James Walker, Rita Borgo, and Mark W. Jones. "TimeNotes: A Study on Effective Chart Visualization and Interaction Techniques for Time-Series Data". en. In: *IEEE Transactions on Visualization and Computer Graphics* 22.1 (Jan. 2016), pp. 549–558. ISSN: 1077-2626. DOI: 10.1109/TVCG.2015.2467751. URL: http://ieeexplore.ieee.org/document/7192735/ (visited on 03/25/2022).

[21] Syeda Noor Zehra Naqvi, Sofia Yfantidou, and Esteban Zimányi. "Time series databases and influxdb". In: *Studienarbeit, Université Libre de Bruxelles* 12 (2017), pp. 1–44.

[22] Xin Chen, Jian Zhang, Chi-Wing Fu, Jean-Daniel Fekete, and Yunhai Wang. "Pyramid-based Scatterplots Sampling for Progressive and Streaming Data Visualization". In: *IEEE Transactions on Visualization and Computer Graphics* 28.1 (Jan. 2022), pp. 593–603. ISSN: 1077-2626, 1941-0506, 2160-9306. DOI: 10.1109/TVCG.2021.3114880. URL: https://ieeexplore.ieee.org/document/9552916/ (visited on 02/16/2024).

[23] Michail Schwab, David Saffo, Nicholas Bond, Shash Sinha, Cody Dunne, Jeff Huang, James Tompkin, and Michelle A. Borkin. "Scalable Scalable Vector Graphics: Automatic Translation of Interactive SVGs to a Multithread VDOM for Fast Rendering". In: *IEEE Transactions on Visualization and Computer Graphics* 28.9 (Sept. 1, 2022), pp. 3219–3234. ISSN: 1077-2626, 1941-0506, 2160-9306. DOI: 10.1109/TVCG.2021.3059294. URL: https://ieeexplore.ieee.org/document/9354592/ (visited on 02/16/2024).

[24] Evgeniy Yur'evich Gorodov and Vasiliy Vasil'evich Gubarev. "Analytical Review of Data Visualization Methods in Application to Big Data". en. In: *Journal of Electrical and Computer Engineering* 2013 (2013), pp. 1–7. ISSN: 2090-0147, 2090-0155. DOI: 10.1155/2013/969458. URL: http://www.hindawi.com/journals/jece/2013/969458/ (visited on 03/05/2023).

[25] Nikos Bikakis. "Big Data Visualization Tools". en. In: *arXiv:1801.08336 [cs]* (Feb. 2018). URL: http://arxiv.org/abs/1801.08336 (visited on 01/26/2022).

[26] James A. Bednar and Julia Signell. *Common plotting pitfalls that get worse with large data*. github.com/holoviz/datashader/examples/user_guide/1_Plotting_Pitfalls.ipynb.

[27] M. A. Breddels. "Interactive (statistical) visualisation and exploration of a billion objects with vaex". en. In: *Proceedings of the International Astronomical Union* 12.S325 (Oct. 2016), pp. 299–304. ISSN: 1743-9213, 1743-9221. DOI: 10.1017/S1743921316012795. URL: https://www.cambridge.org/core/product/identifier/S1743921316012795/type/journal_article (visited on 03/28/2022).

[28] Holoviz-community. *Datashader, quickly and accurately render even the largest data*. https://github.com/holoviz/datashader.

[29] Joseph M Hellerstein, Ron Avnur, Andy Chou, Christian Hidber, Chris Olston, Vijayshankar Raman, Tali Roth, and Peter J Haas. "Interactive data analysis: The control project". In: *Computer* 32.8 (1999), pp. 51–59. DOI: 10.1109/2.781635.

[30] Kexin Rong and Peter Bailis. "ASAP: prioritizing attention via time series smoothing". en. In: *Proceedings of the VLDB Endowment* 10.11 (Aug. 2017), pp. 1358–1369. ISSN: 2150-8097. DOI: 10.14778/3137628.3137645. URL: https://dl.acm.org/doi/10.14778/3137628.3137645 (visited on 01/24/2023).

[31] Paul Rosen, Ashley Suh, Christopher Salgado, and Mustafa Hajij. *TopoLines: Topological Smoothing for Line Charts*. Apr. 3, 2020. arXiv: 1906.09457[cs]. URL: http://arxiv.org/abs/1906.09457 (visited on 03/01/2024).

[32] Guangtao Zhai and Xiongkuo Min. "Perceptual image quality assessment: a survey". In: *Science China Information Sciences* 63 (2020), pp. 1–52. DOI: 10.1007/s11432-019-2757-1.

[33] Tak-chung Fu, Ying-kit Hung, and Fu-lai Chung. "Improvement algorithms of perceptually important point identification for time series data mining". In: *2017 IEEE 4th International Conference on Soft Computing & Machine Intelligence (ISCMI)*. 2017 IEEE 4th International Conference on Soft Computing & Machine Intelligence (ISCMI). Mauritius: IEEE, Nov. 2017, pp. 11–15. ISBN: 978-1-5386-1314-6. DOI: 10.1109/ISCMI.2017.8279589. URL: http://ieeexplore.ieee.org/document/8279589/ (visited on 02/16/2024).

[34] Amaia Gil, Marco Quartulli, Igor G Olaizola, and Basilio Sierra. "Towards smart data selection from time series using statistical methods". In: *IEEE Access* 9 (2021), pp. 44390–44401. DOI: 10.1109/ACCESS.2021.3066686.

[35] Puleum Bae, Keun-Woo Lim, Woo-Sung Jung, and Young-Bae Ko. "Practical implementation of M4 for web visualization service". In: *Journal of Communications and Networks* 19.4 (2017), pp. 384–391. DOI: 10.1109/JCN.2017.000062.

[36] Wenzhong Shi and ChuiKwan Cheung. "Performance evaluation of line simplification algorithms for vector generalization". In: *the cartographic journal* 43.1 (2006), pp. 27–44. DOI: 10.1179/000870406X93490.

[37] Paul Rosen and Ghulam Jilani Quadri. "LineSmooth: An Analytical Framework for Evaluating the Effectiveness of Smoothing Techniques on Line Charts". In: *IEEE Transactions on Visualization and Computer Graphics* 27.2 (Feb. 2021), pp. 1536–1546. ISSN: 1077-2626, 1941-0506, 2160-9306. DOI: 10.1109/TVCG.2020.3030421. URL: https://ieeexplore.ieee.org/document/9222269/ (visited on 02/21/2024).

[38] Steven M Pincus. "Approximate entropy as a measure of system complexity." In: *Proceedings of the national academy of sciences* 88.6 (1991), pp. 2297–2301.

[39] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. "The UCR time series archive". In: *IEEE/CAA Journal of Automatica Sinica* 6.6 (2019), pp. 1293–1305. DOI: 10.1109/JAS.2019.1911747.

[40] Javier Franco, Ander Garcia, and Amaia Gil. "Multivariate Adaptive Downsampling Algorithm for Industry 4.0 Data Visualization". In: *16th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2021)*. Springer. 2022, pp. 588–597. DOI: 10.1007/978-3-030-87869-6_56.

[41] Christopher Mutschler, Holger Ziekow, and Zbigniew Jerzak. "The DEBS 2013 grand challenge". In: *Proceedings of the 7th ACM international conference on Distributed event-based systems*. 2013, pp. 289–294. DOI: 10.1145/2488222.2488283.

[42] Zbigniew Jerzak, Thomas Heinze, Matthias Fehr, Daniel Gröber, Raik Hartung, and Nenad Stojanovic. "The DEBS 2012 grand challenge". In: *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*. 2012, pp. 393–398. DOI: 10.1145/2335484.2335536.

[43] Prasoon Kottarathil. *Bitcoin Historical Dataset*. kaggle.com/datasets/btcinusd. Mar. 2022. URL: https://www.kaggle.com/datasets/prasoonkottarathil/btcinusd.

[44] Zhou Wang and Alan C Bovik. "Mean squared error: Love it or leave it? A new look at signal fidelity measures". In: *IEEE signal processing magazine* 26.1 (2009), pp. 98–117. DOI: 10.1109/MSP.2008.930649.

[45] De Rosal Igantius Moses Setiadi. "PSNR vs SSIM: imperceptibility quality assessment for image steganography". In: *Multimedia Tools and Applications* 80.6 (2021), pp. 8423–8444. DOI: 10.1007/s11042-020-10035-z.

[46] Jim Nilsson and Tomas Akenine-Möller. "Understanding ssim". In: *arXiv preprint arXiv:2006.13846* (2020).

[47] Chaofeng Li and Alan C Bovik. "Content-partitioned structural similarity index for image quality assessment". In: *Signal Processing: Image Communication* 25.7 (2010), pp. 517–526.

[48] Dae Hyun Kim, Vidya Setlur, and Maneesh Agrawala. "Towards understanding how readers integrate charts and captions: A case study with line charts". In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 2021, pp. 1–11. DOI: 10.1145/3411764.3445443.

[49] Kaggle-inc. *Kaggle's State of Machine Learning and Data Science 2022*. English. Tech. rep. Oct. 2022. URL: https://www.kaggle.com/kaggle-survey-2022.

[50] William J Leler. "Human vision, anti-aliasing, and the cheap 4000 line display". In: *ACM Siggraph Computer Graphics* 14.3 (1980), pp. 308–313. DOI: 10.1145/965105.807509.

[51] Umme Sara, Morium Akter, and Mohammad Shorif Uddin. "Image quality assessment through FSIM, SSIM, MSE and PSNR—a comparative study". In: *Journal of Computer and Communications* 7.3 (2019), pp. 8–18. DOI: 10.4236/jcc.2019.73002.

4

Magnitude and Rotation Invariant Detection of Transportation Modes

In prior work, we demonstrated that traditional machine learning approaches can perform on par with deep learning for PSG-based sleep stage scoring [1], which led to the formulation of **RG2-B**. Building on this foundation, this chapter explores the effectiveness of traditional machine learning in the context of a smartphone movement data use case. Specifically, we detail our ML pipeline for the 2024 Sussex-Huawei locomotion-transportation (SHL) challenge.

The SHL dataset consists of 1,088 hours of labeled training data (15GB in binary format), with this year's objective being the classification of transportation modes using 5-second of movement data $\in \{ACC,\ GYR,\ MAG\}$. Additionally, for each 5-second window, one sensor modality was randomly missing, requiring solutions to handle missing sensor data. During the competition, we encountered significant challenges in generalizing from the training set to the validation set, largely due to a noticeable distribution shift in the feature space. By focusing on mitigating this distribution shift and implementing rotation-invariant feature transformations to handle unknown device orientations, we successfully improved generalization, ultimately securing first place in the challenge.

My (joint) contributions to this chapter are as follows:

- Interactively visualizing all 4B+ data points in the train set using Plotly-Resampler.
- Observing a large distribution shift between the train and test data.

- Designing magnitude- and rotation-invariant feature transformations.
- Evaluating various feature combinations and signal transformations.
  - Analyzing the effect of z-normalization prior to PSD based feature-extraction.
  - Using visual-analytics to select informed PSD energy bands.
  - Performing an ablation on the effectiveness of various signal transformations.
  - Observing the poor performance of the widely used SMV-based transformations.
- Developing the final ML model.

Magnitude and Rotation Invariant Detection of Transportation Modes with Missing Data Modalities

Jeroen Van Der Donckt [1], Jonas Van Der Donckt [1], and Sofie Van Hoecke

Published in proceedings of the "ACM International Joint Conference on Pervasive and Ubiquitous Computing, 2024, pp. 597-602"

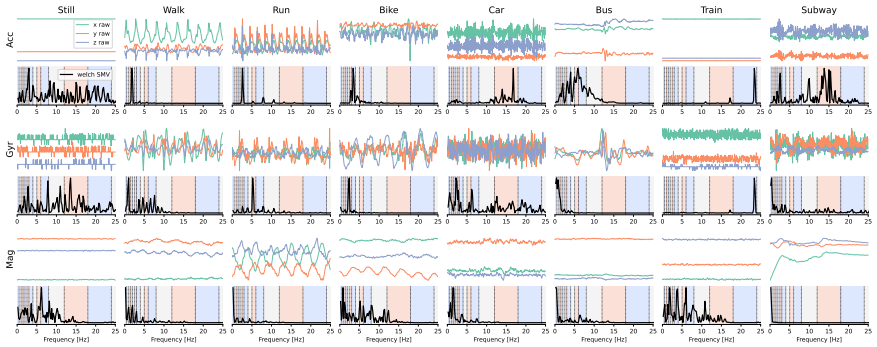


Figure 4.1: Visual representation of utilized frequency bins (in the feature vector) for the power spectrum density of the signal magnitude vector per signal modality (rows) for each transportation mode (columns).

Abstract This work presents the solution of the Signal Sleuths team for the 2024 SHL recognition challenge. The challenge involves detecting transportation modes using shuffled, non-overlapping 5-second windows of phone movement data, with exactly one of the three available modalities (accelerometer, gyroscope, magnetometer) randomly missing. Data analysis indicated a significant distribution shift between train and validation data, necessitating a magnitude and rotation-invariant approach. We utilize traditional machine learning, focusing on robust processing, feature extraction, and rotation-invariant aggregation. An ablation study showed that relying solely on the frequently used signal magnitude vector results in the poorest performance. Conversely, our proposed rotation-invariant aggregation demonstrated substantial improvement over using rotation-aware features, while also reducing the feature vector length. Moreover, z-normalization proved crucial for creating robust spectral features.

4.1 Introduction

Accurate and near-real-time detection of human transportation modes is crucial for applications such as automatic activity recognition on smartphones [2] or smart-

[1] Contributed equally

watches [3], route recommendations, and context-aware service adaptations (e.g. service switching to car mode) [4]. Moreover, through ubiquitous devices such as smartphones or wearable devices, determining these transportation modes enables longitudinal behavior monitoring, which is essential for chronic disease management and follow-up [5, 6].

However, creating a reference dataset that includes qualitatively annotated transportation mode data, representative of real-life settings, is challenging. To address this, the University of Sussex-Huawei locomotion-transportation (SHL) dataset was developed. The SHL dataset contains qualitatively annotated longitudinal locomotion data recorded via smartphones by three participants [7, 8]. To capture the variability of typical phone carry positions, participants carried four smartphones on different body locations (i.e., hand, torso, hip, and bag). Transportation modes in the SHL dataset were precisely post-annotated using 30-second interval images to minimize recall errors [8].

Given our experience with tackling multimodal time series classification using traditional machine learning, we chose to not apply deep learning for this challenge [1]. Our proposed pipeline consists of (i) processing, (ii) feature extraction, and (iii) a traditional machine learning model (i.e., CatBoost - gradient boosted trees) for each missing modality. In particular, we aim to construct an efficient yet performant pipeline, allowing us to investigate the impact of various processing steps and our novel rotation-invariant feature aggregation, over different feature subsets.

4.2 SHL 2024 Challenge Dataset

The 2024 SHL challenge, now in its sixth iteration, aims to detect transportation modes using multimodal sensor data from a single smartphone. Specifically, non-overlapping 5-second windows of the smartphone's accelerometer (Acc), gyroscope (Gyr), and magnetometer (Mag) data sampled at 100Hz are provided, requiring the prediction of the current locomotion mode at a sample-level (500 predictions per window). This year's challenge is particularly difficult due to four key aspects. First, the non-continuous (i.e., shuffled) test set makes post-processing predictions infeasible.

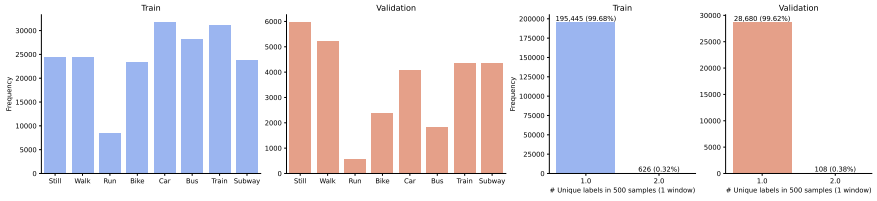


Figure 4.2: Label distribution for train and validation dataset (column 1-2) and number of unique labels per window (column 3-4).

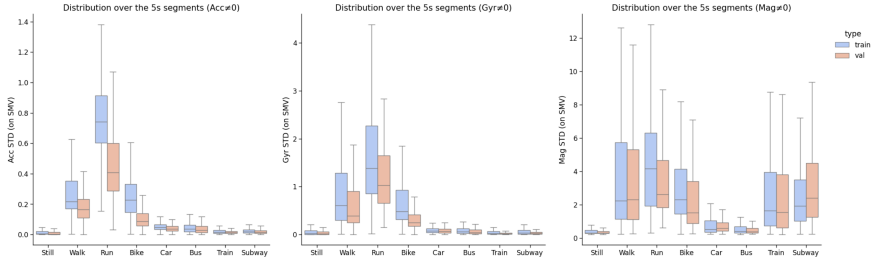


Figure 4.3: Data distribution (of the standard deviation (SD)) for the train and validation dataset.

This way, the organizers aim to evaluate the ability to detect locomotion modes in near-real time. Second, the test dataset does not provide the smartphone location (e.g., hands, hips, torso, or bag), necessitating a location-independent model. Third, as (exactly) one of the data modalities, i.e. Acc, Gyr, or Mag, is masked with zeros in both the validation and test datasets, the approach must be robust to missing data modalities. Fourth, the train dataset consists of data from a single user (subject 1), while the validation and test datasets are a mix of data from subjects 2 and 3, encouraging the development of user-independent models.

4.2.1 Exploratory Data Analysis

Figure 4.2 shows the distribution of transportation modes (i.e., the labels) and the number of unique modes per window in the train and validation datasets. Given that fewer than 0.4% of the 5s windows contain multiple labels, we adapted the objective to predicting a single activity label for each window.

Figure 4.3 compares the distribution (standard deviation) of the signal magnitude vector (SMV) for each 5-second window across the different labels in the train and validation datasets. To ensure a fair comparison, the validation data was trimmed (per modality) to include only windows where the visualized modality was not missing. It is important to perform this comparison across different labels, as each locomotion mode can affect the feature distribution. From Figure 4.3, we observe a distinct distribution shift between the train and validation data, especially for the Acc and Gyr modalities. This shift is most noticeable for the walk, run, and bike locomotion modes. In contrast, the Mag data does not demonstrate such a shift. There may be several causes for this distribution shift, such as variability between recording devices (e.g., smartphone firmware [8]) or variability between subjects, as the train and validation datasets comprise different users. Consequently, solutions trained on solely the train data demonstrate limited generalizability. Particularly, Widhalm P. et al. [9] observed a significant drop in prediction accuracy between train and validation data. Addressing this distribution shift will therefore be a key aspect in this year's challenge.

By utilizing our `plotly-resampler` toolkit [10], we were capable of effectively

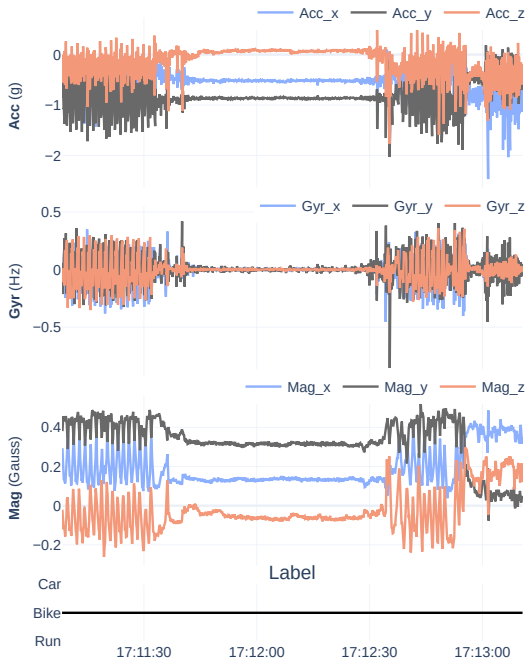


Figure 4.4: Exploratory time series visualization of phone movement data from train set (location = Hips) and labels.

analyzing all modalities of the train and validation data along with the labels, totaling ±4.5B data points, through interactive line chart visualization. This analysis provided insights into the relationship between the training/validation data and the labels. Note that this analysis could not be performed on the test set, as this data was shuffled, thereby removing the temporal aspect. Figure 4.4 presents an interesting excerpt from this analysis. The continuously labeled "Bike" data contains a segment of about 1 minute with nearly no movement (i.e., 17:11:30-17:12:30). This does not necessarily indicate mislabeled data as, for instance, the subject could be waiting at a stoplight. Evidently, postprocessing demonstrated substantial performance improvements (absolute 10% gain) on the SHL dataset [11]. However, relying solely on shuffled 5-second windows and being unable to use postprocessing—which demonstrated substantial performance improvements (absolute 10% gain) on the SHL dataset [11]—makes accurately classifying such small windows nearly impossible.

4.3 Algorithm Pipeline

When devising our approach, we focused on two aspects: (i) rotation invariance due to the unknown phone position in the test set, and (ii) magnitude invariance to cope with

the distribution shift observed between the train and validation set (see Figure 4.3). The resulting pipeline consists of 3 steps; (1) processing, (2) feature extraction, and (3) modeling.

4.3.1 Processing

During data loading, we scale each axis of the various modalities by fixed constants to convert them into more interpretable units. Specifically, accelerometer data is divided by 9.81 to convert m/s² to g units, gyroscope data is divided by 2π to convert rad/s to Hz, and magnetometer data is divided by 100 to convert μT to Gauss. Figure 4.4 displays these scaled signals.

Next, we aim to derive new signals (per modality) that are more robust to sensor placement and orientation. This involves computing the SMV over the (transformed) x, y, and z axes for each modality $\mathcal{M} \in \{Acc, Gyr, Mag\}$ [11]. The SMV for each modality is computed using the formula:

$$SMV_{t,\mathcal{M}} = \sqrt{\mathcal{M}_{t,x}^2 + \mathcal{M}_{t,y}^2 + \mathcal{M}_{t,z}^2}$$

After computing the SMV on the raw data, we apply several axis transformations: the first-order gradient, the second-order gradient, and the integral—after which the SMV is calculated on this transformed data.
The first and second-order gradients for each modality $\mathcal{M}$ and axis $a \in \{x, y, z\}$ are computed using the `numpy.gradient` function:

$$dt^1_{t,\mathcal{M},a} = \frac{\mathcal{M}_{t+1,a} - \mathcal{M}_{t-1,a}}{2}$$

$$dt^2_{t,\mathcal{M},a} = \frac{dt^1_{t+1,\mathcal{M},a} - dt^1_{t-1,\mathcal{M},a}}{2}$$

The integral for each modality $\mathcal{M}$ and axis $a \in \{x, y, z\}$ is computed via the `scipy.cumtrapz` function, based on the formula:

$$Integral_{t,\mathcal{M},a} = \int_0^t \mathcal{M}_{\tau,a}\, d\tau$$

By computing the SMV on both the raw data and the outputs of these three axis transformations, we derive 4 rotation-invariant signals per modality: SMV, $SMV\left(dt^1\right)$, $SMV\left(dt^2\right)$, and $SMV\left(Integral\right)$. These four signals, together with the raw (i.e., three-axial) signals, will be considered for feature extraction in the next step.

4.3.2 Feature Extraction

Table 4.1 provides an overview of the extracted features for each available signal. Our `tsflex` toolkit was used for convenient feature extraction [12]. Entropy, fractal, and

Hjorth features were computed using the `antropy` toolkit [13], while other features were derived from functions provided by the `numpy` and `scipy` libraries [14, 15].

Most spectral features represent frequency-band energy. The frequency bounds of these bands were determined using a visual analytic approach, as illustrated in the teaser Figure 4.1.

To create a more magnitude-invariant feature set, we refrained from including magnitude-related features from the time domain (e.g., quantiles, peak-to-peak, mean, std). Moreover, we apply z-normalization before computing the discrete cosine transform (DCT) and power spectral density (PSD) features. We hypothesize that z-normalization can make the spectral representation features less sensitive to amplitude variations, enhancing magnitude robustness and thereby reducing susceptibility to distribution shifts.

In total, 70 features were extracted per signal, resulting in 980 features overall: 2 modalities (as one of the three is always missing) $\times$ (3 raw + 4 processed signals) $\times$ 70 features. In Section 4.4, we present the impact of certain (processed) signal features and their combinations on model performance.

Table 4.1: Overview of computed features per signal (N=70).

Domain	Notes	Features	n
Spectral	Computed on both PSD and DCT magnitudes of the z-normalized signals. So n needs to be multiplied by two.	Energy in frequency bands (Hz): [.1-.5, .5-1, 1-1.5, 1.5-2, 2-2.5, 2.5-3, 3-4, 4-5, 5-6, 6-8, 8-12, 12-18, 18-24, 24-28, 28-32, 32-40, 40-50]	17
		Spectral centroid & bandwidth	2
		Ratio between two highest amplitudes	1
		Amplitude: max, std & skew	3
		Top (peak) frequency	1
		Mean, std, and skew (Hz) of top 5 frequencies with highest amplitude	3
	PSD	Spectral entropy	1
Time	autocorrelation (ACF)	Absolute mean, skew, std	3
		Most prominent ACF frequency	1
		Zero crossing rate & slope sign change	2
		Differential & spectral entropy	2
	Signal as-is	Differential entropy	1
		Mean crossing rate	1
		skew & kurtosis	2
		Hjorth mobility & complexity	2
		Katz fractal dimension	1

4.3.3 Model

We focused on traditional machine learning models, specifically investigating the performance of CatBoost, a gradient-boosted trees algorithm known for its strong performance without parameter tuning [16], making it suitable for an ablation study on feature subsets. We limited the CatBoost model to 1,000 iterations (trees) and used "Balanced" as the `auto_class_weight` parameter. To cope with missing data, we created a new model for each missing modality configuration by excluding the missing modality from the (training) feature vector. Consequently, three different models were constructed, each using only two modalities.

4.3.4 Postprocessing

In contrast to most traditional approaches, we did not train on the entire train dataset to make final predictions on the test data. Instead, we leveraged K-fold cross-validation (CV), making predictions after fitting each fold, and then aggregated these predictions by utilizing majority voting (MV). We deliberately opted for a 3-fold CV, as using an odd number as K results in only 50% data overlap across the training folds. Using this approach, we effectively perform bootstrapping through CV on the training data.

4.3.5 Rotation Invariant Aggregation

To improve the robustness of the feature set computed on the raw axis signals (i.e., non-SMV-transformed), we proposed and examined three novel *rotation-invariant statistical aggregation* approaches. Specifically, the raw $\{x, y, z\}$ signal features are condensed to summary statistics using: `stat2`: { mean, std }, `stat3`: { mean, std, skew }, or `sort`: { min, mid, max }. These approaches effectively discard axial information while retaining the overall feature information. Note that `stat2` converts the three $\{x, y, z\}$ features into two summary statistics (resulting in a 1/3 compression ratio), while `stat3` and `sort` maintain the same input-output ratio.

Sensor orientation affects the sign of axial measurements. For instance, flipping a sensor along one axis produces opposite signs for the measurements along that axis. Consequently, rotation-invariant statistical aggregation is only relevant for sign-invariant features (e.g., kurtosis, mean crossing rate, and spectral domain features). As such, from Table 4.1, skewness is the only feature that is influenced by the sign, resulting in it being discarded when performing the rotation-invariant aggregation.

4.4 Experimental Results and Discussion

Experiments were performed on a server computer (Arch Linux), with an AMD Ryzen 5 2600x CPU, 48GB of DDR4 RAM, and an Nvidia RTX 2070 GPU. Table 4.2 presents the out-of-fold (OOF) and validation (Val) macro F1 scores for the various

investigated feature combinations. Each experiment (represented by a row in the table) involved training 12 models: 4 (3 folds + 1 full fit) x 3 (for the 3 missing modalities), taking a total of $\pm$10 minutes to complete. Inference on the validation set took $\pm$2 seconds, including 3-fold MV, with processing and feature extraction taking $\pm$ 2 minutes.

A first notable observation is that using only the `SMV` signal results in the worst validation performance. Yet, `SMV` has been commonly employed as the sole processing configuration in many studies [11, 9]. Second, our proposed rotation-invariant statistical aggregation approaches (i.e. $\texttt{rot\_inv}_{stat2}$, $\texttt{rot\_inv}_{stat3}$, and $\texttt{rot\_inv}_{sort}$) all outperform the `raw` configuration (i.e., no feature aggregation) and the `SMV`) processing. These findings highlight the potential of statistical aggregation to effectively remove the axial (i.e., rotation-aware) information while preserving overall feature information. Third, we observe a consistent performance improvement when applying cross-fold-based majority voting (Val_{MV}) compared to a model fully trained on the train set (Val).

Analyzing the performance of SMV computed on gradient and integral axis transformations reveals that incorporating the second-order gradient ($dt2$) yields the highest performance boost, followed by the first-order gradient ($dt1$), with the $integral$ contributing the least. Interestingly, combining $dt1$ and $dt2$ provides a lower boost than combining either gradient transformation with the $integral$. This might be attributed to the high correlation between the gradient transformations, making their combination less informative.

When combining the rotation-invariant statistical aggregation with the SMV and the SMVs on the gradients/integral axis transformations, we observe similar trends. Among the rotation-invariant statistical aggregations, `stat2` and `sort` exhibit the best performance. Whereas again, `dt2` performs slightly better than `dt1`, with `integral`

Table 4.2: Macro F1 scores of the different investigated feature combinations.

	Overall		Acc = 0		Gyr = 0		Mag = 0		
Configuration	**Val**	**Val$_{MV}$**	**OOF**	**Val**	**OOF**	**Val**	**OOF**	**Val**	**# feats**
rot_inv$_{sort}$ + SMV + SMV$_{dt2}$	0.7244	0.7255	0.8039	0.7512	0.8538	0.7120	0.8177	0.6713	694
rot_inv$_{stat2}$ + SMV + SMV$_{dt2}$	0.7197	0.7255	0.8019	0.7474	0.8530	0.7179	0.8168	0.6720	556
rot_inv$_{sort}$ + SMV + SMV$_{dt1}$	0.7149	0.7195	0.8032	0.7386	0.8524	0.7105	0.8175	0.6684	694
rot_inv$_{stat2}$ + SMV + SMV$_{dt1}$	0.7146	0.7174	0.8004	0.7348	0.8523	0.7121	0.8163	0.6681	556
rot_inv$_{sort}$ + SMV	0.7006	0.7048	0.7928	0.7087	0.8488	0.7052	0.8150	0.6586	554
rot_inv$_{stat2}$ + SMV	0.6986	0.7040	0.7911	0.7045	0.8472	0.7102	0.8139	0.6562	416
SMV + SMV$_{dt1}$ + SMV$_{dt2}$ + SMV$_{integral}$	0.6983	0.7012	0.7785	0.7335	0.8274	0.6898	0.7907	0.6356	560
SMV + SMV$_{dt2}$ + SMV$_{integral}$	0.6940	0.6969	0.7734	0.7298	0.8231	0.6804	0.7904	0.6349	420
SMV + SMV$_{dt1}$ + SMV$_{integral}$	0.6903	0.6919	0.7723	0.7193	0.8226	0.6877	0.7888	0.6296	420
SMV + SMV$_{dt1}$ + SMV$_{dt2}$	0.6889	0.6952	0.7654	0.7240	0.8292	0.6842	0.7860	0.6392	420
SMV + SMV$_{dt2}$	0.6858	0.6876	0.7586	0.7190	0.8239	0.6694	0.7857	0.6339	280
SMV + SMV$_{dt1}$	0.6832	0.6864	0.7566	0.7073	0.8258	0.6822	0.7860	0.6305	280
rot_inv$_{sort}$	0.682	0.6828	0.7840	0.7031	0.8226	0.6728	0.7924	0.6312	414
rot_inv$_{stat3}$	0.6751	0.6749	0.7790	0.6893	0.8181	0.6725	0.7892	0.6235	414
rot_inv$_{stat2}$	0.6744	0.6758	0.7795	0.6913	0.8186	0.6716	0.7890	0.6221	276
raw	0.6681	0.6695	0.7828	0.6782	0.8235	0.6700	0.7937	0.6134	420
SMV + SMV$_{integral}$	0.6579	0.6615	0.7450	0.6680	0.8038	0.6632	0.7806	0.6131	280
SMV	0.6511	0.6542	0.7215	0.6564	0.8036	0.6547	0.7728	0.6166	140

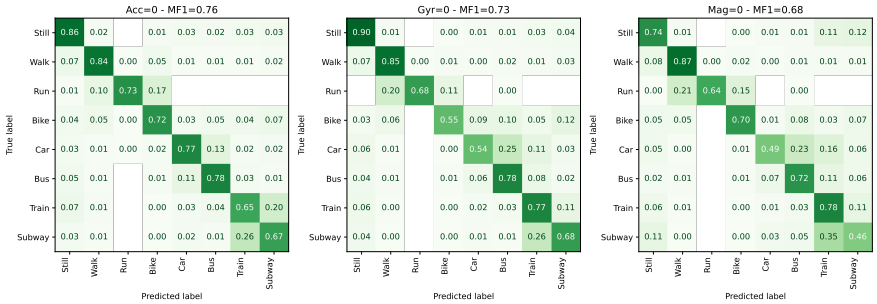


Figure 4.5: Confusion matrix for the final model (i.e., rot_inv$_{stat2}$ + SMV + SMV$_{dt2}$) on validation dataset (including all 4 phone locations). The validation predictions were obtained using the 3-fold MV method, as described in Section 4.3.4.

resulting in the smallest improvement.

Although not presented in Table 4.2, we also investigated the impact of z-normalization before computing the PSD and DCT features. On average, using z-normalization resulted in a consistent 1-1.5% (absolute) improvement in macro F1 on the validation data.

4.4.1 Impact Distribution Shift

A consistent trend observed in Table 4.2 and in Figure 4.5 is that missing magnetometer data (i.e., `MAG = 0`) results in the poorest validation performance, despite having a comparable OOF score to the two other scenarios. This may be attributable to the magnetometer being the only modality that does not suffer from a substantial distribution shift, as shown in Figure 4.3. The smallest drop in performance between the OOF and the Val scores is observed for the `ACC = 0` case, likely due to the accelerometer data exhibiting the largest distribution shifts, as indicated by Figure 4.3.

4.4.2 Final Model

For the final model, we selected the rot_inv$_{stat2}$ + SMV + SMV$_{dt2}$ feature configuration, as the rot_inv$_{sort}$ variant of this configuration resulted in the same Val$_{MV}$ score while having a larger feature vector (see Table 4.2).

Figure 4.5 shows the confusion matrices for this final model. An unexpected observation is that the models do not distinguish "Run" that well. Given the pronounced frequency-component of running, as illustrated in Figure 4.1, we anticipated better performance in recognizing this activity. We hypothesize that training on data from only one user might have caused the model to capture that user's specific running behavior too closely, as substantiated by the large distribution shift for running, observed in Figure 4.3. Consistent with previous research, we observe typical train-subway and

car-bus confusion [11]. Finally, the availability of gyroscope data appears to enhance the predictive performance for the "Bike" class.

4.4.3 Improving the Test Score

To generate the final test predictions, we retrained our final models (3-fold CV) using several tricks to improve the test score over the above-reported validation score:

- Exclude hand location: According to the challenge description, the test dataset comprises only torso, hips, and bag locations. Analysis of prediction errors per location indicated that hand location performed the worst, aligning with the observation of Gjoreski et al. [8]. Experimental results indicate an improvement in Val_{MV} from 0.7255 to 0.7506.
- Train on validation data: Since there were no restrictions on what data may be used for the training, including the validation during training should allow the model to better capture (any) remaining distribution shift and thereby learn the variability across multiple subjects.

4.5 Conclusion

This work presents the findings and final approach of the "Signal Sleuths" team for the 2024 Sussex-Huawei Locomotion-Transportation recognition challenge. During exploratory data analysis, we identified a substantial distribution shift between train and validation/test data, making this a key challenge in this year's edition. To address this, we utilized a traditional machine learning pipeline, enabling us to perform an ablation study on various magnitude and rotation-invariant feature configurations. In particular, we performed an ablation study on (i) including features from different processed signals as well as (ii) a novel approach in which we make rotation-aware signals rotation-invariant through statistical aggregation, and (iii) performing z-normalization prior to spectral feature computation. Results indicated that solely relying on SMV yields the poorest performance. Moreover, our proposed rotation-invariant statistical aggregation demonstrated a substantial improvement over using rotation-aware features or SMV alone, with the added benefit of reducing the feature vector length, underscoring the efficacy of this novel technique. Furthermore, z-normalization of the data proved to be beneficial when creating robust spectral features. Our final model consists of the best feature subset combination, retrained on both the training and validation data to better capture inter-user variability, while also excluding the hand data as this location is not present in the test data. The recognition result for the testing dataset will be presented in the summary paper of the challenge [17].

References

[1] Jeroen Van Der Donckt, Jonas Van Der Donckt, Emiel Deprost, Nicolas Vandenbussche, Michael Rademaker, Gilles Vandewiele, and Sofie Van Hoecke. "Do not sleep on traditional machine learning: Simple and interpretable techniques are competitive to deep learning for sleep scoring". In: *Biomedical Signal Processing and Control* 81 (2023), p. 104429.

[2] Marija Stojchevska, Mathias De Brouwer, Martijn Courteaux, Bram Steenwinckel, Sofie Van Hoecke, and Femke Ongenae. "Unlocking the potential of smartphone and ambient sensors for ADL detection". In: *Scientific Reports* 14.1 (2024).

[3] Subhas Chandra Mukhopadhyay. "Wearable Sensors for Human Activity Monitoring: A Review". In: *IEEE Sensors Journal* 15.3 (2015), pp. 1321–1330.

[4] Sasank Reddy, Min Mun, Jeff Burke, Deborah Estrin, Mark Hansen, and Mani Srivastava. "Using mobile phones to determine transportation modes". In: *ACM Trans. Sen. Netw.* 6.2 (2010). ISSN: 1550-4859.

[5] Mirza Mansoor Baig, Hamid GholamHosseini, Aasia A. Moqeem, Farhaan Mirza, and Maria Lindén. "A Systematic Review of Wearable Patient Monitoring Systems – Current Challenges and Opportunities for Clinical Adoption". en. In: *Journal of Medical Systems* 41.7 (July 2017), p. 115. ISSN: 0148-5598, 1573-689X. DOI: 10.1007/s10916-017-0760-1. URL: http://link.springer.com/10.1007/s10916-017-0760-1 (visited on 05/24/2023).

[6] Jonas Van Der Donckt, Mathias De Brouwer, Pieter Moens, Marija Stojchevska, Bram Steenwinckel, Stef Pletinck, Nicolas Vandenbussche, Annelis Goris, Koen Paemeleire, Femke Ongenae, et al. "From self-reporting to monitoring for improved migraine management". In: *Engineer meets Physician (EmP)*. 2022.

[7] Lin Wang, Hristijan Gjoreski, Mathias Ciliberto, Sami Mekki, Stefan Valentin, and Daniel Roggen. "Enabling Reproducible Research in Sensor-Based Transportation Mode Recognition With the Sussex-Huawei Dataset". en. In: *IEEE Access* 7 (2019), pp. 10870–10891. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2890793. URL: https://ieeexplore.ieee.org/document/8600317/ (visited on 06/07/2024).

[8] Hristijan Gjoreski, Mathias Ciliberto, Lin Wang, Francisco Javier Ordonez Morales, Sami Mekki, Stefan Valentin, and Daniel Roggen. "The University of Sussex-Huawei Locomotion and Transportation Dataset for Multimodal Analytics With Mobile Devices". en. In: *IEEE Access* 6 (2018), pp. 42592–42604. ISSN: 2169-3536. (Visited on 05/23/2024).

[9] Peter Widhalm, Maximilian Leodolter, and Norbert Brändle. "Top in the Lab, Flop in the Field?: Evaluation of a Sensor-based Travel Activity Classifier with the SHL Dataset". en. In: *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*. Singapore Singapore: ACM, Oct. 2018, pp. 1479–1487. ISBN: 978-1-4503-5966-5. (Visited on 06/07/2024).

[10] Jonas Van Der Donckt, Jeroen Van Der Donckt, Emiel Deprost, and Sofie Van Hoecke. "Plotly-Resampler: Effective Visual Analytics for Large Time Series". In: *2022 IEEE Visualization and Visual Analytics (VIS)*. Oklahoma City, OK, USA: IEEE, Oct. 2022, pp. 21–25. ISBN: 978-1-66548-812-9. DOI: 10.1109/VIS54862.2022.00013. URL: https://ieeexplore.ieee.org/document/9973221/ (visited on 06/16/2023).

[11] Lin Wang, Hristijan Gjoreski, Mathias Ciliberto, Sami Mekki, Stefan Valentin, and Daniel Roggen. "Benchmarking the SHL Recognition Challenge with Classical and Deep-Learning Pipelines". In: *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*. UbiComp '18. Singapore, Singapore: Association for Computing Machinery, 2018, pp. 1626–1635. ISBN: 9781450359665.

[12] Jonas Van Der Donckt, Jeroen Van Der Donckt, Emiel Deprost, and Sofie Van Hoecke. "tsflex: Flexible time series processing & feature extraction". en. In: *SoftwareX* 17 (Jan. 2022), p. 100971. ISSN: 23527110. DOI: 10.1016/j.softx.2021.100971. URL: https://linkinghub.elsevier.com/retrieve/pii/S2352711021001904 (visited on 03/03/2022).

[13] Vallat Rafael. *Antropy: time-efficient algorithms for computing the complexity of time-series.* https://github.com/raphaelvallat/antropy. 2018.

[14] Charles R. Harris et al. "Array programming with NumPy". en. In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. ISSN: 0028-0836, 1476-4687. (Visited on 06/16/2023).

[15] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. "SciPy 1.0: fundamental algorithms for scientific computing in Python". In: *Nature methods* 17.3 (2020), pp. 261–272.

[16] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. "CatBoost: unbiased boosting with categorical features". In: *Advances in neural information processing systems* 31 (2018).

[17] Lin Wang, Mathias Ciliberto, Hristijan Gjoreski, P. Lago, T. Okita, and Daniel Roggen. "Summary of SHL challenge 2024: Motion sensor-based locomotion and transportation mode recognition in missing data scenario". In: 2024.

5

Robust Workout Activity Detection with Multi-Wearable Data Augmentation

Building on the insights from Chapter 4, this chapter further explores the application of traditional machine learning methods for wearable time series data by participating in the 2024 WEAR dataset challenge, contributing to **RG2-B**. Unlike the pre-segmented data used in the other challenge, the WEAR dataset consists of continuous multi-wearable recordings, allowing competitors to define their own segmentation strategies.

Leveraging our `tsflex` toolkit (Appendix B and our prior work [1], which demonstrated the effectiveness of multi-resolution features for constructing expressive feature vectors, we refined this approach for this use case. The continuous nature of the dataset also enabled the use of post-processing techniques as well as generating visualizations of model outputs, offering valuable insights into the sources of wrong predictions.

During data analysis, we identified inconsistencies in wearable orientations, both within and across recordings. To address this variability, we developed novel augmentation strategies designed to enhance model robustness, effectively functioning as mini-ensembles. By combining these techniques, we secured first place in the competition, outperforming all other participants, as well as SotA HAR ML approaches mentioned in the dataset paper [2].

My (joint) contributions to this chapter are as follows:

- Observing inconsistencies in device orientation within and between recordings.
- Designing rotation-invariant feature augmentations.

- Evaluating various feature combinations and signal transformations.
  - Performing an ablation on the effectiveness of various signal transformations.
  - Observing poor performance of SMV-based transformations.
- Implementing vectorized probabilistic smoothing using a weighting function (e.g., Gaussian, uniform).
- Visually inspecting model predictions
- Developing the final ML model.

Left-Right Swapping and Upper-Lower Limb Pairing for Robust Multi-Wearable Workout Activity Detection

Jonas Van Der Donckt [1], Jeroen Van Der Donckt [1], and Sofie Van Hoecke

Published in proceedings of the "ACM International Joint Conference on Pervasive and Ubiquitous Computing, 2024, pp. 545-550"

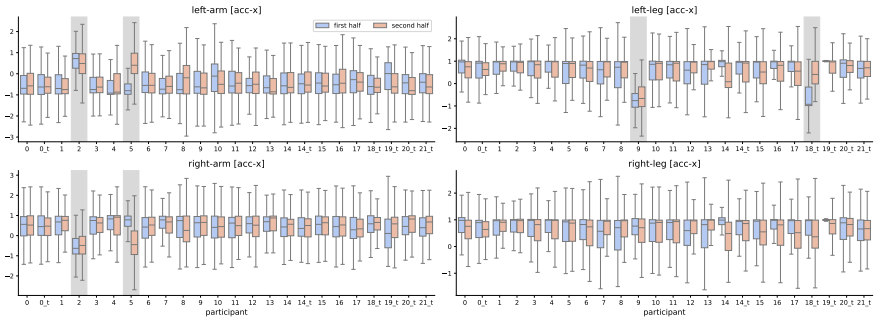


Figure 5.1: Distribution of x-axis accelerometer data for each participant recording, split into first and second halves, for all four limbs. Deviating patterns are highlighted in gray, $\_t$ indicates whether the participant recording belongs to the test data.

Abstract This work presents the solution of the Signal Sleuths team for the 2024 HASCA WEAR challenge. The challenge focuses on detecting 18 workout activities (and the null class) using accelerometer data from 4 wearables—one worn on each limb. Data analysis revealed inconsistencies in wearable orientation within and across participants, leading to exploring novel multi-wearable data augmentation techniques. We investigate three models using a fixed feature set: (i) "raw": using all data as is, (ii) "left-right swapping": augmenting data by swapping left and right limb pairs, and (iii) "upper-lower limb pairing": stacking data by using upper-lower limb pair combinations (2 wearables). Our experiments utilize traditional machine learning with multi-window feature extraction and temporal smoothing. Using 3-fold cross-validation, the raw model achieves a macro F1-score of 90.01%, whereas left-right swapping and upper-lower limb paring improve the scores to 91.30% and 91.87% respectively.

5.1 Introduction

Data-driven processing of wearable data facilitates numerous applications, ranging from automatic activity recognition [3, 4], aiding in tracking workout progress or

[1] Contributed equally

gathering long-term health insights [5], to intelligent service adaptations, such as adaptive heart-rate monitoring based on detected activities [6].

Recently, Bock et al. introduced the WEAR dataset [2], which utilizes an acquisition protocol involving four wearable devices (one worn on each limb) collecting three-axial accelerometer data and includes both first- and third-person video data. The dataset consists of 18 participants, each performing 18 workout activities in one or multiple sessions. High-quality activity labels for the 18 workout activities and a "null" class (thus 19 classes in total) were obtained by using the third-person video data. The WEAR dataset stands out due to its combination of both multi-wearable and video data, and the availability of untrimmed, continuous data streams for each session. Moreover, the workouts were conducted across 11 diverse outdoor locations, introducing variability in surface types—ranging from meadows to city parks and concrete walkways.

Given our track record with multimodal time series classification using traditional machine learning, we opted to not employ deep learning for this challenge [1]. Our proposed pipeline consists of (i) multi-window feature extraction, (ii) data augmentation, (iii) a traditional machine learning model (i.e., CatBoost - gradient boosted trees), and (iv) post-processing of model predictions. In particular, we design and evaluate two novel data augmentation techniques for multi-wearable data; *left-right swapping* and *upper-lower limb pairing*. As such, we aim to construct multiple robust models whose predictions are leveraged during post-processing to further enhance the performance.

5.2 2024 WEAR Dataset Challenge

The 2024 HASCA WEAR Challenge aims to detect 18 distinct workout activities (and a null class), using inertial data from multiple wearable devices. Each participant wore a Bangle.js v1 watch [7] on every limb (left-arm, right-arm, left-leg, right-leg) in a fixed orientation. All four wearables collected three-axial accelerometer data at 50 Hz, with a range of [-8g, 8g]. Participants were suggested to perform 18 different workout activities for ± 90 seconds and in two sessions, i.e., $\pm$ 9 activities per session, which were combined into a single continuous recording.

The initial dataset, described in the WEAR dataset paper [2], includes data from 18 participants and serves as the training set for this challenge. The test set contains recordings of 6 participants; of which two participants are also present in the training set. The challenge is evaluated using the sample-wise macro F1 score. Notably, the WEAR dataset paper [2] provides inertial baselines along with their validation procedures, allowing challenge participants to position the performance of their approaches.

A particular interesting aspect of the WEAR Challenge is that the data is provided "as-is" in its untrimmed format, without any segmentation. As such, challenge participants have full flexibility to define their data windows, strides, and post-processing rules.

Although each participant performed the same set of 18 workout activities, they

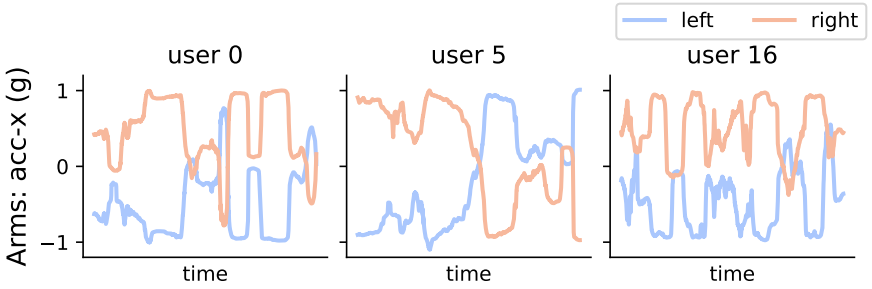


Figure 5.2: Median-smoothed arm accelerometer x-axis data using a 120-second smoothing window for three participants from the training set.

had the freedom to determine the exact activity sequence and how each sequence was performed (multiple short sessions or one large session). This variability makes the dataset more representative of real-life workout scenarios. Additionally, several workout activities are similar to, or variants of others (e.g., lunges vs. lunges complex), adding a significant layer of complexity to the challenge.

5.2.1 Exploratory Analysis

In the WEAR dataset paper, the authors stated that the wearables were worn in a fixed orientation [2]. However, our exploratory analysis, outlined below, revealed several inconsistencies in device orientation both across- and within-participant recordings.

Teaser Figure 5.1 displays the distribution of x-axis accelerometer data for each participant across all four wearables. We focused on the x-axis accelerometer data because, as shown in Figure 1 of the WEAR dataset paper [2], the x-axis should align with the gravity component (i.e., 1 g), with the sign depending on the wearable orientation. In Figure 5.1, we split the data for each participant into the first and second halves, roughly corresponding to the two suggested recording sessions.

From Figure 5.1, we observe that for participant 2, the orientation of the left- and right-arm wearables appears to be inverted in both sessions. Similarly, the left-leg data for participant 9 seems to be inverted in the two sessions. This suggests that participant 9 likely wore the wearable in the opposite orientation, while for participant 2, it is also plausible that the left- and right-arm wearables were swapped.

Additionally, participant 5 and test participant 18 exhibit changes in orientation between their sessions (i.e., between the first and second half). We suspect that test participant 18 wore the wearable in opposite orientation during the first session. For participant 5, it is possible that the left- and right-arm wearables were worn on incorrect arms during the second session, or that they were worn on the correct arm but in opposite orientation. Figure 5.2 complements the observation for participant 5 by showing the median-smoothed x-axis accelerometer signal from the wearables on

Table 5.1: Overview of computed features (N=14).

Domain	Notes	Features	n
	PSD	Spectral entropy	1
Time	RAW / SMV signals	min, max, ptp, iqr	4
		std, skew, kurtosis	3
		Hjorth mobility & complexity	2
		mean crossing rate	1
		differential entropy	1
		Petrosian fractal dimension	1
		Katz fractal dimension	1

both the left and right arms. The switch in the median value midway for participant 5 indicates a change in device orientation. Such within-participant changes in orientation likely result from participants accidentally switching the wearable orientation between sessions, as all sessions are combined into a single user file. We hypothesize that dealing with these variations in orientation, as opposed to incorrectly assuming a fixed orientation, will lead to the design of more robust activity detection models.

Finally, we observed that for user 10, a large segment of left-arm data was missing. This missing data segment was mitigated by imputing the missing segment with the right-arm data. Also for test set participant 19, Figure 5.1 indicates that for the first session, there is nearly no variation in the x-axis data for both legs, contrary to his second session recording and all other participants.

5.3 Algorithm Pipeline

5.3.1 Preprocessing

In addition to the raw data, we will also consider the signal magnitude vector (SMV) for feature extraction. To do so, the SMV is calculated in a preprocessing step for each wearable $\mathcal{W}$ using the formula:

$$SMV_{t,\mathcal{W}} = \sqrt{\mathcal{W}_{t,x}^2 + \mathcal{W}_{t,y}^2 + \mathcal{W}_{t,z}^2}$$

5.3.2 Feature Extraction

To construct a performant feature vector, we relied on multi-resolution feature extraction, a method proven to be highly effective for continuous time series classification and capable of matching SotA deep learning approaches [1]. Figure 5.3 illustrates our utilized multi-resolution window configuration for each prediction time-step. Feature windows that start and end at the prediction time-step are incorporated in the feature

vector. We use a 0.5 second stride, meaning each consecutive time-step has a 0.5 second gap.

Table 5.1 provides an overview of the extracted features on each window. Our `tsflex` toolkit was used for convenient and efficient multi-window feature extraction [8]. Spectral entropy, fractal, and Hjorth features were computed using the `antropy` toolkit [9], while other features were derived from functions provided by the `numpy` and `scipy` libraries [10, 11].

In total, 14 features were extracted per window, resulting in 166 multi-window features per wearable accelerometer axis for each time-step: 2 (start and end) $\times$ 6 windows $\times$ 14 features - 2 (spectral entropy is not computed on 1-second windows). Overall, this results in 1992 features; 166 features x 3 accelerometer axes x 4 wearables.

5.3.3 Data Augmentation

In this study, we propose and investigate three data augmentation techniques aimed at creating more robust models. The first technique, *rotation-invariant* statistical aggregation, is a generic approach that aims to effectively discard axial information while retaining the overall feature information. The other two proposed techniques, *Left-Right swapping* and *Upper-Lower limb pairing*, expand the data and are solely applicable to multi-wearable setups.

5.3.3.1 Rotation-Invariant Aggregation

To improve the robustness of feature sets computed on raw axial signals (i.e., non-SMV transformed), we examined three novel *rotation-invariant statistical aggregation* methods, which are applicable to any three-axial modality, such as gyroscope, magnetometer or here accelerometer data. Specifically, the raw $\{x, y, z\}$ signal features are condensed into summary statistics using: `stat2`: { mean, std }, `stat3`: { mean, std, skew }, or `sort`: { min, mid, max }. Similar to SMV, these methods discard axial information while retaining overall feature information. Notably, `stat2` compresses the three x, y, z features into two summary statistics (a 1/3 compression ratio), while `stat3` and `sort` maintain the same input-output ratio.

5.3.3.2 Left-Right Swapping

left-right swapping (LR-swapping) is a multi-wearable data augmentation technique that expands the feature matrix by including all possible combinations of upper and lower left-right swaps. Specifically, the data is augmented with the following combinations: {"no-swap", "upper LR-swap", "lower LR-swap", "upper & lower LR-swap" }. Remark how the "no-swap" group corresponds to the feature matrix used in the "default" (i.e., raw) configuration. Remark that this swapping can also be performed on

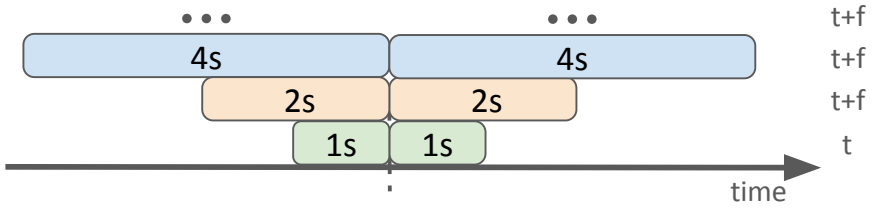


Figure 5.3: Multi-resolution feature extraction. For each prediction timestamp, future and past window sizes of { 1s, 2s, 4s, 8s, 16s, 32s } are used, resulting in 6 × 2 feature windows. For the 1-second window, only time domain features (`t`) are extracted, while for all larger windows, both time and frequency domain features (`t+f`) are computed.

the validation data, which can then be aggregated by using the majority voting across the different swap combinations for the same prediction time-step.

5.3.3.3 Upper-Lower Limb Paring

upper-lower limb pairing (UL-pairing) is a multi-wearable augmentation technique that pairs one upper limb with one lower limb per feature vector, resulting in the use of two wearables per feature vector instead of four. The data is augmented by forming all possible upper-lower limb pair combinations: { left-arm & left-leg, left-arm & right-leg, right-arm & left-leg, right-arm & right-leg }. This approach increases the original feature matrix size by 4× but reduces the number of features (in the vector) by half. Similar to LR-swapping, this technique can be applied to the validation data, followed by prediction time-step aggregation of the different combinations.

5.3.4 Model

Given the power of traditional machine learning models matching deep learning in certain time series classification tasks [1], we focused on traditional machine learning models, specifically evaluating the performance of CatBoost. Catboost is a gradient-boosted trees algorithm known for its strong performance without requiring extensive parameter tuning [12], making it particularly suitable for conducting an ablation study on the above-introduced data augmentation techniques. For our experiments, we limited the CatBoost model to 1,000 iterations (or trees) and set the `auto_class_weight` parameter to "Balanced". To speed up the training, we restricted the tree depth to 5 and used a border count of 32. Other parameters were kept to their default values.

5.3.5 Postprocessing

To improve our predictions, we applied two commonly used postprocessing techniques: *k-fold majority voting*, which exploits model diversity, and *temporal prediction smoothing*,

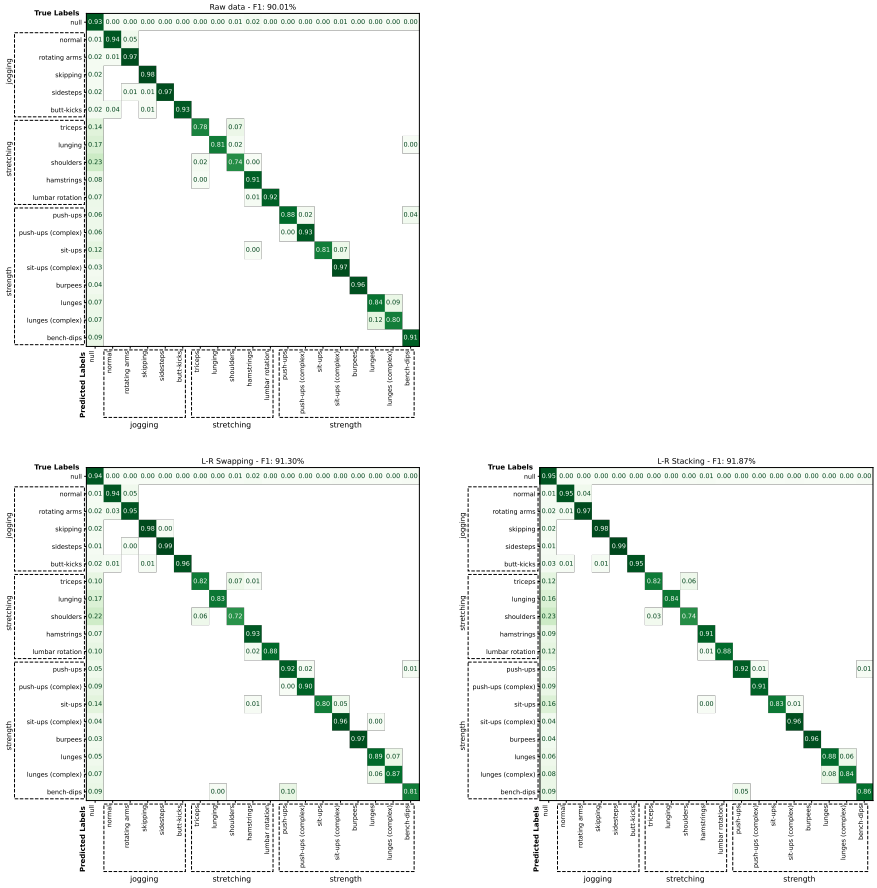


Figure 5.4: Normalized confusion matrix of the grouped 3-fold predictions (with temporal prediction smoothing).

which leverages the high temporal correlation of labels. As a final boost to our test set predictions, we performed *rule-based activity boosting*, which exploits the characteristics of the study protocol, making this boost not applicable in practice.

5.3.5.1 K-fold Majority Voting

Instead of training on the entire train dataset to make final predictions on the test data, we utilized K-fold cross-validation (CV). The predictions made after fitting each fold are aggregated by utilizing majority voting (MV) on the prediction probabilities. We deliberately opted for a 3-fold CV, as this results in only 50% data overlap across the training folds. Using this approach, we effectively perform bootstrapping through CV on the training data. When using an empirical leave-one-subject-out (LOSO) 3-Fold

CV approach, we observed that this majority voting aided in boosting the performance with 1% (absolute) on average.

5.3.5.2 Temporal Prediction Smoothing

We implemented prediction smoothing to leverage the high temporal correlation between consecutive feature vectors (and thus labels). This was done using a vectorized approach with a distribution function to weight the smoothing process. Specifically, for predictions with a 0.5s stride, we utilized a 10-10 receptive left-right field (i.e., 5s on each side) and a normal distribution function with $\sigma = 6$.

5.3.5.3 Rule-Based Activity Boosting

An exploratory analysis of the training set labels revealed that each of the 18 workout activities were present for at least 50 seconds for every participant recording. As such, we adopted this $>$50s presence rule as a post-hoc method to boost underrepresented classes in regions where other activities were selected (with low probabilities). Figure 5.5 demonstrates this boosting method applied to a user session from the test set.

5.4 Experimental Results and Discussion

In accordance with the WEAR dataset benchmarking [2], all experiments were conducted using grouped 3-fold cross-validation, with subjects being grouped per fold. Using the macro F1 metric, we analyze the predictions on the validation (out-of-fold) set and assess the impact of smoothing, indicated by respectively the $F1$ and $F1_{PP}$ columns in Table 5.2. The experiments were performed on a server computer (Arch Linux), with an AMD Ryzen 5 2600x CPU, 48GB of DDR4 RAM, and an Nvidia RTX 2070 GPU. The total execution time for all experiments, including preprocessing, feature extraction, data augmentation, and postprocessing, was less than 1 hour.

A first notable observation from Table 5.2 is that features derived from the `SMV` signal resulted in the poorest validation performance. Yet, `SMV` has been commonly

Table 5.2: Macro F1 scores of the investigated approaches.

	Val. score		**Data size**		
Approach	**F1**	**F1$_{PP}$**	**# feats**	**N$_{augm}$/N**	**rot. inv.**
UL-pairing	0.9154	0.9187	996	4	$\pm$
LR-swapping	0.9084	0.9130	1992	4	$\pm$
raw	0.8953	0.9001	1992	1	$\times$
rot_inv$_{stat3}$	0.8837	0.8904	1992	1	✓
rot_inv$_{sort}$	0.8815	0.8891	1992	1	✓
rot_inv$_{stat2}$	0.8806	0.8882	1328	1	✓
SMV	0.8055	0.8146	664	1	✓

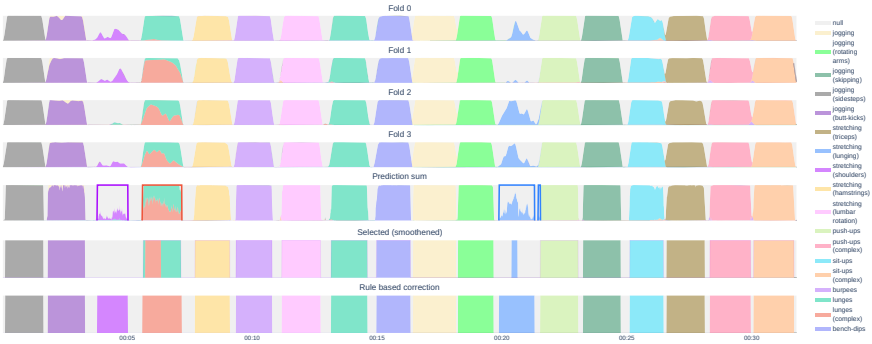


Figure 5.5: Stacked prediction plot of a test participant recording. The upper three subplots show the 0.5-second predictions for each of the three training folds. The fourth subplot displays the sum of these predictions, and the resulting k-fold majority-voted prediction is shown in the fifth subplot. The blue line in the "Prediction Sum" subplot illustrates our rule-based postprocessing, identifying regions where an underrepresented class (e.g., "stretching (lunging)") is present but not selected due to another class having a higher probability (e.g., "null"). The bottom subplot displays our final prediction, incorporating these rule-based postprocessing corrections applied to the smoothed majority-voted selection from the above plot.

employed in many wearable-based activity detection studies [13].

Second, our proposed rotation-invariant statistical aggregation approaches (i.e. $\texttt{rot\_inv}_{sort}$, $\texttt{rot\_inv}_{stat3}$, and $\texttt{rot\_inv}_{stat2}$) outperformed the `SMV` configuration by an absolute margin of $\pm 7.5\%$. Interestingly, rot_inv_{stat2}, which has a 1/3 compression ratio, showed only a minimal decrease in F1 score (absolute 0.1-0.3%) compared to the other two, non-compressed, approaches.

Third, the three rotation-invariant statistical aggregation techniques were surpassed by the `raw` approach by an absolute $\pm 1\%$. This suggests that rotation information, captured by the axial components in the raw features, is of large importance for this challenge. Although our exploratory data analysis revealed that some recordings/sessions involved wearables being swapped or worn in an opposite orientation, the majority of the data demonstrated a consistent orientation. We suspect that this consistency limited the performance improvement from rotation-invariant feature aggregation. This insight led us to designing the left-right swapping and upper-lower limb pairing techniques, which retain orientation information while augmenting the data to enhance model robustness.

Fourth, both proposed data augmentation techniques outperformed the `raw` configuration, with an absolute improvement of 1% for the LR-swapping and 1.5% for the UL-pairing. Notably, UL-pairing achieved higher performance while utilizing features from only two wearables (per feature vector). This observation hints towards substantial redundancy between left-right wearable data. Moreover, the UL-pairing technique also proves robust to missing wearable data, as long as data from one of the

left-right limb wearables is available.

Last, for all configurations, temporal prediction smoothing consistently improved performance, with absolute gains ranging from 0.3% to 0.9%.

Figure 5.4 complements Table 5.2 by showing the confusion matrices for our three best experiments. Note that these results can be positioned against the inertial baselines obtained by Bock et al. [2], since we used a similar grouped 3-fold CV procedure, the same stride (i.e., 0.5s), and temporal prediction smoothing. The highest macro F1 score from an inertial-only model in the WEAR dataset paper, achieved by the Attend and Discriminate (A- and D-) deep learning model architecture [14], was 83.08%. All our experiments, except for the SMV configuration, surpassed this performance by a substantial margin, demonstrating the expressiveness of a multi-resolution feature vector (using only limited, i.e., 14, feature functions).

Interestingly, all confusion matrices show limited confusion among activity classes but substantial confusion with the null class. This is especially pronounced for stretching-based activities, which have the greatest margin for improvement. Additionally, we observe a notable improvement in distinguishing between lunges and complex lunges, with a 5% absolute increase from the raw data to both data augmentation configurations.

5.4.1 Test Set Predictions

To generate predictions for the test set, we utilize the UL-pairing model along with all three postprocessing techniques detailed in Section 5.3.5. Specifically, the UL-pairing model combined with smoothing achieved a macro F1 score of 91.87% on the validation data (see Table 5.2 and Figure 5.4).

Figure 5.5 illustrates the three postprocessing steps (k-fold majority voting, temporal prediction smoothing, and rule-based prediction boosting) applied to a participant recording from the test set. Since we employ a 0.5-second stride, predictions were made in 0.5-second intervals. To comply with the competition's requirement for sample-wise predictions, we expanded these 0.5-second predictions to the nearest sample-wise timestamp. Empirical validation of this expansion on the 3-fold results showed a negligible performance decrease of 0.02% (resulting in an expanded macro F1 score of 91.85%), confirming that a 0.5-second stride is adequately fine-grained.

5.5 Conclusion

This chapter presents the findings and final approach of the "Signal Sleuths" team for the 2024 HASCA WEAR challenge. During exploratory data analysis, we identified inconsistencies in wearable orientation both within and across participant recordings, leading to the exploration of rotation-invariant techniques. We employed a fixed set of multi-resolution features and focused on a traditional machine learning pipeline, enabling

us to conduct an ablation study on novel rotation-invariant and multi-wearable data augmentation techniques. Specifically, we investigated: (i) relying solely on the SMV features, (ii) a novel approach that makes rotation-aware features rotation-invariant through statistical aggregation, and (iii) two data augmentation techniques: swapping left-right wearables to enhance model robustness (LR-swapping) and using only one upper and one lower limb wearable pair (UL-pairing). In addition to these techniques, we evaluated the impact of temporal prediction smoothing as a postprocessing step.

Our results indicated that solely relying on SMV features yields the poorest performance. Yet, this has been commonly used as the de facto technique in many works. Moreover, our proposed rotation-invariant statistical aggregation demonstrated a substantial improvement over SMV, but could not outperform the rotation-aware model (i.e. raw), underscoring the significance of rotation information in this dataset. Both LR-swapping and UL-pairing outperformed the raw model, with UL-pairing achieving the highest sample-wise macro F1 score of 91.85% (using temporal prediction smoothing). Given that UL-pairing uses data from only one upper and one lower limb wearable, we hypothesize a high redundancy in left-right wearable data. Furthermore, we assessed the impact of temporal prediction smoothing, which consistently improved performance by an absolute 0.3% to 0.9%.

Comparing our results with the benchmark approaches from the WEAR dataset paper, we demonstrated the competitiveness of a multi-resolution feature vector combined with a traditional machine-learning pipeline.

In conclusion, through our proposed multi-wearable data augmentation and several post-processing steps, we designed more robust models for multi-wearable workout activity detection.

References

[1] Jeroen Van Der Donckt, Jonas Van Der Donckt, Emiel Deprost, Nicolas Vandenbussche, Michael Rademaker, Gilles Vandewiele, and Sofie Van Hoecke. "Do not sleep on traditional machine learning: Simple and interpretable techniques are competitive to deep learning for sleep scoring". In: *Biomedical Signal Processing and Control* 81 (2023), p. 104429.

[2] Marius Bock, Hilde Kuehne, Kristof Van Laerhoven, and Michael Moeller. "Wear: An outdoor sports dataset for wearable and egocentric activity recognition". In: *arXiv preprint arXiv:2304.05088* (2023).

[3] Subhas Chandra Mukhopadhyay. "Wearable Sensors for Human Activity Monitoring: A Review". In: *IEEE Sensors Journal* 15.3 (2015), pp. 1321–1330. DOI: 10.1109/JSEN.2014.2370945.

[4] Marija Stojchevska, Mathias De Brouwer, Martijn Courteaux, Bram Steenwinckel, Sofie Van Hoecke, and Femke Ongenae. "Unlocking the potential of smartphone and ambient sensors for ADL detection". In: *Scientific Reports* 14.1 (2024).

[5] Jonas Van Der Donckt, Mathias De Brouwer, Pieter Moens, Marija Stojchevska, Bram Steenwinckel, Stef Pletinck, Nicolas Vandenbussche, Annelis Goris, Koen Paemeleire, Femke Ongenae, et al. "From self-reporting to monitoring for improved migraine management". In: *Engineer meets Physician (EmP)*. 2022.

[6] Zhixian Yan, Vigneshwaran Subbaraju, Dipanjan Chakraborty, Archan Misra, and Karl Aberer. "Energy-efficient continuous activity recognition on mobile phones: An activity-adaptive approach". In: *2012 16th international symposium on wearable computers*. Ieee. 2012, pp. 17–24.

[7] Espruino. *Bangle.js - Espruino*. 2023. URL: https://www.espruino.com/Bangle.js (visited on 07/01/2024).

[8] Jonas Van Der Donckt, Jeroen Van Der Donckt, Emiel Deprost, and Sofie Van Hoecke. "tsflex: Flexible time series processing & feature extraction". en. In: *SoftwareX* 17 (Jan. 2022), p. 100971. ISSN: 23527110. DOI: 10.1016/j.softx.2021.100971. URL: https://linkinghub.elsevier.com/retrieve/pii/S2352711021001904 (visited on 03/03/2022).

[9] Vallat Rafael. *Antropy: time-efficient algorithms for computing the complexity of time-series.* https://github.com/raphaelvallat/antropy. 2018.

[10] Charles R. Harris et al. "Array programming with NumPy". en. In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/s41586-020-2649-2. URL: https://www.nature.com/articles/s41586-020-2649-2 (visited on 06/16/2023).

[11] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. "SciPy 1.0: fundamental algorithms for scientific computing in Python". In: *Nature methods* 17.3 (2020), pp. 261–272.

[12] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. "CatBoost: unbiased boosting with categorical features". In: *Advances in neural information processing systems* 31 (2018).

[13] Lin Wang, Hristijan Gjoreski, Mathias Ciliberto, Sami Mekki, Stefan Valentin, and Daniel Roggen. "Benchmarking the SHL recognition challenge with classical and deep-learning pipelines". In: *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*. 2018, pp. 1626–1635.

[14] Alireza Abedin, Mahsa Ehsanpour, Qinfeng Shi, Hamid Rezatofighi, and Damith C Ranasinghe. "Attend and discriminate: Beyond the state-of-the-art for human activity recognition using wearable sensors". In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5.1 (2021), pp. 1–22.

6

Mitigating Data Quality Challenges in Ambulatory Wrist-worn Wearable Monitoring

Section 1.3 of Chapter 1 highlighted the potential of wearables for real-world monitoring studies. While these devices enable continuous, objective data collection, their use in everyday environments presents significant data quality challenges that can undermine the reliability of downstream analyses. Moreover, when smartphone applications are used to enrich wearable data, additional complexities arise, including user compliance issues and data entry errors.

This chapter addresses these data quality challenges by proposing actionable countermeasures, with a particular focus on the role of data visualization in identifying and mitigating issues, contributing towards **RG3-A**. Using two real-world datasets (mBrain21 and ETRI Lifelog 2020), we materialize solutions designed to improve data integrity and usability.

My contributions can be summarized as follows:

- Identifying key challenges in real-world monitoring studies, including those involving ecological momentary assessment (EMA) labels.
- Providing practical countermeasures, supported by visualizations and code examples.
- Demonstrating the application of these countermeasures on two datasets where applicable.

- Enhancing transparency and reproducibility by open-sourcing the qualitative codebase and making a portion of the mBrain21 dataset publicly available.

Mitigating Data Quality Challenges in Ambulatory Wrist-worn Wearable Monitoring Through Analytical and Practical Approaches

Jonas Van Der Donckt, Nicolas Vandenbussche, Jeroen Van Der Donckt, Stephanie Chen, Marija Stojchevska, Mathias De Brouwer, Bram Steenwinckel, Koen Paemeleire, Femke Ongenae, and Sofie Van Hoecke

Published in "Nature Scientific Reports, Vol. 14, 2024, Article number 17545"

Abstract Chronic disease management and follow-up are vital for realizing sustained patient well-being and optimal health outcomes. Recent advancements in wearable technologies, particularly wrist-worn devices, offer promising solutions for longitudinal patient monitoring, replacing subjective, intermittent self-reporting with objective, continuous monitoring. However, collecting and analyzing data from wearables presents several challenges, such as data entry errors, non-wear periods, missing data, and wearable artifacts. In this work, we explore these data analysis challenges using two real-world datasets (mBrain21 and ETRI lifelog2020). We introduce practical countermeasures, including participant compliance visualizations, interaction-triggered questionnaires to assess personal bias, and an optimized pipeline for detecting non-wear periods. Additionally, we propose a visualization-oriented approach to validate processing pipelines using scalable tools such as `tsflex` and Plotly-Resampler. Lastly, we present a bootstrapping methodology to evaluate the variability of wearable-derived features in the presence of partially missing data segments. Prioritizing transparency and reproducibility, we provide open access to our detailed code examples, facilitating adaptation in future wearable research. In conclusion, our contributions provide actionable approaches for improving wearable data collection and analysis.

6.1 Introduction

In recent years, wearable sensing has seen a vast increase in both research and commercialization, driven by the reduced networking and hardware costs as well as the non-intrusive nature of these devices [1]. Wearable technologies offer promising solutions for patient monitoring by continuously acquiring objective physiological data unobtrusively and at scale. As such, wearable sensing could potentially ease the strain on the healthcare system, particularly in managing chronic diseases [2, 3]. For instance, diabetes patients could benefit from real-time tracking of blood sugar levels through wearable sensing and timely intervention [4]. Similarly, patients with cardiovascular conditions might use wearable sensors to monitor vital signs, providing early detection of anomalies and enabling prompt medical attention [5].

To effectively implement remote monitoring, it is essential to integrate data en-

tries from patients and/or healthcare providers with data from wearable sensors in ambulatory settings [6]. This integration necessitates evaluating the wearable's ability to detect specific events, such as fall detection for the elderly [7], or identifying and validating biomarkers in real-life settings [8]. Remote monitoring has proven valuable in tracking and analyzing chronic events in certain populations, such as headache attacks of migraine patients [9, 10, 11], seizures in epilepsy patients [12], or depressive episodes [1].

Given the potential of remote monitoring, there are an increasing number of studies that collect wearable data along with acute event data in ambulatory settings. However, significant challenges arise when analyzing wearable data in real-world scenarios [13, 14]. These analysis challenges stem from issues related to data quality, which occur throughout participant acquisition, data collection, and retrospective analysis. They encompass problems related to participants, monitoring devices, logging applications, and technologies [15, 16]. Many studies in ambulatory wearable monitoring currently either overlook or sidestep these challenges [10, 11, 17]. This chapter aims to address this gap by offering practical and actionable countermeasures to the below-identified data quality challenges.

Specifically, we categorize the data quality challenges into two groups: those related to (i) participants and monitoring applications, and those related to (ii) wearables in real-world settings, thereby excluding other domain challenges such as technology, system architecture, and scalability [16, 18]. The (i) participant and monitoring application category includes problems with data entry and quality, lack of participant compliance and motivation, unverified assumptions, and personal biases. The (ii) wearable category focuses on issues like non-wear periods, wearable artifacts, and missing wearable data.

To address (i) participant and application-related challenges, we introduce a novel participant compliance visualization technique to monitor participant motivation in near-real-time. In addition, we propose interaction-triggered questionnaires to reduce and filter data entry errors. For the (ii) wearable analysis challenges, we present an efficient and performant non-wear detection pipeline for processing wearable data at larger scales. Additionally, we propose a generic, visualization-oriented approach to validate signal processing pipelines. Lastly, we outline a bootstrapping technique to assess the variability of wearable-derived features on partially missing data segments.

To elucidate these challenges, we draw upon our first-hand experience during the mBrain study [9]. By utilizing an excerpt of this mBrain21 dataset along with the ETRI lifelog 2020 dataset [19], we substantiate our proposed countermeasures with reproducible implementations.As such, this work aims to aid future monitoring studies in bridging the gap between recognizing the occurrence of the identified challenges and the practical applicability of countermeasures.

6.2 Related Work

Over the past decade, research interest in ambulatory wearable-based monitoring studies has significantly increased, leading to multiple works indicating challenges and limitations inherent to such studies. In this section, we outline works that consider these challenges.

Schmidt et al. (2018) [20] provided guidelines and practical implementation details from their field study to enhance the accuracy of manual data entries for ecological momentary assessment (EMA) in ambulatory wearable monitoring. They emphasized brevity in EMA duration, targeting core study goals, and daily screenings of wearable signal modalities to routinely assess data quality. This is crucial to perform timely re-instructions to participants when a decline in data quality becomes apparent. Furthermore, they suggested configuring EMA applications to match participants' circadian rhythm and proposed incremental reward systems to sustain participant engagement. However, their work lacked practical examples of participant interventions and did not focus on providing methodologies to leverage collected ambulatory wearable data, along with EMA events, in downstream analysis.

In 2019, Schmidt et al. [13], expanded on these guidelines by integrating wearable data processing, but did not specifically put this in the context of the other challenges associated with field studies.

Balbim et al. (2021) [21] discussed data quality challenges associated with Fitbit physical activity (PA) trackers, focusing on study preparation, intervention delivery, and study closeout. However, they did not address data analysis, leaving a gap in methodologies for handling data post-collection.

Cho et al. (2021) [16] conducted a systematic review, identifying three overarching factors influencing wearable data quality: device- and technical-related, user-related, and data governance-related factors. Device- and technical-related factors include hardware issues such as sensor degradation or malfunction, software issues related to the quality of proprietary wearable algorithms, and networking issues such as data loss during transmission and recording. User-related factors involve non-wear periods and user errors stemming from wearable misplacement or poor skin contact. Lastly, data governance-related factors arise from the lack of standardization, inconsistency in algorithms across different devices, and sensor placement variations. While their review concentrated on elucidating these factors, it did not provide methodologies to address them.

Similarly, Sriram et al. (2009) [18] identified three highly comparable factors that affect data quality; sensor or device-related, human or user-related, and system architecture factors.

Aligning with both taxonomies, this work focuses on challenges related to user-related and device-related factors, thereby omitting technical and data governance challenges.

In recent work, Böttcher et al. [22] assessed the data quality of the wrist-worn Empatica E4 wearable, particularly in the context of epilepsy monitoring, using multiple datasets from hospitalized and ambulatory care settings. They evaluated data quality through computing signal quality scores for several physiological signal modalities of the Empatica E4 and computed a wearable-on-body score, along with a data completeness score, representing the ratio between the actual recorded volume and expected data volume. Their findings suggested superior data quality and completeness during nighttime (20h-8h). Notably, wearable streaming revealed a higher data loss compared to on-device logging. However, their work did not provide actionable methodologies for improving data quality or conducting analysis during study collection. Additionally, while they shared analysis results publicly (GitHub), documentation was minimal, and source data was not shared due to data-sharing agreements.

In conclusion, while considerable research has highlighted data quality challenges prevalent in wearable monitoring studies, there remains a significant gap in providing actionable countermeasures, tangible examples, and streamlined code for addressing post hoc analysis issues. Furthermore, few studies offer access to their data or code, complicating the assessment of their methodologies' broader applicability. By addressing these gaps, our work aims to provide practical solutions and reproducible methods to improve user and wearable-related data quality and analysis in ambulatory wearable-based monitoring studies.

6.3 Methodology

This section outlines our approach to tackling data quality challenges. First, we introduce two distinct datasets employed to demonstrate these challenges and highlight their characteristics. Next, we define the scope of our work, distilling the specific data quality challenges we aim to address. Finally, we describe the programming environment and tools chosen to tackle these challenges.

6.3.1 Datasets

We materialize data quality challenges by using examples from the mBrain21 and ETRI lifeLog 2020 datasets, whose characteristics are outlined in Table 6.1.

The second wave of the mBrain study, i.e., mBrain21, monitored 30 patients diagnosed with chronic headache disorders over 90 days. Data from four participants who consented to public distribution were used in this work. Monitoring involved smartphone sensors (i.e., movement, application usage), the Empatica E4 wrist-worn wearable, and a dedicated logging application to record headache events, medication intake, and daily questionnaire responses [23]. mBrain21 participants were instructed to wear the Empatica device for at least 8 hours per day. The Empatica E4 streamed data to the logging application, which sent it to internal servers after a two-minute buffer.

Table 6.1: Comparative overview of the characteristics of the two datasets selected for this work.

	mBrain21*	**ETRI lifelog 2020**
Subjects	4*	22
Country	Belgium	South Korea
Age range (median, 95% CI)	/	8 [21, 33]
Sex (% female)	/	41%
Study duration	90 days	28 days
Wearable type	Empatica E4	Empatica E4
Wearable placement	Wrist	Wrist
Recording mode	streaming	device
Labeling	self-report	self-report

*This dataset only provides an excerpt of 4 participants out of the 30 participants from the second wave of the mBrain study due to informed consent availability reasons

This near real-time wearable data stream was utilized to construct automatic timelines of activity and stress predictions, as shown in Figure 6.1. The primary objective of the mBrain study was to analyze ambulatory wearable data in relation to headache intervals. For instance, wearable movement data was utilized to investigate changes in movement behavior during cluster headache, as demonstrated in Vandenbussche et al. (2023) [24].

The ETRI lifelog 2020 study monitored 22 participants for 28 days to acquire data-driven descriptions of human life from various perspectives [19]. Specifically, the ETRI dataset is composed of a smartphone, the Empatica E4 wearable, and Withings sleep-quality monitoring mat. Participants utilized a dedicated logging application to self-report their activity, social state (alone, with someone, with a group), semantic location (e.g., home, work), and emotional state (valence, arousal). These self-reported labels were presented to the user via a timeline. Unlike the streaming approach in mBrain, participants in the ETRI study were tasked with offloading the on-device logged Empatica data to a computer, which then uploaded it to Empatica's cloud.

Both studies under consideration utilized the Empatica E4, a medically graded wristband that captures physiological and movement data. The E4 contains a three-axis accelerometer that samples at 32 Hz with a range of +/-2 g. The 64 Hz blood volume pulse (BVP) signal is constructed from a proprietary on-device algorithm that leverages the green and red exposure photoplethysmography (PPG) signals [25]. This derived BVP signal serves as input for proprietary algorithms that compute the interbeat interval (IBI) timings and the mean heart rate (HR). The skin surface temperature (TEMP) is acquired at 4 Hz via a thermopile sensor. Lastly, the skin conductance or electrodermal activity (EDA) is acquired at 4 Hz via two AgCl electrodes.

We deliberately selected these two datasets given our direct experience with the mBrain dataset and the well-documented nature and availability of the ETRI lifelog 2020 dataset. Since both datasets are recorded by different research institutes and

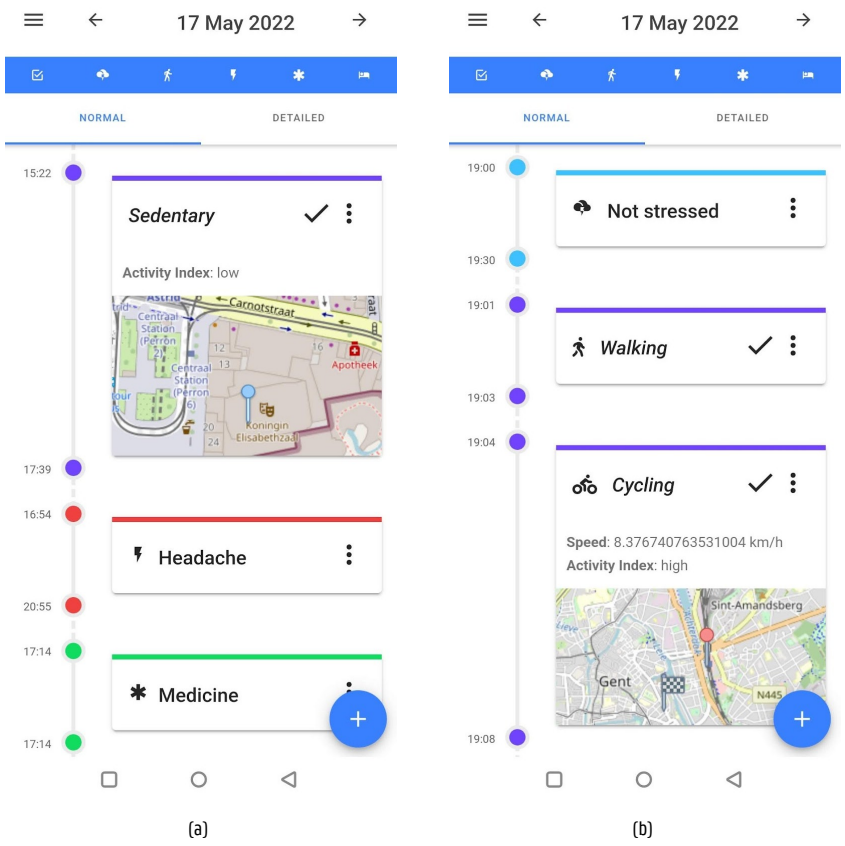


Figure 6.1: mBrain application timeline of a dummy participant, showcasing contextual data, including user-defined semantic locations (e.g. "Work").

capture different demographic populations, we believe that they should demonstrate a certain genericity of our presented methodologies. As we were not involved with the ETRI lifelog's data collection, we rely on examples from the mBrain study to illustrate countermeasures for participant data entry challenges.

6.3.2 Selecting Data Quality Challenges

This work focuses on challenges related to data completeness and correctness in ambulatory monitoring studies, specifically those using wrist-worn devices along with an application for ambulatory label acquisition. Our emphasis stems from first-hand experience with the mBrain project, which fits this study type. Moreover, such studies are frequently employed by smaller-scale research to assess the wearables' potential of detecting these ambulatory labeled events of interest, such as affect, headaches, and

stress [10, 11, 13].

We categorize the data quality challenges into two domains, which occur within the taxonomies Cho et al. [16] and Sriram et al. [18]: (1) participant data entry challenges, and (2) wearable analysis challenges. The first category, participant data entry challenges, encompasses participant and application-oriented challenges that impact data quality. These include participant compliance and motivation (Challenge 1; C1), implicitness assumptions (C2), data entry errors (C3), and personal bias (C4). Conversely, the wearable data challenges concentrate on wearable-related analysis challenges, including wearable non-wear (C5), wearable artifacts (C6), and the analysis of "windows-of-interest" with missing or anomalous data (C7). For each of these seven identified challenges, we provide detailed insights into their causes, impacts, and potential countermeasures. Wherever possible, we illustrate these countermeasures with concrete visualizations and implementation examples, applicable to either the data analysis side (retrospective) or the application side (prospective or reactive).

6.3.3 Programming Environments

Longitudinal wearable monitoring studies produce large datasets. As indicated by the Kaggle 2022 data-science survey, notebook-based environments, particularly those using IPython, are the go-to tools for data scientists [26]. Interactive notebook-based formats drive data exploration, which is crucial in every step of the data science process [27]. Consequently, this study employs IPython environments to illustrate methods for tackling data-centric challenges. The Python packages utilized in this work are listed and managed using the Poetry Python package manager [28].

Notably, both Weed et al. (2022) [29] and Böttcher et al. (2022) [22] utilized MATLAB to perform their wearable analyses. However, we believe that Python's scalability, open-source nature, larger community, and easier integration with other technologies, along with cost-effectiveness and flexibility, is a more suitable choice.

6.4 Participant Data Entry Challenges

In this section, human- and application-related data entry challenges of wearable monitoring studies are covered.

6.4.1 Challenge 1: Participant Compliance

Participant motivation to wear a wearable device, interact with questionnaires, and submit events of interest is crucial for obtaining a qualitative dataset. However, as seen in numerous cases, including our own mBrain study [9], motivation and study interaction tend to decrease over time [20, 30]. This decline can be attributed to several factors, including inconvenient wearable connection processes, wearable design

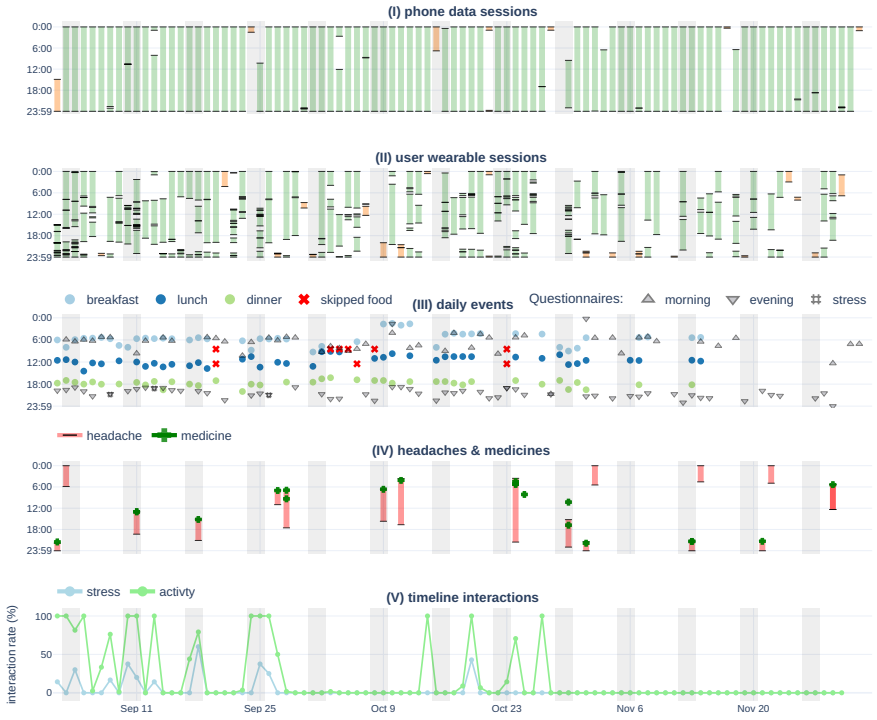


Figure 6.2: mBrain study interaction visualization of a single participant for a period of 90 days. Note. The figure consists of several subplots with a shared x-axis, each providing different layers of information about the participant's activity and interactions. Subplots (i) and (ii) display phone and wearable data sessions over time, with each bar on the x-axis representing a unique day. For the first four plots, the y-axis indicates the time of day, revealing patterns of data fragmentation and daily volumes over extended periods. Gray-shaded areas indicate weekends. The mBrain study requires a minimum of eight hours of wearable data daily. This compliance is color-coded in the first two subplots: green represents days with more than 8 hours, while orange indicates less than 8 hours. The daily events subplot (iii) provides an overview of food intake and questionnaire interactions. Subplot (iv) provides a visual record of the participant's headaches and medication intake. The final subplot (v) shows the interaction rate (%) on the y-axis, illustrating the frequency of participant interactions with stress and activity timeline events derived from the wearable data stream.

aesthetics, frequent or poorly timed EMA collection, and a negative user experience with the study (including adverse reactions to the wearable device).

To minimize user burden, each labor-intensive component, including wearable use and EMAs, should target the core goal of the research, with minimal overhead [20, 31]. Frequent or long-term manual data entry can lead to response fatigue, reducing data quality and accuracy [32].

Maintaining participant commitment throughout the study is essential for high-

quality data and labels, both in frequency and completeness [20]. Automated input processing, through digitally acquiring user data via structured forms on smartphones is a valid alternative to traditional double manual data entry [33]. Additionally, strategies such as incremental rewards [34, 35], gamification [36, 37], and periodic contact with participants [38] can help sustain motivation. Lastly, querying participants about their experiences, feelings, and gains from participating at the study's conclusion can provide valuable insights into maintaining long-term engagement.

6.4.1.1 Countermeasures

From a data science perspective, several solutions can be devised to address these challenges. For instance, compliance-based visualizations have been utilized to assess participant interactions during ambulatory monitoring studies [20].

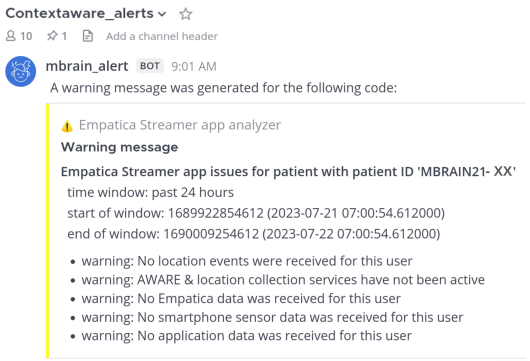


Figure 6.3: Example of an mBrain alert message, shown to the study coordinators when no wearable data is received from a participant.

During the mBrain study, we developed a methodology to evaluate participant motivation continuity by (i) automatically generating interaction rate reports from incoming data streams, and (ii) incorporating real-time notification via webhooks to alert study coordinators when participants fall below interaction thresholds. Figures 6.2 and 6.3 illustrate this methodology applied to the mBrain study. Specifically, scheduled jobs were utilized to generate these daily participant compliance reports, which, similar to visualizations from Rawassizadeh et al. [39], displayed intervals for which there is wearable and phone data, and the evolution of answered questionnaires. This visualization provides an overview of participant compliance to study coordinators, facilitating timely re-instruction when needed. To showcase the generalizability of our proposed method, we applied this compliance visualization to the ETRI lifelog 2020 dataset (Supplemental Figure 2).

While monitoring platforms like Empatica Care lab and REDCap [40] offer similar features, remote health monitoring often involves various data types that cannot be

stored in a single database platform (e.g., high-frequency wearable data versus survey responses). This necessitates custom solutions for a comprehensive compliance overview.

In the mBrain study, previous-day compliance data was utilized to produce daily alerts, as shown in Figure 6.3, notifying study coordinators when participants logged less than 8 hours of data. Based on these compliance assessments (i.e., reports and alert messages) coordinators could decide whether to contact participants to understand their reasons for reduced compliance. Within the mBrain study, the clinician-neurologist sent—whenever needed, and with caution not to overload participants as well—personalized messages to the participants via a dedicated interface. This methodology, empirically evaluated during the mBrain study, allowed physicians to track participant interactions and enhance study compliance through a built-in monitoring system with rapid notification capabilities.

6.4.2 Challenge 2: Implicitness Assumptions

A common assumption in data collection is that the absence of a recorded event implies it did not occur [9, 41]. For instance, if no medication event is logged for a day, it might be assumed that the patient did not take any medication, although the participant may have simply forgotten to log it. Therefore, it is essential to verify these implicitness assumptions through direct questioning about these (absent) self-reported events. Given the straightforwardness of this approach, it is possible that no prior studies have explicitly reported using these checks.

6.4.2.1 Countermeasures

In the mBrain study, morning questionnaire responses serve to validate or disprove assumptions regarding headaches and medication use from the previous day, as shown in Figure 6.4 (a). Subplot (b) of Figure 6.4 displays notifications sent to the user when their responses contradict these assumptions, such as replying "No" to one of the questions in (a). While these additional queries improve data accuracy, they also increase the burden on participants. Therefore, these checks should be confined to parameters crucial to the study's analysis. Notably, regularity in questionnaires may improve study compliance by creating a routine for the participant to interact with the study environment, and therefore possibly nudge the participant toward other components of the study [42].

6.4.3 Challenge 3: Data Entry Errors

Data entry errors often arise from accidental user mistakes during interactions, primarily due to suboptimal design choices [43]. As such, enhancing user experience through cognitively ergonomic designs can significantly reduce these human errors [44]. A proactive strategy involves conducting a pilot phase with a small group of participants

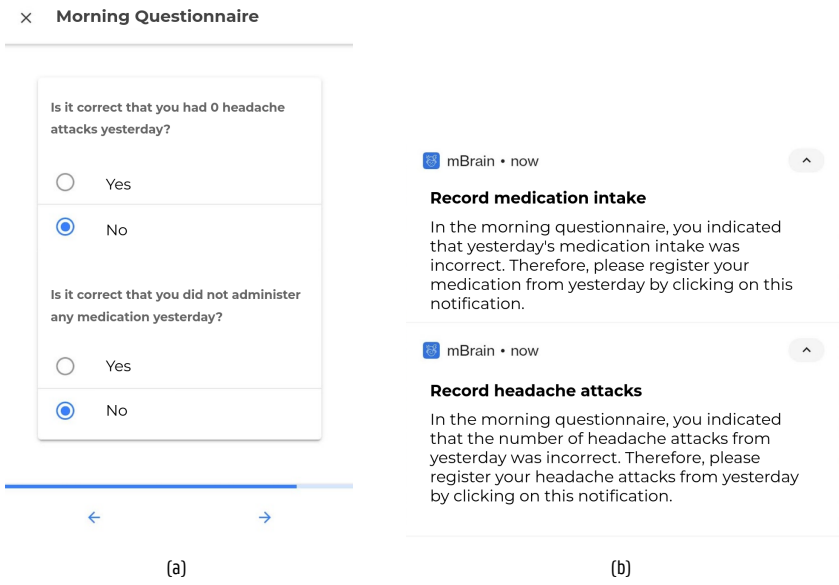


Figure 6.4: (a) Screenshot of questions in the mBrain study's morning questionnaire evaluating implicitness for headache and medication events. (b) Notifications are activated based on responses to the implicitness questions.

and analysts to identify and correct issues related to implicitness assumptions and data entry before full-scale monitoring [21].

Temporal inaccuracies, a specific category of data entry errors, typically result from users' uncertainty in allocating exact timestamps to events [20]. These inaccuracies are influenced by recall bias (misdating past events) and predictive bias (misdating future events). Notably, EMAs emerged as a strategy to evaluate immediate experiences in participants' everyday settings, thereby minimizing recall bias [45]. Moreover, temporal accuracy can be enhanced by integrating contextual data, such as location and activities, into an automated timeline to counteract recall bias [46].

6.4.3.1 Countermeasures

Within the mBrain study, we conducted two pilot phases to factor out data entry errors and assess the robustness of our infrastructure in managing higher user loads [47]. We also propose using an extensive intake procedure, where participants review all components of the logging application and the wearable device with the study coordinator. This does not only clarify the process but also benefits the participant's motivation to perform data entries. Providing a detailed manual during the intake, in which all the intricacies of the application and study procedure are described, further supports this goal [9]. This intake procedure was successfully validated within the mBrain study.

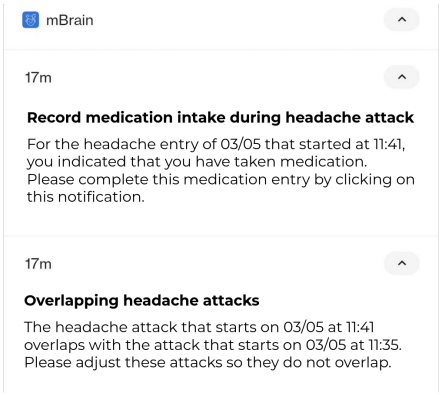


Figure 6.5: Example mBrain application notifications when conflicting data entries were made by a participant.

A reactive measure implemented based on pilot study errors includes sanity checks to reduce data entry errors. This system prevents logging multiple concurrent events of the same type and generates alerts for entries with improbable dates, such as logging a headache that occurred two weeks in the past or is set for a future date. Figure 6.5 showcases notifications in the mBrain study that are triggered by users' incomplete or ambiguous data entries. Remark that these notifications may increase user burden.

To tackle temporal inaccuracies, we propose using an automated user timeline, as employed within the mBrain study, sourced from smartphone and wearable data. This timeline, as shown in Figure 6.1, improves the temporal specificity when pinpointing stress or headache occurrences. Another countermeasure is allowing participants to specify a time range instead of a single, definitive timestamp. This approach recognizes and accommodates the user's temporal uncertainty, such as by letting them denote a span for both the start and end time of a headache event. However, this flexibility might complicate the user experience and eventual analysis, so it should be carefully aligned with the study's objectives to ensure its analytical value.

6.4.4 Challenge 4: Personal Bias

In remote health monitoring studies, where participant self-reporting plays a central role, personal bias emerges as a substantial challenge. This bias arises from the subjective nature of self-reported data and the various ways in which individual perceptions, beliefs, and motivations influence these reports [20]. Recognizing and correcting for personal biases enhances the overall validity of the study. It ensures that the conclusions drawn are genuinely reflective of the observed phenomena, not skewed by individual user tendencies or perceptions.

Addressing personal bias necessitates a multifaceted approach, including rigorous study design, participant education (to, e.g., homogenize definitions of concepts like

stress), regular reminders, intuitive technology interfaces, and integrating objective monitoring tools. Researchers should always consider personal biases when interpreting subjectively labeled data. Modeling participants as latent factors during analysis, by for instance modeling the participant as a random effect with a linear mixed (effect) model (LMM), is a recommended practice [48]. Including the participant as a random effect allows for the modeling of the variability between participants and helps in accounting for the within-subject correlation due to repeated measurements on the same participant. This way, any inherent individual bias or subject-specific characteristics (like baseline levels) that might affect the outcome variable can be taken into consideration during analysis.

Beyond self-reporting, personal bias can also manifest in device wear behavior. For instance, during the mBrain study, we noted that participants wore the wearable devices less frequently during headache episodes (Figure 6.7). Some participants might also avoid wearing devices during more intensive activities, potentially skewing the findings. A general countermeasure to this challenge is to instruct participants that the monitoring study aims to observe each aspect of their daily life and that they therefore

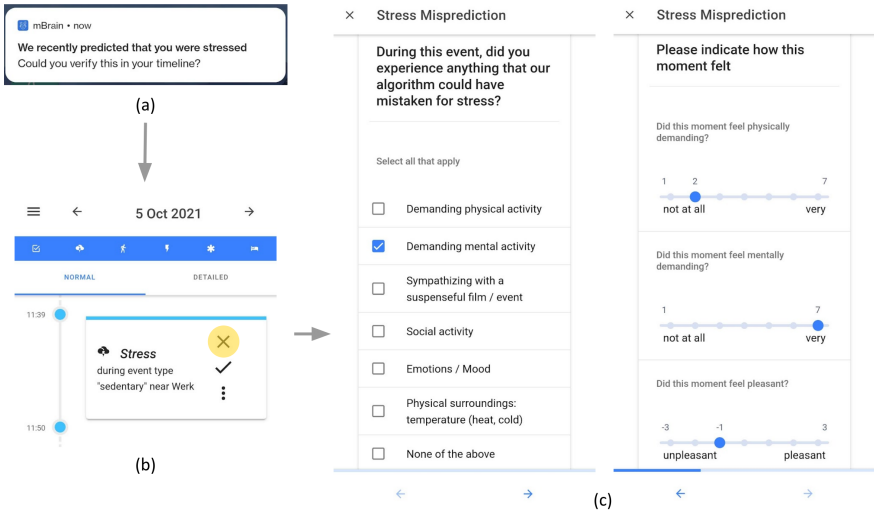


Figure 6.6: mBrain stress event interaction and its corresponding misprediction questionnaire. *Note*: When a stress-system-activation (e.g., a sudden non-activity induced increase in skin conductance responses) is detected in the streamed wearable data, a notification is sent to the user as shown in (a). This notification aids in reducing the interaction latency of the participant. When clicking on this notification, the participant is guided toward the mBrain timeline in which the recent stress event is shown, as depicted by (b). The yellow circle indicates that the participant re-labeled the stress-event period to be non-stressful. This, in turn, prompts the participant whether they have time to fill in a questionnaire that gauges for more contextual information about this event. This questionnaire is portrayed in (c) and indicates that the user was performing a demanding mental activity that was not perceived as really pleasant, possibly explaining the stress response.

must keep wearing the wearable whenever possible. Moreover, when such behavior is observed through compliance reports, participants can also be contacted during the study period (see C1).

Finally, the Hawthorne effect, i.e., the modification of participant behavior in response to being observed, can also affect data representativity [49]. It has been demonstrated that the Hawthorne effect appears to last for a limited time [50]. Hence, monitoring studies that have an adequate duration (i.e., up to 3-6 months) can mitigate this issue.

Figure 6.7: mBrain study wearable wear behavior overview of a single participant. The upper subplot illustrates the available wearable sessions, using similar bar intervals as Figure 6.2, providing an overview of wearable usage. In this subplot, weekends are marked in gray, and headache intervals in red. This participant has an average wearable data ratio of 44%, whereas the available data ratio during headaches is 39%. The lower left subplot depicts the average data ratio for the time of day throughout the study period. This subgraph reveals a notable decline in wearable use between 17h30 and 22h30. Conversely, the lower right subplot utilizes a heatmap to display the average data ratio against the time of day, distributed over each day of the week, highlighting discernible patterns in wear frequency. This heatmap elucidates that this specific participant has a tendency for reduced wearable use on Fridays and Saturdays, while Wednesdays exhibit the most wearable use. Remark how the reduced wearable usage during the evening period, shown by the through in the lower left subplot, is also discernable in this heatmap visualization.

6.4.4.1 Countermeasures

We propose using interaction-triggered questionnaires to gauge causes and contextual information related to the accuracy of highly personal events such as stress. Figure 6.6 (c) depicts such a questionnaire from the mBrain study, demonstrating how—although the participant disproved the stress prediction—the event was perceived as mentally demanding and slightly unpleasant, which might be indicated as stress by other users.

Moreover, participant interaction visualizations, as illustrated by Figure 6.7 for the mBrain study, allow coordinators to observe whether trends in non-wear or non-activity (e.g., no wearable data during evening periods). Insights from these visualizations can be utilized to send custom messages to participants to gauge their behavior. By assessing this personal bias, researchers can better understand how individual differences in lifestyle, behavior, and interaction with the device might influence the data collected. This leads to a more accurate interpretation of the health metrics derived from wearable devices.

6.4.5 Summary

To conclude this participant data entry section, Table 6.2 summarizes the four identified participant- and data-entry-related challenges and their corresponding countermeasures.

Table 6.2: Summary of the participant data entry challenges and their countermeasures.

Challenge	Countermeasures / Actions
Participant Compliance	
High user burden & response fatigue	App interactions: Minimize overhead; each component should target the core goal Wearable: Strive for a convenient wearable experience (e.g., connection process, battery life, …) *(reactive) Monitor participant compliance (+ reinstruct when needed)*
Decline in motivation	*(proactive) Periodic contact with study coordinator* *(proactive) Incremental reward systems & gamification* *(retrospective) Query experiences during takeout to pinpoint motivation hurdles*
Implicitness Assumptions	
Event absence assumption	(proactive) Utilize daily questionnaires to validate these assumptions (should always relate to core goal)
Data Entry Errors	
Application Entry Errors	*(proactive) Pilot phase to factor out errors* *(reactive) Sanity checks & notifications*
Temporal inaccuracies	*(proactive) Providing contextual data reduces recall bias* *(proactive) Gauge for temporal certainty*
Personal Bias	
Labeling bias	*(reactive) Gauge for contextual information / reasoning* *(retrospective) Include participant effect during analysis*
Wear behavior	*(proactive) Instruct participants to wear the device all the time* *(reactive) Monitor wear behavior and interfere when needed*
Hawthorne effect	Monitor for a sufficient duration (e.g., 3-6 months)

6.5 Wearable Analysis Challenges

This section addresses challenges related to the quality of ambulatory wearable data in the context of performing data analysis. The objective of such data analysis is primarily to examine "windows of interest", which can include event-related time spans (such as headache periods) or specific intervals of the day (like nighttime). We identify three key challenges: 1) data streaming when the wearable is not worn (C5), 2) artifacts introduced by the wearable device (C6), and 3) strategies for analyzing wearable data that include missing or spurious data (C7), arising from scenarios like non-wear and device-generated artifacts.

6.5.1 Challenge 5: Wearable not on Body

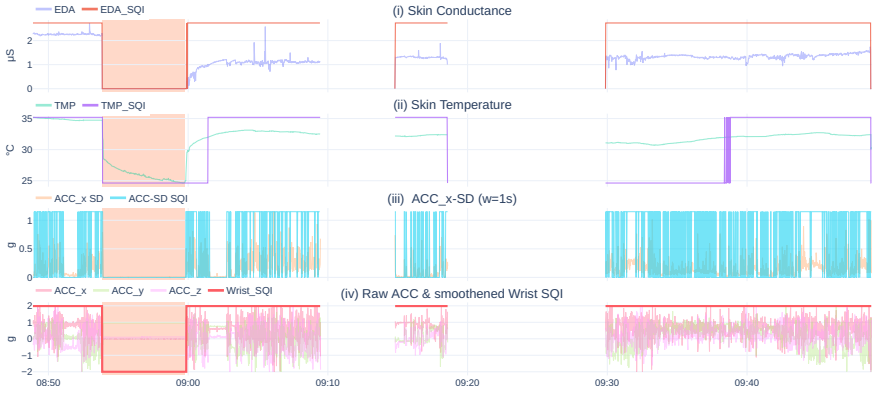
Non-wear periods, where the wearable records data despite not being worn, have long been acknowledged as a crucial challenge in actigraphy research [51, 52]. To address this, a variety of approaches have been established, which utilize wearable movement (ACC) signals for detecting non-wear. This detection is often performed as a preprocessing step, filtering the data before further analyses. Ahmadi et al. (2020) [53] evaluated five non-wear detection algorithms using only wrist-worn accelerometer data, finding that standard deviation-based algorithms effectively detect non-wear periods that last at least 30 minutes. However, this granularity may be insufficient when aiming to analyze specific time-located events.

Recently, there has been an increase in the development of non-wear detection algorithms that incorporate physiological parameters, such as skin temperature and skin conductance, in addition to wearable movement. Vert et al. (2022) [54] utilized the GENEActiv wrist-worn wearable, which includes a near-body temperature sensor along with a light sensor. By utilizing the rate-of-change of the temperature signal, their algorithm is able to detect non-wear periods for intervals as short as 5 minutes. Remark that this high temporal specificity is unattainable when exclusively using movement signals. Vert et al. also emphasized the importance of detecting such shorter non-wear periods in free-living scenarios, which often include short removals, e.g., when showering or washing hands. Similarly, Pagnamenta et al. (2022) [55] integrated temperature data into their non-wear detection algorithms for the Axivity AX3 wrist-worn wearable. They used a relative temperature threshold of 3°C to identify non-wear periods for 5-minute windows, achieving high sensitivity and specificity compared to algorithms relying solely on accelerometer data. Lastly, Böttcher et al. (2022) [22] developed an on-body score for the Empatica E4, combining the skin conductance, skin temperature, and movement signals. This binary on-body score is computed for 1-minute intervals and assesses data quality in retrospective datasets. However, these approaches often require hyperparameter configuration specific to the wearable and climate, limiting their generalizability. Additionally, only limited efforts have been made towards optimizing these algorithms.

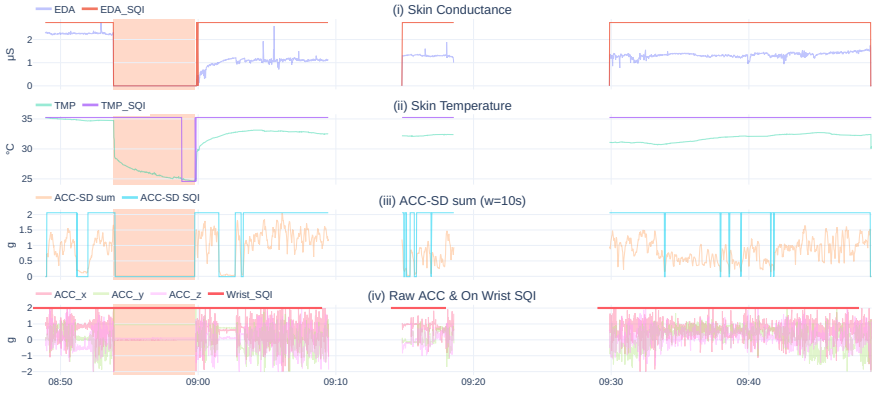
6.5.1.1 Countermeasures

Given that both datasets under consideration in this work utilize the Empatica E4, and previous studies indicated enhanced accuracy when incorporating physiological signal modalities, we refine Böttcher's algorithm to be more efficient and sensitive. Table 6.3 compares parameter values for both algorithm versions.

Specifically, we simplified the movement standard deviation computation by considering only the x-axis of the ACC signal, as opposed to calculating the standard deviation for all three axes, followed by a summation. This simplification improves the efficiency, as ACC-based operations proved to be the bottleneck of Böttcher's algorithm,



(a) Our refined non-wear algorithm



(b) Non-wear algorithm of Böttcher et al. [22]

Figure 6.8: Visual comparison of Böttcher's and our refined non-wear detection algorithm on the same excerpt. *Note*: The red-shaded area in each subplot of both (a) and (b) represents a labeled non-wear interval. Subplot (i) and (ii) in subfigure (a) and (b) depict the signal-specific SQIs for the skin conductance and temperature, while subplot (iii) represents the standard deviation of the ACC and the corresponding ACC-SD SQI. Subplot (iv) shows the three-axis accelerometer data alongside the resulting "Wrist_SQI". A low Wrist_SQI value between 08h55 and 09h00 in subfigure (a) denotes non-wear. Examining this time interval in subplots (i) and (ii) of (a), a notable decline in skin conductance and temperature is observed, leading to low SQI values. Minimal movement within this interval also reflects a low SQI value in subplot (iii). Conversely, in subfigure (b), this non-wear bout remains undetected, primarily due to the valid temperature SQI range (i.e., between 25 °C and 40 °C). This lower bound may be set too low, as only the last part of the skin temperature segment during this non-wear period results in a low SQI.

Table 6.3: Algorithmic and parameter-based comparison of two non-wear algorithms.

	Bottcher et al.	**Refined (ours)**
Movement SQI	ACC-SD sum (SD window = 10s) >= 0.2g	ACC_x-SD (SD window = 1s) >= 0.1 g
Skin temperature SQI	25 °C <= valid <= 40 °C	>= 32 °C
Skin conductance SQI	>= 0.05 μS	>= 0.03 μS
SQI processing	1-minute mean per SQI >= 1% on-body ->valid	Reindexing (i.e., ensuring a shared index)
SQI aggregation	OR-aggregation	OR-aggregation smoothing
Inference	38 ms per hour(*)	6 ms per hour(*)
Granularity	1 minute	0.25 seconds

(*) Both inference timings were computed on the same hardware.
A reference notebook with both implementations and timing details can be found here.

given its high sample rate (32 Hz). Using a smaller window size for the sliding window SD computation (1 second instead of 10 seconds) further enhanced efficiency. Empirical validation indicated a high correlation between the simplified and the original ACC-SD signal. Next on, in alignment with the work of Böttcher, the ACC-SD signal is transformed into a binary signal quality index (SQI) by using a threshold value. Both versions employ thresholding for skin conductance and temperature, resulting in signal-specific SQIs, with different empirically determined threshold values.

After this step, our approach deviates more from Böttcher's algorithm. In particular, Böttcher aggregates each of the three SQIs to a binary value per 1-minute window by determining whether more than 1% of that SQI is considered on-body. Subsequently, these three 1-minute SQIs are combined into a single binary SQI that becomes on-body if at least one of the signals is on-body (OR-operation).

Conversely, our approach first ensures that the three SQIs are aligned by reindexing them to the timestamps of the skin conductance SQI signal (4 Hz). Subsequently, in line with Böttcher, the three SQIs are combined via an OR-operation. This combined SQI signal is then smoothed using a 1-minute window, factoring out brief instances of wear and non-wear misdetections. This results in our final "Wrist_SQI", illustrated in Figure 6.8 (a).

Figure 6.8 compares Böttcher's and our non-wear detection algorithms, highlighting Böttcher's lower skin temperature sensitivity. Further implementation details of both algorithms can be found on GitHub. When tested on a consumer-grade desktop (AMD Ryzen 2600x), our refined non-wear detection pipeline demonstrated an inference time of 6ms per hour of E4 wearable data, a substantial improvement from Böttcher's 38ms (Table 6.3).This efficiency is crucial for longitudinal studies analyzing months of data for numerous participants or when using constrained devices. Additionally, our pipeline provides predictions with 0.25-second granularity, in contrast to the 1-minute

coarseness of Böttcher's algorithm. Supplemental S3 assesses the non-wear detection accuracy of both algorithms using a labeled subset from the mBrain dataset.

6.5.2 Challenge 6: Wearable Artifacts

Wearable artifacts, which cause spurious signal values, primarily result from either external factors (e.g., environmental noise, humidity) or sensor degradation. Unlike the controlled conditions within laboratory studies, ambulatory research is subject to varying external conditions, thereby requiring methodologies to identify or mitigate impacted modalities to account for such artifacts. Among these, motion-induced artifacts are the most prevalent and have a notable impact on the wearable's physiological modalities, including photoplethysmography and skin conductance signals [13]. Improper use of wearable devices, often due to not adhering to the recommended wearing guidelines, can also lead to artifacts, as observed by Stuyck et al. (2022) [56]. Additionally, sensor degradation, such as the polarization of skin conductance electrodes, is another major source of artifact generation [1, 57].

Since motion artifacts are a primary cause of signal corruption in wearable devices, numerous studies have turned to nighttime data as a means to mitigate this. Böttcher et al. (2022) [22] demonstrated that data collected between 20h to 8h exhibited substantially higher quality than daytime data. Siirtola et al. (2018) [10] conducted a wearable monitoring study, using the Empatica E4, on migraine patients to predict the likelihood of a migraine attack within the next day. They explicitly relied on nighttime data to compute reliable features. Uchida et al. (2022) [58] found that the median skin temperature acquired during the night via wrist-worn devices can indicate the fertility phase in women, demonstrating the reliability of nighttime data.

However, the efficacy of relying solely on nighttime data may vary depending on the study's specific objective. For objectives such as just-in-time interventions or analyzing physiological responses during daytime events (e.g., stress episodes or headaches), daytime data is crucial. In these cases, techniques like signal processing or signal estimation, depicted in Figure 6.9, can improve data analysis reliability.

Signal estimation leverages data-driven algorithms to enhance data quality by predicting or extracting a signal from noisy data. For instance, Reiss et al. (2019) [59]

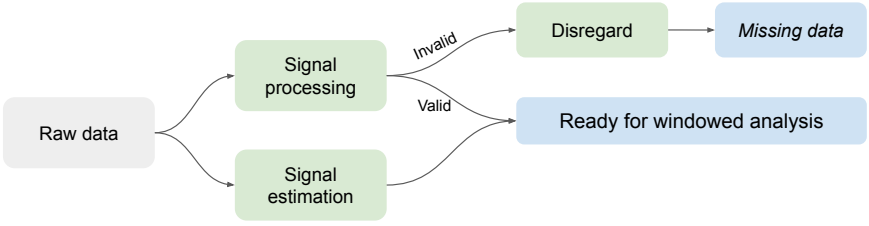


Figure 6.9: Flowchart for handling artifacts in raw ambulatory (daytime) wearable data.

used spectral representations of the Empatica E4's BVP signal to estimate the average instantaneous heart rate over 8-second intervals. However, a limitation of signal estimation methodologies is that they often replace the original signal without indicating the reliability of their estimations and typically require a reference ground signal truth which is leveraged by the data-driven technique [60].

In contrast, signal processing refines raw signals into more usable data for further analysis. Unlike signal estimation, signal processing often includes validity scores and is generally more interpretable. Consequently, our research emphasizes the visual application and analysis of signal processing techniques.

Figure 6.10: Skin conductance signal processing to discern valid and invalid regions and the resulting processed signal. *Note*: The figure consists of two vertically stacked subplots that share the same x-axis. The upper subplot displays the raw EDA signal depicted by the gray line, with valid and invalid SQI regions distinguished by green and red backgrounds, respectively. The processed EDA data is illustrated by an orange line. Remark that there is no one-on-one relationship between the processed EDA data and the valid regions. This is because the duration and frequency of these invalid regions affect the eventual retention of the raw EDA signal. Specifically, brief and infrequent invalid segments, like those until 12h05, can be effectively imputed using interpolation, resulting in no data exclusion in the processed EDA signal. Conversely, as the frequency and/or duration of invalid segments increases, evidenced between 12h05 and 12h06, successful interpolation is compromised, resulting in disregarding these invalid regions. Moreover, processed EDA segments, but shorter than 60 seconds (e.g., valid segments between 12h06 and 12h08), are excluded given their limited analytical utility. The lower subplot elucidates the components of the skin conductance SQI. In alignment with the non-wear detection pipeline, multiple sub-SQIs are utilized. The noise amplitude of the EDA, averaged over a two-second window, is delineated by a purple line. This signal is thresholded to determine the noise sub-SQI, marked by the green line.

6.5.2.1 Countermeasures

In alignment with our non-wear detection, signal processing solutions often utilize SQIs to differentiate valid from invalid segments. Visualizations are instrumental in shaping and evaluating these pipelines. As such, we introduce a generic visualization approach that we frequently employ through a skin conductance processing use case, shown in Figure 6.10.

Essentially, our approach utilizes multiple vertically stacked subplots, all sharing a common x-axis that denotes time. The uppermost subplot displays both the raw and processed signals, enabling a direct visual comparison. In this subplot, background shading accentuates the SQI outcome, simplifying the distinction between valid and invalid segments and their impact on the processed signal. Subsequent subplots provide insights into the components used in the processing pipeline, illuminating the composition of the final SQI seen in the upper subplot.

Notably, the visualization displayed in Figure 6.10 is realized by employing our widely adopted open-source Python tools. The processing pipeline is constructed using `tsflex`, an efficient toolkit that offers functionality to wrap and serialize data processing functions for time series data, facilitating convenience and easy deployment [61]. The visualization is rendered with Plotly-Resampler, a highly scalable time series visualization tool, which facilitates back-testing on large amounts of data [62]. It is this interplay between efficient signal processing and scalable interactive visualization that drives thorough analysis and broad exploration of large data volumes [63].

6.5.3 Challenge 7: Missing Wearable Data

In ambulatory studies, encountering missing segments of wearable data is inevitable. Missing data can stem from non-random processes where the likelihood of missing data depends on other unobserved symptoms, such as the presence of severe symptoms or certain periods of the day (e.g., morning showers) [64]. Another non-random factor that may contribute to missing data is the type of activity that is performed, as certain activities may contribute towards increased artifacts, thereby introducing more missing data during signal processing (see Figure 6.9). Next to these two non-random sources of missing data, device particularities can contribute to random missing data, where the missing data likelihood is unrelated to any, possibly latent, factor. For instance, during the mBrain study, the absence of an automatic reconnection functionality for the E4 device led to data loss whenever Bluetooth connectivity was disrupted. Users had to manually restart and reconnect the device, resulting in reduced data volume compared to on-device logging, even further compounded by the increased battery consumption due to Bluetooth streaming [22].

Given the inevitability and the high prevalence of missing wearable data in ambulatory studies, its impact on study results should be considered.

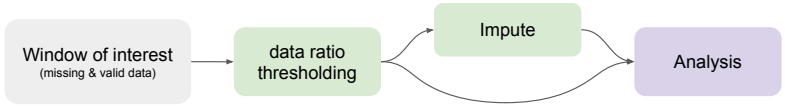


Figure 6.11: Flowchart illustrating the methodology for performing data analysis using incomplete data.

6.5.3.1 Countermeasures

In the previous subsection, we outlined how signal processing and signal estimation methodologies can be utilized to address spurious data segments, leading to enhanced or excluded segments. The resulting processed data is then suitable for qualitative analysis. As depicted in Figure 6.11, we particularly focus on "window-of-interest"-based analyses. This approach is widely employed in wearable monitoring research to partition collected data into windows for which the analysis will take place [10, 11, 65]. These analysis windows should be derived from the study's research question. For example, if the study aims to understand the precursors of headache attacks, the window-of-interest could include wearable data from the day before an attack [24]. After defining these "windows-of-interest", one can evaluate the proportion of missing and valid data within them. Figure 6.12 presents a complementary cumulative distribution plot of two participants, showing data availability across different data ratios. From this plot, one can easily interpret the number of samples that are available for a given data ratio per participant.

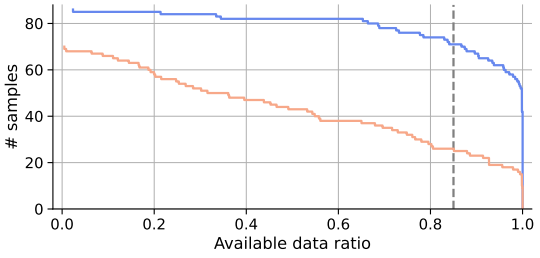


Figure 6.12: Complementary cumulative distribution plot of the window-of-interest data ratios for two participants. *Note*: The y-axis represents the number of available window-of-interest samples, while the x-axis indicates the corresponding data ratio. Each curve in the plot represents the complementary cumulative distribution of a participant, providing a visual assessment of overall data availability per participant. Furthermore, when utilizing a data-ratio threshold, exemplified via the dashed vertical gray line for the data-ratio of 0.85, this visualization allows determining the remaining number of samples adhering to this threshold.

A possible countermeasure to this missing data is imputation, which replaces missing values with aggregated imputation values [66, 67, 68]. Weed et al. (2022) [29] validated multiple imputation techniques for a 5-day window computation of actigraphy metrics

using wearable movement data, finding that median same-time-of-day imputation yields the best results. A recent survey of Di et al. [64] provides an overview of the most common imputation strategies applied to digital health time series data, also highlighting the efficacy of time-of-day-based imputations and the rise of deep learning in this domain.

To rigorously assess the impact of missing data segments on analytical results, it is generally recommended to utilize bootstrapping combined with gap simulation [69, 70]. However, such analyses are often neglected, as literature tends to exclude "windows-of-interest" that contain missing data [29]. In addition, there is limited research available on data imputation for variable and short-term (i.e., sub-day) intervals, such as stress events or headache periods [29]. We are therefore among the first to introduce a detailed procedure for assessing the impact of missing wearable data on outcome metrics, as such allowing the inclusion of windows with missing data. Figure 6.13 illustrates this procedure using a wearable accelerometer signal.

Starting with the window-of-interest, the first step entails selecting a processed, gap-free series and then computing the analysis metrics to obtain the *reference* metric values. For a wearable movement use case, as exemplified by Figure 6.13, this step is illustrated in subplots (ii) and (iv).

The second step involves gap bootstrapping. Specifically, one or multiple gaps are induced to achieve a certain data retention ratio. The gap induction method should mimic how missing data appears in other incomplete windows of interest. When dealing with wearable data, arbitrarily removing points is illogical. Instead, block-based gap induction methods which represent non-wear bouts are recommended [69]. We provide a comprehensive comparison of gap induction methodologies applied to wearable data bootstrapping in Supplemental S4.

To facilitate statistical analysis, multiple repeats of the second step are conducted on each chosen, processed, and complete reference series. This yields a set of metric values under varying simulated gap conditions, allowing observing the distribution and spread relative to the reference (gap-free) metric value, as shown in subplot (v) of Figure 6.13.

When gaps of varying data retention ratios are simulated, we can explore the impact of the data retention ratio on metric variability. Swarm plots or box plots can visualize this by showing distributions for each combination of metric, data retention ratio, and reference series, as depicted in Figure 6.14. Used alongside the cumulative data-ratio plot in Figure 6.12, this facilitates data-driven decisions regarding the data retention ratio threshold for windows-of-interest.

Remark that the proposed bootstrapping analysis has certain limitations. A significant drawback is that the gap induction procedure does not account for potential biases related to the specific times when participants are not wearing the watch, as outlined in *Challenge 4 (bias)*. Conversely, solely including complete windows of interest, which is common practice in literature, may also introduce certain biases in the downstream

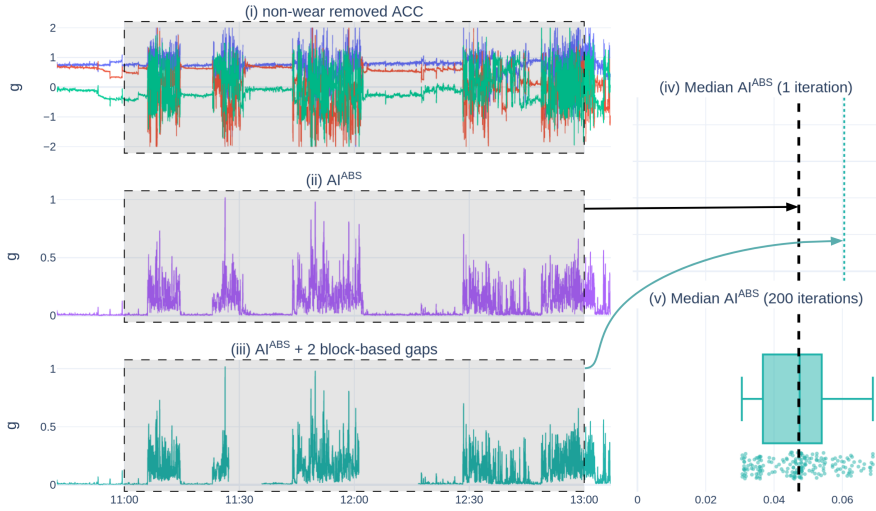


Figure 6.13: Overview of a single block-based bootstrapping iteration using the median as desired metric. *Note*: The figure comprises three vertically stacked subplots on the left that share an x-axis, with the window-of-interest highlighted by a gray-shaded area. The two vertical subplots on the right side share an x-axis as well. Subplot (i) depicts an excerpt of processed wearable movement data, for which non-wear periods have been removed. Remark that no non-wear periods were detected, and no data is missing, resulting in a complete valid segment for our window of interest. Subplot (ii) visualizes the transformed ACC data of (i) into a second-per-second activity intensity index, AI^{ABS}, in accordance with [71]. This AI^{ABS} signal is then utilized to compute our desired metric values, specifically, the median value of all data within our window of interest. This reference metric value is represented by a bold dashed black line in subplots (iv) and (v). Subsequently, gap-based bootstrapping is employed utilizing the complete movement intensity data from subplot (ii) as input. In particular, one or multiple block-based gaps are generated to create a gap-induced signal, shown in subplot (iii), maintaining a specific retention data ratio, which is in this illustration 0.6. The modified signal is then used to compute the desired metric, which is depicted by the vertical green dotted line in subplot (iv). Each bootstrap iteration results in adding another data point in subplot (v), which can then be utilized to assess the spread for a given data retention ratio. Further specifics can be found on Github.

analysis.

Furthermore, these analyses are more representative when all windows-of-interest occur at fixed time spans (e.g., the wakeful period from 10h to 20h on the day prior to a headache event), instead of varying day-time ranges (e.g., three hours before a stress event). This is because the time of day influences the occurrence and nature of gaps in the data, as indicated by Weed et al. (2022) [29].

As mentioned earlier, imputation is a viable option for dealing with partially missing data. The impact of imputation methods can also be analyzed using our proposed bootstrapping spread analysis methodology.

For temporal cyclical data, such as circadian dependent data, cosinor-based rhyth-

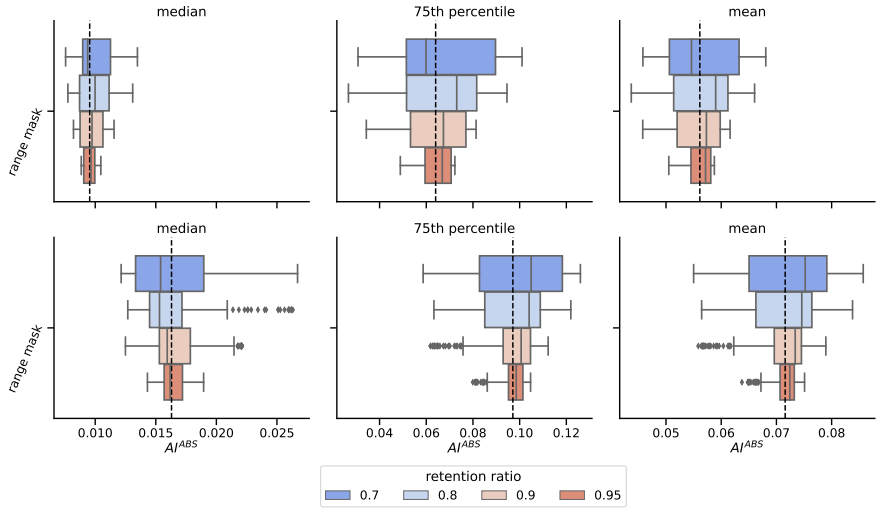


Figure 6.14: Spread analysis of block-based gap bootstrapping for various data ratios and metrics. *Note*: Each row in the figure represents a distinct reference series, signifying a window of interest from a unique moment. Different columns correspond to varying metrics, with the vertical dashed black line illustrating the metric value of the gap-free reference series. In creating this specific visualization, the accelerometer data from the Empatica E4 was transformed into a second-by-second activity index, AI^{ABS}, as per the methodology detailed by (Bai et al., 2016) and illustrated in Figure 6.13. The considered metrics are the 50th percentile, 75th percentile, and mean values calculated from the AI^{ABS} data of the selected time window.

mometry may prove useful as it is a gap-robust methodology that can deal with imbalance [72, 73]. Lastly, it is also advisable to consult literature to cross-reference the data ratios employed in prior research, if available.

6.5.4 Summary

To conclude the wearable data quality section, we summarize the presented three challenges and their countermeasures in Table 6.4.

6.6 Limitations

In this work, we addressed seven data quality challenges in ambulatory wearable monitoring studies, focusing on issues related to participants, monitoring applications, and wearable devices.

While we touched upon participant-related aspects, including user burden and application experience, we did not extensively explore psychological dimensions. For example, intrinsic motivation, which significantly influences study engagement, was not

Table 6.4: Summary of wearable data quality challenges and their countermeasures.

Challenge	Countermeasures / Actions
Wearable non-wear	
	Perform non-wear detection as a preprocessing (data filtering) step
Wearable artifacts	
	steer clear off: Utilize nighttime data (overall higher data quality) Signal processing: discern validity of signals (& enhance) Signal estimation: estimate target signal Visualizations of signal processing and estimation steps are crucial for quality assessment
Missing and Spurious data	
	Visualize available (processed) data retention ratios for participants Computing metrics with gaps or imputation: Bootstrapping techniques aid in assessing the spread of your outcome metric for a given data-ratio

covered [74]. Therefore, psychological aspects should also be considered in longitudinal studies.

We recognize that our study did not quantitatively measure the impact of several countermeasures beyond non-wear detection. While certain mBrain study design choices, such as the specific validation questions used for data completion, were made based on insights available at the time, their effects on data quality and availability were not systematically assessed through methods like A/B testing. This limitation restricts our ability to generalize the benefits of these optimizations to other settings, as they may be influenced by social, technical, or other biases specific to the mBrain study.

Regarding wearable-related challenges, we focused on introducing innovative methodologies targeting wearable data quality, especially for daytime-based analyses. However, our wearable-related countermeasures are not devoid of limitations. Both the ETRI and mBrain datasets rely on the Empatica E4 wearable device, constraining our analytical examples to a single device that has been discontinued. While we believe that most of our countermeasures are wearable agnostic, device-specific characteristics might affect data quality and subsequent analyses. Via this notebook, we aim to showcase a certain generality of our non-wear and signal processing pipelines towards the Empatica EmbracePlus device, which is the successor of the E4. Future research should extend our methodologies to a diverse range of wearable devices, including smartphones and chest-strap wearables.

Another limitation is that we did not explore wearable synchronization extensively since only a single wearable was utilized in both studies. In the mBrain study, the Empatica E4 device was connected to the phone, whose timestamp was used to synchronize the Empatica, thus mitigating the smartphone and wearable synchronization challenge. However, this challenge is addressed in literature, such as the work of Wolling et al. (2021) [75], which provides a methodology for synchronizing multiple wearable devices that share a highly correlated signal, such as heart rate.

We also refrained from discussing the measurement sensitivity of certain wearable device types. For instance, if the objective of an ambulatory wearable study is to investigate activity patterns in participants, wrist-worn devices tend to be less accurate than chest or hip-based wearables in capturing activity energy expenditure (AEE) [76]. Milstein et al. (2020) [77] specifically evaluated the reliability of the Empatica E4's skin conductance signal using the MindWare Mobile Impedance Cardiograph device to acquire palm skin conductance data as reference. Their results concluded that the E4 was not able to produce reliable EDA data, which may be attributed to lower sweat gland density on the wrist compared to the hand palm [78]. Therefore, it is paramount during study design to first consult literature regarding the measurement sensitivity and limitations of your device at hand.

Lastly, our work focused on enhancing the data quality during collection and processing, without explicitly addressing the impact of these steps on model training and decision-making.

In summary, while our research offers valuable insights and methodologies for improving wearable data quality, it is crucial to consider its limitations and the need for future research to validate and extend its applicability and robustness.

6.7 Conclusion

Recent advancements in wearable sensing, particularly wrist-worn devices, offer promising solutions for longitudinal follow-up of chronic patients by shifting from intermittent, subjective self-reporting to objective, continuous monitoring. However, integrating and analyzing wearable data with health-related records presents unique challenges. We distinguished two main categories of data-quality challenges; (i) participant- and data-entry-related challenges, and (ii) wearable-related analysis challenges.

For each identified challenge, we provided insights into the causes, effects, and countermeasures. Particularly, we built upon our first-hand experience gathered during the mBrain study and utilized two public real-world datasets to illustrate both the challenges and the proposed countermeasures. This way, our work aimed to practically address the overlooked challenges in data collection and retrospective analysis in ambulatory wearable monitoring studies.

Regarding the participant- and data-entry-related challenges, a key overarching conclusion is that any component requiring user interactions should be intricately tied to the research objective and demand minimal user effort [13, 20]. The selected wearable device should align with the research goal in terms of measuring sensitivity and user burden, with minimizing user burden being paramount in longitudinal research settings [21]. Participant compliance can be monitored via compliance visualizations leveraging near real-time participant data streams, enabling timely re-instruction. To mitigate implicitness assumptions and minimize the likelihood of data entry errors, it is advisable to conduct monitoring studies in incremental waves, starting with a pilot

study. Questionnaires can help address implicitness assumptions, and incorporating tailored questionnaires that gauge for context can aid in assessing personal bias for highly subjective event labels like stress.

Turning to wearable-related data quality challenges, visualization plays a critical role in evaluating the quality of different signal modalities during data processing and analysis steps. Tools like `tsflex` and Plotly-Resampler enhance the ability to process and visualize these data modalities efficiently and at scale. We introduced an algorithm that performs better in both inference speed and accuracy for identifying non-wear periods, developed using these toolkits. A non-wear detection pipeline is essential to filter out non-wear bouts before further processing and analysis. Finally, we propose a bootstrapping methodology to assess the impact of incorporating incomplete windows-of-interest on analysis metrics.

In conclusion, we present practical solutions to prominent challenges in ambulatory monitoring research, thereby enhancing the quality and efficacy of data collection and analysis. By openly sharing our code scripts and a subset of the mBrain study data, we facilitate reproducibility and enable direct applicability in real-world settings.

Additional Information

Data Availability Statement

All code and a patient sample of the mBrain study are publicly available at https://github.com/predict-idlab/data-quality-challenges-wearables and https://www.kaggle.com/datasets/jonvdrdo/mbrain21/data, respectively.

Supplementary Information

All supplementary information is available at GitHub

References

[1] J. Heikenfeld et al. “Wearable sensors: modalities, challenges, and prospects”. en. In: *Lab on a Chip* 18.2 (2018), pp. 217–248. ISSN: 1473-0197, 1473-0189. DOI: 10.1039/C7LC00914C. URL: http://xlink.rsc.org/?DOI=C7LC00914C (visited on 07/28/2023).

[2] Mirza Mansoor Baig, Hamid GholamHosseini, Aasia A. Moqeem, Farhaan Mirza, and Maria Lindén. “A Systematic Review of Wearable Patient Monitoring Systems – Current Challenges and Opportunities for Clinical Adoption”. en. In: *Journal of Medical Systems* 41.7 (July 2017), p. 115. ISSN: 0148-5598, 1573-689X. DOI: 10.1007/s10916-017-0760-1. URL: http://link.springer.com/10.1007/s10916-017-0760-1 (visited on 05/24/2023).

[3] Monica L Taylor, Emma E Thomas, Centaine L Snoswell, Anthony C Smith, and Liam J Caffery. “Does remote patient monitoring reduce acute care use? A systematic review”. en. In: *BMJ Open* 11.3 (Mar. 2021), e040232. ISSN: 2044-6055, 2044-6055. DOI: 10.1136/bmjopen-2020-040232. URL: https://bmjopen.bmj.com/lookup/doi/10.1136/bmjopen-2020-040232 (visited on 12/12/2023).

[4] David C Klonoff. “Continuous glucose monitoring: roadmap for 21st century diabetes therapy”. In: *Diabetes care* 28.5 (2005). Publisher: Citeseer, pp. 1231–1239.

[5] Karim Bayoumy et al. “Smart wearable devices in cardiovascular care: where we are and how to move forward”. en. In: *Nature Reviews Cardiology* (Mar. 2021). ZSCC: 0000000. ISSN: 1759-5002, 1759-5010. DOI: 10.1038/s41569-021-00522-7. URL: http://www.nature.com/articles/s41569-021-00522-7 (visited on 03/10/2021).

[6] Mary M. Rodgers, Vinay M. Pai, and Richard S. Conroy. “Recent Advances in Wearable Sensors for Health Monitoring”. en. In: *IEEE Sensors Journal* 15.6 (June 2015), pp. 3119–3126. ISSN: 1530-437X, 1558-1748. DOI: 10.1109/JSEN.2014.2357257. URL: https://ieeexplore.ieee.org/document/6899605 (visited on 09/10/2023).

[7] Jay Chen, Karric Kwong, Dennis Chang, Jerry Luk, and Ruzena Bajcsy. “Wearable sensors for reliable fall detection”. In: *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*. IEEE, 2006, pp. 3551–3554.

[8] Jayoung Kim, Alan S. Campbell, Berta Esteban-Fernández de Ávila, and Joseph Wang. “Wearable biosensors for healthcare monitoring”. en. In: *Nature Biotechnology* 37.4 (Apr. 2019). ZSCC: 0000319, pp. 389–406. ISSN: 1087-0156, 1546-1696. DOI: 10.1038/s41587-019-0045-y. URL: http://www.nature.com/articles/s41587-019-0045-y (visited on 08/27/2020).

[9] Mathias De Brouwer et al. "mBrain: towards the continuous follow-up and headache classification of primary headache disorder patients". en. In: *BMC Medical Informatics and Decision Making* 22.1 (Dec. 2022), p. 87. ISSN: 1472-6947. DOI: 10.1186/s12911-022-01813-w. URL: https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/s12911-022-01813-w (visited on 06/16/2023).

[10] Pekka Siirtola, Heli Koskimäki, Henna Mönttinen, and Juha Röning. "Using Sleep Time Data from Wearable Sensors for Early Detection of Migraine Attacks". en. In: *Sensors* 18.5 (Apr. 2018), p. 1374. ISSN: 1424-8220. DOI: 10.3390/s18051374. URL: http://www.mdpi.com/1424-8220/18/5/1374 (visited on 05/15/2023).

[11] Anker Stubberud, Sigrid Hegna Ingvaldsen, Eiliv Brenner, Ingunn Winnberg, Alexander Olsen, Gøril Bruvik Gravdahl, Manjit Singh Matharu, Parashkev Nachev, and Erling Tronvik. "Forecasting migraine with machine learning based on mobile phone diary and wearable data". en. In: *Cephalalgia* 43.5 (May 2023), p. 033310242311692. ISSN: 0333-1024, 1468-2982. DOI: 10.1177/03331024231169244. URL: http://journals.sagepub.com/doi/10.1177/03331024231169244 (visited on 04/28/2023).

[12] Sebastian Böttcher et al. "Detecting Tonic-Clonic Seizures in Multimodal Biosignal Data From Wearables: Methodology Design and Validation". en. In: *JMIR mHealth and uHealth* 9.11 (Nov. 2021), e27674. ISSN: 2291-5222. DOI: 10.2196/27674. URL: https://mhealth.jmir.org/2021/11/e27674 (visited on 08/17/2023).

[13] Philip Schmidt, Attila Reiss, Robert Dürichen, and Kristof Van Laerhoven. "Wearable-Based Affect Recognition—A Review". en. In: *Sensors* 19.19 (Sept. 2019). ZSCC: 0000022, p. 4079. ISSN: 1424-8220. DOI: 10.3390/s19194079. URL: https://www.mdpi.com/1424-8220/19/19/4079 (visited on 12/08/2020).

[14] Yatharth Ranjan et al. "RADAR-Base: Open Source Mobile Health Platform for Collecting, Monitoring, and Analyzing Data Using Sensors, Wearables, and Mobile Devices". en. In: *JMIR mHealth and uHealth* 7.8 (Aug. 2019), e11734. ISSN: 2291-5222. DOI: 10.2196/11734. URL: https://mhealth.jmir.org/2019/8/e11734/ (visited on 05/27/2024).

[15] Stefano Canali, Viola Schiaffonati, and Andrea Aliverti. "Challenges and recommendations for wearable devices in digital health: Data quality, interoperability, health equity, fairness". en. In: *PLOS Digital Health* 1.10 (Oct. 2022). Ed. by Shelagh Mulvaney, e0000104. ISSN: 2767-3170. DOI: 10.1371/journal.pdig.0000104. URL: https://dx.plos.org/10.1371/journal.pdig.0000104 (visited on 09/22/2023).

[16] Sylvia Cho, Ipek Ensari, Chunhua Weng, Michael G Kahn, and Karthik Natarajan. "Factors Affecting the Quality of Person-Generated Wearable Device Data and Associated Challenges: Rapid Systematic Review". en. In: *JMIR mHealth and uHealth* 9.3 (Mar. 2021), e20738. ISSN: 2291-5222. DOI: 10.2196/20738. URL: https://mhealth.jmir.org/2021/3/e20738 (visited on 05/24/2023).

[17] Yue Liao, Carrie Thompson, Susan Peterson, John Mandrola, and Muhammad Shaalan Beg. "The Future of Wearable Technologies and Remote Monitoring in Health Care". en. In: *American Society of Clinical Oncology Educational Book* 39 (May 2019), pp. 115–121. ISSN: 1548-8748, 1548-8756. DOI: 10.1200/EDBK_238919. URL: https://ascopubs.org/doi/10.1200/EDBK_238919 (visited on 12/01/2023).

[18] Janani Sriram, Minho Shin, David Kotz, Anand Rajan, Manoj Sastry, and Mark Yarvis. "Challenges in data quality assurance in pervasive health monitoring systems". In: *Future of Trust in Computing: Proceedings of the First International Conference Future of Trust in Computing 2008*. Springer. 2009, pp. 129–142.

[19] Seungeun Chung, Chi Yoon Jeong, Jeong Mook Lim, Jiyoun Lim, Kyoung Ju Noh, Gague Kim, and Hyuntae Jeong. "Real□world multimodal lifelog dataset for human behavior study". en. In: *ETRI Journal* 44.3 (June 2022), pp. 426–437. ISSN: 1225-6463, 2233-7326. DOI: 10.4218/etrij.2020-0446. URL: https://onlinelibrary.wiley.com/doi/10.4218/etrij.2020-0446 (visited on 01/27/2023).

[20] Philip Schmidt, Attila Reiss, Robert Dürichen, and Kristof Van Laerhoven. "Labelling Affective States "in the Wild": Practical Guidelines and Lessons Learned". en. In: *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*. ZSCC: 0000010. Singapore Singapore: ACM, Oct. 2018, pp. 654–659. ISBN: 978-1-4503-5966-5. DOI: 10.1145/3267305.3267551. URL: https://dl.acm.org/doi/10.1145/3267305.3267551 (visited on 02/26/2021).

[21] Guilherme M Balbim, Isabela G Marques, David X Marquez, Darshilmukesh Patel, Lisa K Sharp, Spyros Kitsiou, and Sharmilee M Nyenhuis. "Using Fitbit as an mHealth Intervention Tool to Promote Physical Activity: Potential Challenges and Solutions". en. In: *JMIR mHealth and uHealth* 9.3 (Mar. 2021), e25289. ISSN: 2291-5222. DOI: 10.2196/25289. URL: https://mhealth.jmir.org/2021/3/e25289 (visited on 05/09/2023).

[22] Sebastian Böttcher et al. "Data quality evaluation in wearable monitoring". en. In: *Scientific Reports* 12.1 (Dec. 2022), p. 21412. ISSN: 2045-2322. DOI: 10.1038/s41598-022-25949-x. URL: https://www.nature.com/articles/s41598-022-25949-x (visited on 12/18/2022).

[23] Jonas Van Der Donckt, Mathias De Brouwer, Pieter Moens, Marija Stojchevska, Bram Steenwinckel, Stef Pletinck, Nicolas Vandenbussche, Annelis Goris, Koen Paemeleire, Femke Ongenae, et al. "From self-reporting to monitoring for improved migraine management". In: *Engineer meets Physician (EmP)*. 2022.

[24] Nicolas Vandenbussche, Jonas Van Der Donckt, Mathias De Brouwer, Bram Steenwinckel, Marija Stojchevska, Femke Ongenae, Sofie Van Hoecke, and Koen Paemeleire. "Patients with chronic cluster headache may show reduced activity energy expenditure on ambulatory wrist actigraphy recordings during daytime attacks". en. In: *Brain and Behavior* 14.1 (Jan. 2024), e3360. ISSN:

2162-3279, 2162-3279. DOI: 10.1002/brb3.3360. URL: https://onlinelibrary.wiley.com/doi/10.1002/brb3.3360 (visited on 01/17/2024).

[25] Empatica S.R.L. *E4 data - BVP expected signal*. en-US. URL: https://support.empatica.com/hc/en-us/articles/360029719792-E4-data-BVP-expected-signal (visited on 09/25/2023).

[26] Kaggle. *Kaggle State of Machine Learning and Data Science Report 2022.pdf*. English. survey. 2022. URL: https://www.kaggle.com/c/kaggle-survey-2022/data.

[27] Jeffrey M. Perkel. "Why Jupyter is data scientists' computational notebook of choice". eng. In: *Nature* 563.7729 (Nov. 2018), pp. 145–146. ISSN: 1476-4687. DOI: 10.1038/d41586-018-07196-1.

[28] PyPoetry. *Poetry - Python dependency management and packaging made easy*. URL: https://python-poetry.org/ (visited on 09/25/2023).

[29] Lara Weed, Renske Lok, Dwijen Chawra, and Jamie Zeitzer. "The Impact of Missing Data and Imputation Methods on the Analysis of 24-Hour Activity Patterns". en. In: *Clocks & Sleep* 4.4 (Sept. 2022), pp. 497–507. ISSN: 2624-5175. DOI: 10.3390/clockssleep4040039. URL: https://www.mdpi.com/2624-5175/4/4/39 (visited on 02/08/2023).

[30] Irene Heger, Kay Deckers, Marjolein De Vugt, Frans Verhey, Anke Oenema, Martin Van Boxtel, and Sebastian Köhler. "Using mHealth for Primary Prevention of Dementia: A Proof-of-Concept Study on Usage Patterns, Appreciation, and Beliefs and Attitudes Regarding Prevention". en. In: *Journal of Alzheimer's Disease* 94.3 (Aug. 2023), pp. 935–948. ISSN: 13872877, 18758908. DOI: 10.3233/JAD-230225. URL: https://www.medra.org/servlet/aliasResolver?alias=iospress&doi=10.3233/JAD-230225 (visited on 09/25/2023).

[31] Amir Muaremi, Bert Arnrich, and Gerhard Tröster. "Towards Measuring Stress with Smartphones and Wearable Devices During Workday and Sleep". en. In: *BioNanoScience* 3.2 (June 2013). ZSCC: 0000245, pp. 172–183. ISSN: 2191-1630, 2191-1649. DOI: 10.1007/s12668-013-0089-2. URL: http://link.springer.com/10.1007/s12668-013-0089-2 (visited on 02/11/2021).

[32] Stephen R. Porter, Michael E. Whitcomb, and William H. Weitzer. "Multiple surveys of students and survey fatigue". en. In: *New Directions for Institutional Research* 2004.121 (2004), pp. 63–73. ISSN: 0271-0579, 1536-075X. DOI: 10.1002/ir.101. URL: https://onlinelibrary.wiley.com/doi/10.1002/ir.101 (visited on 08/17/2023).

[33] Aksel Paulsen, Søren Overgaard, and Jens Martin Lauritsen. "Quality of Data Entry Using Single Entry, Double Entry and Automated Forms Processing–An Example Based on a Study of Patient-Reported Outcomes". en. In: *PLoS ONE* 7.4 (Apr. 2012). Ed. by Andrew Boulle, e35087. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0035087. URL: https://dx.plos.org/10.1371/journal.pone.0035087 (visited on 07/09/2024).

[34] Jennifer Healey, Lama Nachman, Sushmita Subramanian, Junaith Shahabdeen, and Margaret Morris. "Out of the lab and into the fray: Towards modeling emotion in everyday life". In: *Pervasive Computing: 8th International Conference, Pervasive 2010, Helsinki, Finland, May 17-20, 2010. Proceedings 8*. Springer. 2010, pp. 156–173.

[35] Charlotte Ottenstein and Linda Werner. "Compliance in Ambulatory Assessment Studies: Investigating Study and Sample Characteristics as Predictors". en. In: *Assessment* 29.8 (Dec. 2022), pp. 1765–1776. ISSN: 1073-1911, 1552-3489. DOI: 10.1177/10731911211032718. URL: http://journals.sagepub.com/doi/10.1177/10731911211032718 (visited on 09/26/2023).

[36] Niels Van Berkel, Jorge Goncalves, Simo Hosio, and Vassilis Kostakos. "Gamification of Mobile Experience Sampling Improves Data Quality and Quantity". en. In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1.3 (Sept. 2017), pp. 1–21. ISSN: 2474-9567. DOI: 10.1145/3130972. URL: https://dl.acm.org/doi/10.1145/3130972 (visited on 08/16/2023).

[37] Florian Fischer and Sina Kleen. "Possibilities, Problems, and Perspectives of Data Collection by Mobile Apps in Longitudinal Epidemiological Studies: Scoping Review". en. In: *Journal of Medical Internet Research* 23.1 (Jan. 2021), e17691. ISSN: 1438-8871. DOI: 10.2196/17691. URL: http://www.jmir.org/2021/1/e17691/ (visited on 07/09/2024).

[38] Andrew T. Gloster, Marcel Miché, Hanna Wersebe, Thorsten Mikoteit, Jürgen Hoyer, Christian Imboden, Klaus Bader, Andrea H. Meyer, Martin Hatzinger, and Roselind Lieb. "Daily fluctuation of emotions and memories thereof: Design and methods of an experience sampling study of major depression, social phobia, and controls". en. In: *International Journal of Methods in Psychiatric Research* 26.3 (Sept. 2017), e1578. ISSN: 1049-8931, 1557-0657. DOI: 10.1002/mpr.1578. URL: https://onlinelibrary.wiley.com/doi/10.1002/mpr.1578 (visited on 09/26/2023).

[39] Reza Rawassizadeh, Elaheh Momeni, Chelsea Dobbins, Joobin Gharibshah, and Michael Pazzani. "Scalable Daily Human Behavioral Pattern Mining from Multivariate Temporal Data". en. In: *IEEE Transactions on Knowledge and Data Engineering* 28.11 (Nov. 2016), pp. 3098–3112. ISSN: 1041-4347. DOI: 10.1109/TKDE.2016.2592527. URL: http://ieeexplore.ieee.org/document/7523395/ (visited on 05/15/2024).

[40] Paul A. Harris et al. "The REDCap consortium: Building an international community of software platform partners". en. In: *Journal of Biomedical Informatics* 95 (July 2019), p. 103208. ISSN: 15320464. DOI: 10.1016/j.jbi.2019.103208. URL: https://linkinghub.elsevier.com/retrieve/pii/S1532046419301261 (visited on 05/23/2024).

[41] Susan M. Fox-Wasylyshyn and Maher M. El-Masri. "Handling missing data in self-report measures". en. In: *Research in Nursing & Health* 28.6 (Dec. 2005), pp. 488–495. ISSN: 0160-6891, 1098-240X. DOI: 10.1002/nur.20100. URL:

https://onlinelibrary.wiley.com/doi/10.1002/nur.20100 (visited on 09/26/2023).

[42] Josh Colls, Yvonne C Lee, Chang Xu, Cassandra Corrigan, Fengxin Lu, Georgia Marquez-Grap, Meredith Murray, Dong H Suh, and Daniel H Solomon. "Patient adherence with a smartphone app for patient-reported outcomes in rheumatoid arthritis". en. In: *Rheumatology* 60.1 (Jan. 2021), pp. 108–112. ISSN: 1462-0324, 1462-0332. DOI: 10.1093/rheumatology/keaa202. URL: https://academic.oup.com/rheumatology/article/60/1/108/5861028 (visited on 09/26/2023).

[43] Mirza Mansoor Baig, Hamid GholamHosseini, and Martin J. Connolly. "Mobile healthcare applications: system design review, critical issues and challenges". en. In: *Australasian Physical & Engineering Sciences in Medicine* 38.1 (Mar. 2015), pp. 23–38. ISSN: 0158-9938, 1879-5447. DOI: 10.1007/s13246-014-0315-4. URL: http://link.springer.com/10.1007/s13246-014-0315-4 (visited on 12/14/2023).

[44] T Walsh and P C W Beatty. "Human factors error and patient monitoring". en. In: *Physiological Measurement* 23.3 (Aug. 2002), R111–R132. ISSN: 09673334. DOI: 10.1088/0967-3334/23/3/201. URL: https://iopscience.iop.org/article/10.1088/0967-3334/23/3/201 (visited on 08/16/2023).

[45] Mihaly Csikszentmihalyi, Mihaly Csikszentmihalyi, and Reed Larson. "Validity and reliability of the experience-sampling method". In: *Flow and the foundations of positive psychology: The collected works of Mihaly Csikszentmihalyi* (2014). Publisher: Springer, pp. 35–54.

[46] Alexander Hoelzemann and Kristof Van Laerhoven. *A Matter of Annotation: An Empirical Study on In Situ and Self-Recall Activity Annotations from Wearable Sensors*. en. arXiv:2305.08752 [cs]. May 2023. URL: http://arxiv.org/abs/2305.08752 (visited on 05/24/2023).

[47] Vincent Bracke, Merlijn Sebrechts, Bart Moons, Jeroen Hoebeke, Filip De Turck, and Bruno Volckaert. "Design and evaluation of a scalable Internet of Things backend for smart ports". en. In: *Software: Practice and Experience* 51.7 (July 2021), pp. 1557–1579. ISSN: 0038-0644, 1097-024X. DOI: 10.1002/spe.2973. URL: https://onlinelibrary.wiley.com/doi/10.1002/spe.2973 (visited on 11/29/2023).

[48] Shaoxiong Sun et al. "The utility of wearable devices in assessing ambulatory impairments of people with multiple sclerosis in free-living conditions". en. In: *Computer Methods and Programs in Biomedicine* 227 (Dec. 2022), p. 107204. ISSN: 01692607. DOI: 10.1016/j.cmpb.2022.107204. URL: https://linkinghub.elsevier.com/retrieve/pii/S0169260722005855 (visited on 10/05/2023).

[49] Christophe Mombers, Kathleen Legako, and Annette Gilchrist. "Identifying medical wearables and sensor technologies that deliver data on clinical endpoints: Editorial". en. In: *British Journal of Clinical Pharmacology* 81.2 (Feb. 2016), pp. 196–198. ISSN: 03065251. DOI: 10.1111/bcp.12818. URL: https://onlinelibrary.wiley.com/doi/10.1111/bcp.12818 (visited on 08/17/2023).

[50] Alon Vaisman, Grace Bannerman, John Matelski, Kathryn Tinckam, and Susy S Hota. "Out of sight, out of mind: a prospective observational study to estimate the duration of the Hawthorne effect on hand hygiene events". en. In: *BMJ Quality & Safety* 29.11 (Nov. 2020), pp. 932–938. ISSN: 2044-5415, 2044-5423. DOI: 10.1136/bmjqs-2019-010310. URL: https://qualitysafety.bmj.com/lookup/doi/10.1136/bmjqs-2019-010310 (visited on 09/26/2023).

[51] Ann M. Berger, Kimberly K. Wielgus, Stacey Young-McCaughan, Patricia Fischer, Lynne Farr, and Kathryn A. Lee. "Methodological Challenges When Using Actigraphy in Research". en. In: *Journal of Pain and Symptom Management* 36.2 (Aug. 2008), pp. 191–199. ISSN: 08853924. DOI: 10.1016/j.jpainsymman.2007.10.008. URL: https://linkinghub.elsevier.com/retrieve/pii/S0885392408001127 (visited on 06/22/2023).

[52] Jongyoon Choi, Beena Ahmed, and Ricardo Gutierrez-Osuna. "Development and Evaluation of an Ambulatory Stress Monitor Based on Wearable Sensors". In: *IEEE Transactions on Information Technology in Biomedicine* 16.2 (Mar. 2012). ZSCC: NoCitationData[s0] Conference Name: IEEE Transactions on Information Technology in Biomedicine, pp. 279–286. ISSN: 1558-0032. DOI: 10.1109/TITB.2011.2169804.

[53] Matthew N. Ahmadi, Nicole Nathan, Rachel Sutherland, Luke Wolfenden, and Stewart G. Trost. "Non-wear or sleep? Evaluation of five non-wear detection algorithms for raw accelerometer data". en. In: *Journal of Sports Sciences* 38.4 (Feb. 2020), pp. 399–404. ISSN: 0264-0414, 1466-447X. DOI: 10.1080/02640414.2019.1703301. URL: https://www.tandfonline.com/doi/full/10.1080/02640414.2019.1703301 (visited on 06/20/2023).

[54] Adam Vert, Kyle S. Weber, Vanessa Thai, Erin Turner, Kit B. Beyer, Benjamin F Cornish, F. Elizabeth Godkin, Christopher Wong, William E. McIlroy, and Karen Van Ooteghem. "Detecting accelerometer non-wear periods using change in acceleration combined with rate-of-change in temperature". en. In: *BMC Medical Research Methodology* 22.1 (Dec. 2022), p. 147. ISSN: 1471-2288. DOI: 10.1186/s12874-022-01633-6. URL: https://bmcmedresmethodol.biomedcentral.com/articles/10.1186/s12874-022-01633-6 (visited on 06/20/2023).

[55] Sara Pagnamenta, Karoline Blix Grønvik, Kamiar Aminian, Beatrix Vereijken, and Anisoara Paraschiv-Ionescu. "Putting Temperature into the Equation: Development and Validation of Algorithms to Distinguish Non-Wearing from Inactivity and Sleep in Wearable Sensors". en. In: *Sensors* 22.3 (Feb. 2022), p. 1117. ISSN: 1424-8220. DOI: 10.3390/s22031117. URL: https://www.mdpi.com/1424-8220/22/3/1117 (visited on 06/20/2023).

[56] Hans Stuyck, Leonardo Dalla Costa, Axel Cleeremans, and Eva Van Den Bussche. "Validity of the Empatica E4 wristband to estimate resting-state heart rate variability in a lab-based context". en. In: *International Journal of Psychophysiology* 182 (Dec. 2022), pp. 105–118. ISSN: 01678760. DOI: 10.1016/j.ijpsycho.2022.

10.003. URL: https://linkinghub.elsevier.com/retrieve/pii/S0167876022002409 (visited on 05/30/2023).

[57] Hugo F. Posada-Quintero and Ki H. Chon. "Innovations in Electrodermal Activity Data Collection and Signal Processing: A Systematic Review". en. In: *Sensors* 20.2 (Jan. 2020), p. 479. ISSN: 1424-8220. DOI: 10.3390/s20020479. URL: https://www.mdpi.com/1424-8220/20/2/479 (visited on 06/07/2020).

[58] Yuki Uchida and Masahiko Izumizaki. "The use of wearable devices for predicting biphasic basal body temperature to estimate the date of ovulation in women". en. In: *Journal of Thermal Biology* 108 (Aug. 2022), p. 103290. ISSN: 03064565. DOI: 10.1016/j.jtherbio.2022.103290. URL: https://linkinghub.elsevier.com/retrieve/pii/S0306456522001048 (visited on 06/30/2023).

[59] Attila Reiss, Ina Indlekofer, Philip Schmidt, and Kristof Van Laerhoven. "Deep PPG: Large-Scale Heart Rate Estimation with Convolutional Neural Networks". en. In: *Sensors* 19.14 (July 2019). ZSCC: 0000012, p. 3079. ISSN: 1424-8220. DOI: 10.3390/s19143079. URL: https://www.mdpi.com/1424-8220/19/14/3079 (visited on 08/06/2020).

[60] Bernhard A. Moser. "Estimating the signal reconstruction error from threshold-based sampling without knowing the original signal". In: *2017 3rd International Conference on Event-Based Control, Communication and Signal Processing (EBCCSP)*. Funchal, Portugal: IEEE, May 2017, pp. 1–4. ISBN: 978-1-5386-0915-6. DOI: 10.1109/EBCCSP.2017.8022834. URL: http://ieeexplore.ieee.org/document/8022834/ (visited on 11/30/2023).

[61] Jonas Van Der Donckt, Jeroen Van Der Donckt, Emiel Deprost, and Sofie Van Hoecke. "tsflex: Flexible time series processing & feature extraction". en. In: *SoftwareX* 17 (Jan. 2022). ZSCC: 0000000, p. 100971. ISSN: 23527110. DOI: 10.1016/j.softx.2021.100971. URL: https://linkinghub.elsevier.com/retrieve/pii/S2352711021001904 (visited on 03/03/2022).

[62] Jonas Van Der Donckt, Jeroen Van Der Donckt, Emiel Deprost, and Sofie Van Hoecke. "Plotly-Resampler: Effective Visual Analytics for Large Time Series". In: *2022 IEEE Visualization and Visual Analytics (VIS)*. Oklahoma City, OK, USA: IEEE, Oct. 2022, pp. 21–25. ISBN: 978-1-66548-812-9. DOI: 10.1109/VIS54862.2022.00013. URL: https://ieeexplore.ieee.org/document/9973221/ (visited on 06/16/2023).

[63] Jürgen Bernard, Tobias Ruppert, Oliver Goroll, Thorsten May, and Jörn Kohlhammer. "Visual-interactive preprocessing of time series data". In: *Proceedings of SIGRAD 2012; Interactive Visual Analysis of Data; November 29-30; 2012; Växjö; Sweden*. Issue: 081. Citeseer, 2012, pp. 39–48.

[64] Junrui Di, Charmaine Demanuele, Anna Kettermann, F. Isik Karahanoglu, Joseph C. Cappelleri, Andrew Potter, Denise Bury, Jesse M. Cedarbaum, and Bill Byrom. "Considerations to address missing data when deriving clinical trial endpoints from digital health technologies". en. In: *Contemporary Clinical Trials* 113 (Feb. 2022), p. 106661. ISSN: 15517144. DOI: 10.1016/j.cct.2021.106661.

URL: https://linkinghub.elsevier.com/retrieve/pii/S1551714421003979 (visited on 05/22/2024).

[65] Andreas Bulling, Ulf Blanke, and Bernt Schiele. "A tutorial on human activity recognition using body-worn inertial sensors". en. In: *ACM Computing Surveys* 46.3 (Jan. 2014), pp. 1–33. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/2499621. URL: https://dl.acm.org/doi/10.1145/2499621 (visited on 09/15/2023).

[66] Reza Rawassizadeh, Hamidreza Keshavarz, and Michael Pazzani. "Ghost Imputation: Accurately Reconstructing Missing Data of the Off Period". In: *IEEE Transactions on Knowledge and Data Engineering* 32.11 (Nov. 2020), pp. 2185–2197. ISSN: 1041-4347, 1558-2191, 2326-3865. DOI: 10.1109/TKDE.2019.2914653. URL: https://ieeexplore.ieee.org/document/8705333/ (visited on 07/09/2024).

[67] Carlotta Demeniconi and Nitesh Chawla, eds. *Proceedings of the 2020 SIAM International Conference on Data Mining*. en. Philadelphia, PA: Society for Industrial and Applied Mathematics, Jan. 2020. ISBN: 978-1-61197-623-6. DOI: 10.1137/1.9781611976236. URL: https://epubs.siam.org/doi/book/10.1137/1.9781611976236 (visited on 05/21/2024).

[68] Xian Wu, Stephen Mattingly, Shayan Mirjafari, Chao Huang, and Nitesh V. Chawla. "Personalized Imputation on Wearable-Sensory Time Series via Knowledge Transfer". en. In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. Virtual Event Ireland: ACM, Oct. 2020, pp. 1625–1634. ISBN: 978-1-4503-6859-9. DOI: 10.1145/3340531.3411879. URL: https://dl.acm.org/doi/10.1145/3340531.3411879 (visited on 07/09/2024).

[69] Jeremy Berkowitz and Lutz Kilian. "Recent developments in bootstrapping time series". en. In: *Econometric Reviews* 19.1 (Jan. 2000), pp. 1–48. ISSN: 0747-4938, 1532-4168. DOI: 10.1080/07474930008800457. URL: http://www.tandfonline.com/doi/abs/10.1080/07474930008800457 (visited on 08/08/2023).

[70] Bradley Efron. "Missing Data, Imputation, and the Bootstrap". en. In: *Journal of the American Statistical Association* 89.426 (June 1994), pp. 463–475. ISSN: 0162-1459, 1537-274X. DOI: 10.1080/01621459.1994.10476768. URL: https://www.tandfonline.com/doi/full/10.1080/01621459.1994.10476768 (visited on 08/08/2023).

[71] Jiawei Bai, Chongzhi Di, Luo Xiao, Kelly R. Evenson, Andrea Z. LaCroix, Ciprian M. Crainiceanu, and David M. Buchner. "An Activity Index for Raw Accelerometry Data and Its Comparison with Other Activity Metrics". en. In: *PLOS ONE* 11.8 (Aug. 2016). Ed. by Jaroslaw Harezlak. ZSCC: NoCitationData[s0], e0160644. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0160644. URL: https://dx.plos.org/10.1371/journal.pone.0160644 (visited on 12/16/2021).

[72] Germaine Cornelissen. "Cosinor-based rhythmometry". en. In: *Theoretical Biology and Medical Modelling* 11.1 (Dec. 2014), p. 16. ISSN: 1742-4682. DOI: 10.1186/1742-4682-11-16. URL: https://tbiomed.biomedcentral.com/articles/10.1186/1742-4682-11-16 (visited on 02/23/2023).

[73] Miha Moškon. "CosinorPy: a python package for cosinor-based rhythmometry". en. In: *BMC Bioinformatics* 21.1 (Dec. 2020), p. 485. ISSN: 1471-2105. DOI: 10.1186/s12859-020-03830-w. URL: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-020-03830-w (visited on 02/22/2023).

[74] Neal Chalofsky and Vijay Krishna. "Meaningfulness, Commitment, and Engagement:The Intersection of a Deeper Level of Intrinsic Motivation". en. In: *Advances in Developing Human Resources* 11.2 (Apr. 2009), pp. 189–203. ISSN: 1523-4223, 1552-3055. DOI: 10.1177/1523422309333147. URL: http://journals.sagepub.com/doi/10.1177/1523422309333147 (visited on 09/15/2023).

[75] Florian Wolling, Kristof van Laerhoven, Pekka Siirtola, and Juha Roning. "PulSync: The Heart Rate Variability as a Unique Fingerprint for the Alignment of Sensor Data Across Multiple Wearable Devices". en. In: *2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*. ZSCC: 0000001. Kassel, Germany: IEEE, Mar. 2021, pp. 188–193. ISBN: 978-1-66540-424-2. DOI: 10.1109/PerComWorkshops51409.2021.9431015. URL: https://ieeexplore.ieee.org/document/9431015/ (visited on 09/28/2021).

[76] Hans Van Remoortel et al. "Validity of Six Activity Monitors in Chronic Obstructive Pulmonary Disease: A Comparison with Indirect Calorimetry". en. In: *PLoS ONE* 7.6 (June 2012). Ed. by Marco Idzko, e39198. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0039198. URL: https://dx.plos.org/10.1371/journal.pone.0039198 (visited on 04/26/2023).

[77] Nir Milstein and Ilanit Gordon. "Validating Measures of Electrodermal Activity and Heart Rate Variability Derived From the Empatica E4 Utilized in Research Settings That Involve Interactive Dyadic States". en. In: *Frontiers in Behavioral Neuroscience* 14 (Aug. 2020), p. 148. ISSN: 1662-5153. DOI: 10.3389/fnbeh.2020.00148. URL: https://www.frontiersin.org/article/10.3389/fnbeh.2020.00148/full (visited on 05/04/2023).

[78] Masato Asahina, Anupama Poudel, and Shigeki Hirano. "Sweating on the palm and sole: physiological and clinical relevance". en. In: *Clinical Autonomic Research* 25.3 (June 2015), pp. 153–159. ISSN: 0959-9851, 1619-1560. DOI: 10.1007/s10286-015-0282-1. URL: http://link.springer.com/10.1007/s10286-015-0282-1 (visited on 09/15/2023).

7

Patients with Chronic Cluster Headache May Show Reduced Activity Energy Expenditure on Ambulatory Wrist Actigraphy Recordings During Daytime Attacks

Chapter 1 discussed the potential of wearable data for the longitudinal monitoring of chronic conditions, such as primary headache disorders, while also highlighting the analytical challenges these studies entail. Chapter 6 built on this by providing a detailed overview of the most pressing data quality issues, accompanied by practical guidelines. To further ground these insights in practice, this chapter and Chapter 8 put these approaches into action through a real-world use case based on a newly developed dataset.

A key initiative of this Dissertation was the (co)creation of the mBrain dataset, a real-world wearable dataset developed through a multidisciplinary collaboration between my research team, PreDict-IDLab and the Department of Neurology at Ghent University Hospital. This chapter presents a first use case, i.e., cluster headache, for which we did qualitative analysis of the dataset, focusing on movement patterns during cluster headache attacks. Due to the low prevalence of chronic cluster headache (CCH) [1], the sample size was limited to four participants. Consequently, we performed an intra-subject analysis, comparing movement during headache periods to matched headache-free periods for each participant. Contrary to the clinically established hypothesis of increased movement during CH attacks, the analysis revealed a reduction

in movement, particularly in the pre-ictal and ictal phases. This finding suggests that clinical knowledge does not always align with real-world observations, emphasizing the need for contextualized analysis in wearable research.

My contributions are achieved by collaborating with a PhD student from the neurology department (Nicolas Vandenbussche, MD), and can be summarized as follows:

- Cocreating the mBrain dataset.
- Formulating hypotheses on movement activity changes during CH attacks.
- Analyzing the data and conducting statistical tests to assess movement differences.
- Interpreting results and hypothesizing underlying causes, particularly in relation to acute treatment effects.

Patients with Chronic Cluster Headache May Show Reduced Activity Energy Expenditure on Ambulatory Wrist Actigraphy Recordings During Daytime Attacks

Nicolas Vandenbussche[1], Jonas Van Der Donckt [1], Mathias De Brouwer, Bram Steenwinckel, Marija Stojchevska, Sofie Van Hoecke, and Koen Pameleire

Published in "Brain and Behavior, Vol. 14, 2024, e3360"

Structured Abstract

Objective To investigate the changes in activity energy expenditure (AEE) throughout daytime cluster headache (CH) attacks in patients with chronic CH and to evaluate the usefulness of actigraphy as a digital biomarker of CH attacks.

Background CH is a primary headache disorder characterized by attacks of severe to very severe unilateral pain (orbital, supraorbital, temporal, or in any combination of these sites), with ipsilateral cranial autonomic symptoms and/or a sense of restlessness or agitation. We hypothesized increased AEE from hyperactivity during attacks measured by actigraphy.

Methods An observational study including patients with chronic CH was conducted. During 21 days, patients wore an actigraphy device on the nondominant wrist and recorded CH attack-related data in a dedicated smartphone application. Accelerometer data were used for the calculation of AEE before and during daytime CH attacks that occurred in ambulatory settings, and without restrictions on acute and preventive headache treatment. We compared the activity and movements during the pre-ictal, ictal, and postictal phases with data from wrist-worn actigraphy with time-concordant intervals during non-headache periods.

Results Four patients provided 34 attacks, of which 15 attacks met the eligibility criteria for further analysis. In contrast with the initial hypothesis of increased energy expenditure during CH attacks, a decrease in movement was observed during the pre-ictal phase (30 min before onset to onset) and during the headache phase. A significant decrease ($p < .01$) in the proportion of high-intensity movement during headache attacks, of which the majority were oxygen-treated, was observed. This trend was less present for low-intensity movements.

Conclusion The unexpected decrease in AEE during the pre-ictal and headache phase of daytime CH attacks in patients with chronic CH under acute and preventive treatment in ambulatory settings has important implications for future research on wrist actigraphy in CH.

Clinical Trials Registration Number: NCT04949204 (www.ClinicalTrials.gov).

[1] Contributed equally

7.1 Introduction

Cluster headache (CH) is a primary headache disorder characterized by attacks of severe to very severe unilateral pain which is orbital, supraorbital, temporal or in any combination of these sites, lasting 15-180 min when untreated [2]. The international classification of headache disorders, third edition (ICHD-3) criteria for CH furthermore require that CH attacks are associated with ipsilateral cranial autonomic symptoms, for example, lacrimation, and/or a sense of restlessness or agitation [2]. Pre-ictal symptoms were only recently studied in detail and are in fact very common; they include both local and general symptoms [3, 4]. Patients are diagnosed with episodic cluster headache (ECH) if they have periods of CH attacks that are separated by pain-free periods lasting at least 3 months, whereas the diagnosis of chronic cluster headache (CCH) is given when CH attacks occur for at least 1 year without a remission period or with remissions lasting less than 3 months [2, 4].

Patient descriptions and clinical observations indicate that hyperactivity and/or increased movement may be an associated symptom during CH attacks. Ekbom was the first to report that patients are unable to sit still during a CH attack, this in contrast to migraine patients experiencing increased pain with head movements [5]. Blau asked patients to act out their behaviors during CH attacks and these included pacing while clutching the head, sitting and rocking backwards and forwards while holding the head, pressing the affected eye or temple, and hitting the forehead [6]. Even violent, destructive behavior and self-inflicted injuries have been described [6, 7, 5, 8]. These behaviors seemed to be related to the severity of the pain (Russell, 1981). More recently, CH pain has been conceptualized as exteroceptive pain with associated fight-flight response including motor restlessness and agitation, in contrast to migraine pain described as visceral pain [9].

Actigraphy is a noninvasive method that utilizes continuous monitoring sensors worn on the body to objectively measure movement and physical activity [10]. Therefore, actigraphy may provide insights into patient behaviors during the pre-ictal, ictal and postictal phases of CH attacks. The primary objective of our research is to investigate the differences in activity energy expenditure (AEE) during CH attacks using a wrist-worn actigraphy device in unconstrained ambulatory environments [11]. In line with the existing medical literature, we hypothesize that AEE measured by wrist actigraphy increases during CH attacks. A secondary objective of the study is to examine AEE changes in the pre-ictal phase. By hypothesizing that restlessness and agitation can be measured using actigraphy, this technique holds promise as a digital biomarker for identifying CH attacks.

7.2 Methods

7.2.1 Participants

Patients with a diagnosis of CCH (ICHD-3 diagnosis 3.1.2), recruited non-consecutively from the tertiary headache clinic of the Ghent University Hospital, and not suffering from any other headache syndrome (except for infrequent tension-type headache), were included in the study.

Inclusion criteria were as follows: age between 18 and 65 years, at least 5 CH attacks per week expected (i.e., during intake not in a CH attack-free period), no significant medical comorbidity interfering with movement and activity, no drug or alcohol abuse, and no use of beta-blockers. Participants also needed to have regular sleep-wake rhythms and were not nighttime shift-workers.

All participants had access to their habitual CH attack treatment of subcutaneous sumatriptan 6 mg or high flow oxygen at 12-15 L/min via a non-rebreathing mask and other acute treatments at their own discretion. There were no restrictions on the use of preventive treatments for CH for the duration of the study.

7.2.2 Wrist-Worn Actigraphy Sensor and Smartphone Application

The Empatica E4® device was used for the actigraphy recordings on the non-dominant wrist [12]. This device is a CE-certified medical-grade wearable that offers physiological data acquisition. It has an internal memory that can store up to 60 h of data, but in this study the data were sent in real-time over a Bluetooth Low Energy connection to a

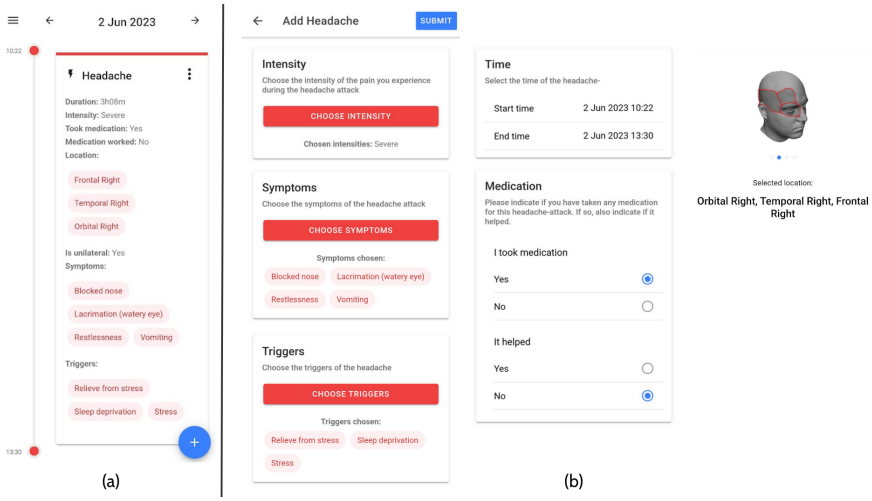


Figure 7.1: Illustration of "mBrain" app timeline view (a) and corresponding headache registration (b).

smartphone that streamed the data to our cloud. Empatica E4® devices have an onboard microelectromechanical system type 3-axis accelerometer that measures continuous gravitational force (g) applied to each of the three spatial dimensions (x, y, and z). The scale is set to ± 2 g. These accelerometer signals are collected at 32 Hz for the three dimensions.

Participants manually documented CH attacks using a dedicated headache smartphone application (the "mBrain" app) developed for this study (Figure 7.1). The recorded CH attack data included the time of onset (hour and minutes) and end (hour and minutes) of each CH attack, the intensity of the CH attack measured on a 5-point Likert Scale ranging from no pain to very severe pain, the use of CH attack medication and the effectiveness of those treatments (part b of Figure 7.1) [13, 14]. Participants were trained during the intake visit to register the details of an experienced CH attack in the application as soon as possible after the CH attack had stopped.

7.2.3 Study Design

The study consisted of a baseline study visit and a final study visit 21 days later. During the study period, participants were instructed to wear the device as much as possible (daytime and nighttime), which ensured consistency in data collection and minimized disruption to daily routines. They were also advised to charge the sensor device at least once a day in the evening. Participants were asked to perform their regular daily activities during the measurement period. However, they were asked to take off the Empatica E4® device when engaging in activities involving water (e.g. showering and swimming), heat (e.g. barbecuing) or cold (e.g. working in freezers).

7.2.4 Absolute Activity Index as a Marker for Activity Energy Expenditure

From the time series data of g-force measurements, the absolute activity index (AI^{ABS}) is calculated and used as the metric of AEE [15]. As shown in Figure 7.2 (a), the AI^{ABS} is formulated as the square root of the mean variance of the raw accelerometer signals, with the variance computed over a fixed period, H. The AI^{ABS} exhibits a robust correlation with activity intensity [15]. Empirical analysis using the Empatica E4® device revealed that the systematic noise variance, σ_i, was negligible, and therefore σ_i was set to 0 for the calculation of the AI^{ABS}, leading to a simplification of the AI^{ABS} formula. In alignment with Bai et al., the AI^{ABS} is computed over a time-period H of 1 s, corresponding to 32 sampling points [15]. Afterward, also in accordance with Bai et al., these second-by-second AI^{ABS} values are aggregated to a per-minute AI, by averaging the 1-s AI^{ABS} values within each 1-min period (Figure 7.2 part b). For time intervals of interest, such as the CH attack interval, the AI^{ABS} values can be represented by a distribution. Considering the non-normal distribution of AI^{ABS}

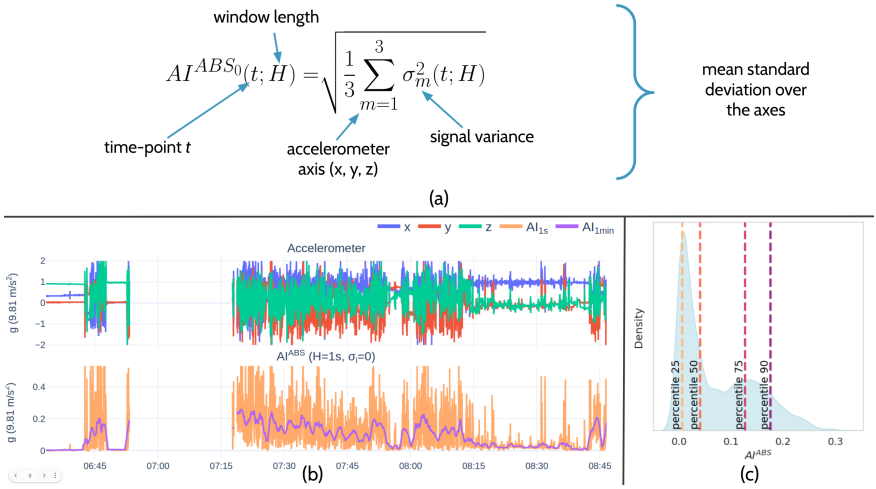


Figure 7.2: Illustration of AI^{ABS} computation and its distribution properties. To calculate the AI^{ABS}, the simplified equation given in (a) was used, which assumes a systematic noise-variance σ_i of zero. Subplot (b) portrays the 1-min aggregated AI^{ABS} for an Empatica E4® accelerometer excerpt. Subfigure (c) demonstrates the distribution and percentiles of the AI^{ABS} of (b). Abbreviations: H = variance window length, t = time point, σ_m = signal variance of wrist acceleration over axis x, y or z.

over these intervals, as depicted in Figure 7.2 (c), we employ a range of quantile-based features for analysis. Specifically, we calculate the median AI^{ABS} (i.e., 50th percentile), as well as the 25th, 75th, and 90th percentiles. We included percentile 90 to examine movement associated with higher energy expenditure, in alignment with our hypothesis.

7.2.5 Eligibility Criteria for the Analysis of CH Attacks and Corresponding Non-CH Intervals

To be eligible for analysis, CH attacks had to fulfill the following criteria. The CH attacks must have occurred during daytime periods without overlap between the headache interval (as documented by the participant during headache registration) and nighttime (between 23h and 7h30). Additionally, a minimum data ratio of 75% during CH attacks was used to ensure a reliable analysis of actigraphy data (i.e., no more than 25% of data missing during the CH attack).

For comparison purposes, intervals without CH attacks from the same participant were used, subject to the following conditions. Non-CH intervals had to be at least 24 h distant from the start and end of CH attacks, and a time range equal to the CH attack was used on the same type of day, either weekdays or during weekends. In cases when multiple eligible non-CH intervals were identified, these intervals were accumulated to construct a single non-CH AI^{ABS} distribution to be paired against the corresponding

CH attack. Data from Belgian holidays were excluded from the analysis.

7.2.6 Analysis and Statistics

Demographic data and participant-specific information (i.e., age, sex, diagnosis, age of onset, and treatment regimens related to CH) are provided in a descriptive manner as proportions and means with standard deviations (SD). Recorded CH attacks are described with mean and SD of CH attack duration (in minutes), mean and SD of CH attack intensity, and the proportion of CH attacks treated with acute therapy, and proportion of acute-treated CH attacks which were successfully managed.

The first analysis assesses AEE during CH attacks by using each CH headache interval as a time range. Specifically, we calculated AI^{ABS} distributions for eligible single CH attacks and their corresponding non-CH data. Percentile AI^{ABS} differences (ΔAI^{ABS}) for single CH attacks were determined by subtracting the CH attack percentile AI^{ABS} value from the non-CH data's percentile value. Furthermore, this analysis also assesses the impact of acute treatment type, which is represented via a color hue.

The second analysis examined AEE distributions for specific time intervals relative to headache onset, including 3-1 h before onset, 1 h to 30 min before onset, 30 min to onset, onset to 30 min after, 30 min to 1 h after, 1-2 h after, and 2-3 h after. Percentile AI^{ABS} interval differences were calculated by subtracting the CH attack interval percentiles from the corresponding non-CH interval percentiles. The number of events may vary for each interval, as the $\geq 75\%$ data ratio must be fulfilled for both CH and non-CH intervals.

Given the high likelihood of non-normality in the separate AI^{ABS} interval distributions as depicted by Figure 7.2 (c), the Wilcoxon signed-rank test was employed to evaluate the statistical significance of AI^{ABS} percentile pairs for CH and their corresponding non-CH intervals. A two-tailed test was used for each of those analyses. Afterward, all results were subjected to Bonferroni correction for multiple testing, therefore, only p-values corrected for multiple testing are presented. Due to the exploratory nature of our pilot study and the lack of previous data, no formal sample size calculation was performed before the onset of the study.

All data processing and transformations were conducted via Python 3.8. Statistical testing was performed using the SciPy package [16]. For exploratory data analysis of the raw wearable data, the Plotly-Resampler tool was used [17]. Via this exploratory analysis, we assessed whether the time-ranges of the headache intervals overlapped with typical sleep periods. Using the 23h - 7h30 nighttime filter, we observed that none of the remaining daytime CH intervals fell within the typical sleep periods of all participants. During the exploratory analysis of the raw wearable data, off-body periods were observed, which refer to intervals where wearable data were present despite the device not being worn [18]. To address this issue, an on-body detection algorithm

Table 7.1: Descriptive characteristics of participants (N=4).

Patient	Diagnosis	Age	Sex	Age at onset CH	Acute treatment	Preventive treatment	Total CH attacks / daytime / … & >=75% data / … & eligible non-CH data
1	CCH	50	Male	36	Oxygen, sumatriptan SC 6 mg	Verapamil 240 mg daily dose	5 / 3 / 3 / 3
2	CCH	64	Male	60	Oxygen, sumatriptan SC 6 mg	Verapamil 480 mg daily dose	14 / 9 / 6 / 6
3	CCH	43	Male	36	Oxygen, sumatriptan SC 6 mg	Verapamil 720 mg daily dose, melatonin 3 mg before bedtime	7 / 1 / 1 / 0
4	CCH	27	Male	15	Oxygen, sumatriptan SC 6 mg, zolmitriptan nasal spray 5mg, paracetamol 250 mg/acetylsalicylic acid 250 mg/caffeine 65 mg	Verapamil 640 mg daily dose, melatonin 5 mg before bedtime	8 / 8 / 6 / 6

was used to identify and exclude off-body periods [19]. This processing step was performed prior to computing the interval data ratios and the AI^{ABS}. In order to efficiently compute the AI^{ABS}, NumPy functions were leveraged through the `tsflex` library [20, 21].

7.2.7 Ethics

This study was approved by the Committee for Medical Ethics of the Ghent University Hospital (internal ID BC-07403, approved June 12, 2020). Patients were fully informed on all aspects of the study (duration, procedures, study visit, etc.) and gave written informed consent at the beginning of the study. Participants received a pseudonymized code throughout the study. Only physician-researchers had the key to decode the participant if required.

7.3 Results

7.3.1 Participants and CH Attacks

Four male participants with CCH were included in the analysis (Table 7.1). All participants had access to high-flow oxygen and/or sumatriptan 6 mg SC injection for the acute treatment of CH attacks. All participants (N = 4) used preventive treatment with verapamil, and two participants used melatonin before bedtime. During the intake discussion, all participants (N = 4) reported experiencing restlessness and/or agitation, and characterized the pain during typical CH attacks as “severe” to “very severe”.

In total, 34 CH attacks were recorded. Fifteen of these attacks from three participants met the daytime and data ratio requirements, while also having eligible non-headache interval(s), and were used for further analysis (seven out of seven CH attacks from one participant could not be used due to six nighttime CH attacks and one CH attack not having an eligible non-CH interval). The mean duration of the analyzed CH attacks was 37 min (SD 25 min) (see Appendix 1). Thirteen out of the

Table 7.2: Descriptive analysis of all cluster headache attacks registered during study.

Description	N	Mean duration (SD) minutes	Mean intensity (SD) *	Acute treatment use (%) / (n/N)	Acute treatment effectiveness (%) / (n/N)
All headaches	34	35 (22)	2.9 (0.7)	82% (28/34)	96% (27/28)
Nighttime headaches	13	32 (21)	3.3 (0.7)	69% (9/13)	100% (9/9)
Daytime headaches	21	37 (23)	2.6 (0.7)	90% (19/21)	95% (18/19)
Daytime headaches w/ ≥75% data during CH attack interval	16	38 (25)	2.6 (0.7)	88% (14/16)	93% (13/14)
Daytime headaches w/ ≥75% data during CH attack interval & eligible non CH attack interval data	15	38 (26)	2.5 (0.6)	87% (13/15)	92% (12/13)

15 CH attacks (87%) were treated with acute therapy: 8 with high-flow oxygen, 2 with combination analgesics paracetamol 250 mg/acetylsalicylic acid 250 mg/caffeine 65 mg, 1 with combination high-flow oxygen and zolmitriptan 5mg nasal spray, and 2 with non-specified treatments. Twelve of these 13 CH attacks were found to be treated effectively by the participants (92%) (Table 7.2). The mean amount of data ratio per individual CH attack was 98.0% (SD 6%). For every analyzed CH attack, the corresponding volume of eligible non-CH data was equal to or higher than the duration of the CH attack itself.

7.3.2 Activity Energy Expenditure During CH Attacks and by Acute Treatment Type

Percentile ΔAI^{ABS} values were calculated for the headache intervals of the 15 qualifying CH attacks, as illustrated in Figure 7.3. When compared to the corresponding non-headache intervals, the AI^{ABS} values during CH attacks show a negative ΔAI^{ABS} trend, indicating a reduction in AEE. This reduction is also represented by a significantly lower 75th percentile (p = 0.007) and 90th percentile (p = 0.0014). No significant differences in AI^{ABS} values were found for the 25th percentile (p = 0.205) and median (p = 0.054) between CH attacks and non-headache intervals) (see Appendix 2). Figure 7.3 also uses color hues to denote the acute treatment type used during a single CH attack. For all four percentiles, all ΔAI^{ABS} values are negative for the CH attacks that were treated with oxygen, indicating no increases in AEE for all of the nine oxygen-treated CH attacks. Appendix 3 presents the statistical significance results exclusively for the oxygen treatment group (N = 9).

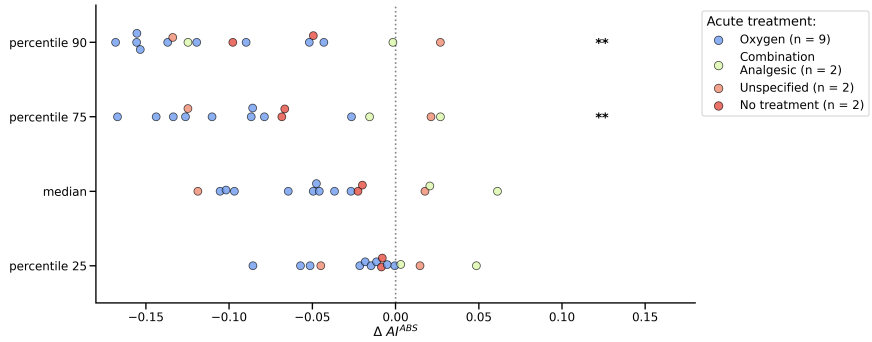


Figure 7.3: Differences in percentile AI^{ABS} (ΔAI^{ABS}) during individual CH attacks and acute treatment type. ΔAI^{ABS} is determined by subtracting the AI^{ABS} percentile value of each CH attack from its corresponding eligible non-CH value (N = 15). Each dot represents the difference for a single CH attack. The acute treatment is represented via a color hue. The treatment types include: no treatment (N = 2), unspecified (N = 2), oxygen (N = 9), and combination analgesic (N = 2). An "unspecified" treatment indicates that the participant reported using medication during the headache event, but no specific medication event was reported afterward. Abbreviations: AI^{ABS} = absolute activity index, ΔAI^{ABS} = difference between absolute activity indices. Levels of statistical significance (after Bonferroni correction for multiple testing): * p < 0.05; ** p < 0.01; *** p < 0.001.

7.3.3 Temporal Patterns of Activity Energy Expenditure Before and During CH Attacks

The decrease of intense movements during headaches, resulting in negative ΔAI^{ABS} values, is observed in Figure 7.3. This decrease is also reflected in Figure 7.4 by the negative yet non-significant ΔAI^{ABS} trend of the "onset to 30 min" interval (N = 14) for the 75th (p =0.215) and 90th percentile (p =0.098). Additionally, Figure 7.4 shows that this negative ΔAI^{ABS} trend is already observed during the pre-ictal phase, specifically for the "-1h to -30 min" (N = 12) and "-30 min to onset" (N = 11) intervals at the 75th and 90th percentiles. The "30 min to 1h" (N=13) interval also exhibits a decreasing, but non-significant, trend. This negative ΔAI^{ABS} trend persists up to 3 h after onset. No statistically significant differences were observed for any of the interval percentiles, which could be attributed to the reduced sample size of the intervals (see Appendix 4). Analysis for the postictal phase can be found in Appendix 5.

7.4 Discussion

To our knowledge, this is the first objective, real-world study utilizing wrist actigraphy to measure the AEE before and during CH attacks in patients with CCH during daytime in the ambulatory setting. Actigraphy in the field of CH has only been used in sleep

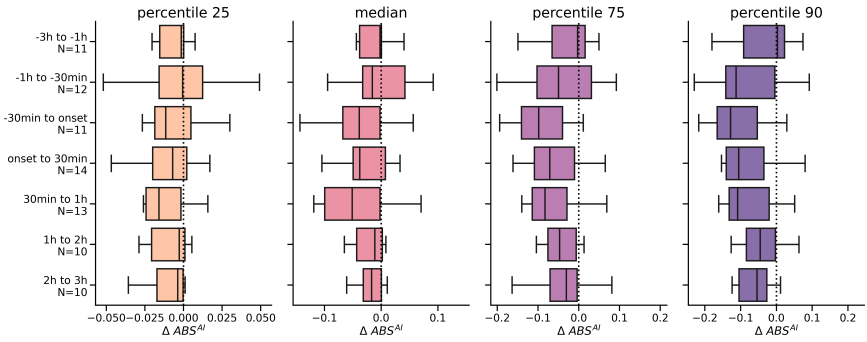


Figure 7.4: Percentile AI^{ABS} differences (ΔAI^{ABS} distributions) for time ranges relative to the headache onsets. The y-axis labels indicate the number of CH attacks considered, which varies due to the $\geq 75\%$ data ratio requirement for both CH attack intervals and non-CH attack intervals. Abbreviations: AI^{ABS} = absolute activity index, ΔAI^{ABS} = difference between absolute activity indices, CH = cluster headache, h = hour, N = number. Levels of statistical significance (after Bonferroni correction for multiple testing): * $p < 0.05$; ** $p < 0.01$; *** $p < 0.001$.

studies so far [22, 23]. Wearable technology has evolved into a low-cost continuous measuring modality providing accurate measurements of activity, movement, and energy expenditure. The technology opens up the potential to more accurately research the behavior of patients in the ictal and interictal phases of headache disorders and holds potential as a digital biomarker for CH attacks.

Our real-world objective actigraphic data challenge the stereotypical idea of behavior during CH attacks in patients using acute treatment. To summarize, our analysis suggests that patients with CCH may show reduced AEE and reduced presence of high-intensity movements not only during the ictal phase but also during the pre-ictal phase and potentially the postdromal phase of CH attacks.

Senses of restlessness and/or agitation are common symptoms of (untreated) CH attacks, hence their integration in the ICHD-3 diagnostic criteria for CH [2, 24]. Almost 70% to almost 90% of patients with CH described typical signs of psychomotor agitation (restlessness) during the CH attack in prospective studies [25]. A prospective clinical survey study in patients with CH found 93% of participants reporting restlessness during the CH attacks or that movement did not exacerbate the pain [26]. Another prospective survey study found that 88.1% of participants with CH exhibited signs of psychomotor agitation (restlessness) with an inability to keep still or performing different types of actions during typical and untreated CH attacks [25]. Previous scientific reports on the clinical features also documented the sense of restlessness or even compulsed movement as a prominent feature of CH [6, 27, 28, 29, 24]. Restlessness or agitation as an associated symptom may even be triggered with calcitonin gene-related peptide or nitroglycerin in laboratory-induced CH attacks [30, 31]. Therefore, restlessness is a highly sensitive and highly specific parameter for CH [24]. Of note that

multiple studies in Asia (including in Taiwan, Japan and Korea) have reported on lower percentages of feelings of restlessness and uncoupling from restless behavior [32, 33, 34].

We infer multiple reasons for our findings of pre-ictal and ictal hypoactivity, which, in turn, create new hypotheses and questions for future research. First, based on the data, the behavioral effects of ictal treatment could force the patient into hypoactivity. This seems especially true for oxygen-treated CH attacks in our dataset, which can be observed in Figure 7.3 and Appendix 3. Due to the specific context of the wrist actigraphy and despite being worn on the non-dominant wrist, it is possible that patients keep their hand and wrist in a more fixed position, for example to hold the oxygen mask or holding their heads with their hands during the CH attacks. It can be speculated that these actions might limit intense hands or body movements. It is plausible that chest-worn or hip-worn actigraphy devices yield different results due to the difference between axial and appendicular motions. Second, although the ICHD-3 criteria require severe to very severe pain for untreated CH attacks, patients in our study reported the severity of the CH attack as moderate, which may affect behavior (see Table 7.2). In our study, no restrictions were imposed on acute treatment. Hence, the capability to manage CH attacks, and the high treatment efficacy, could account for both the moderate CH attack intensity and reduction in AEE. Indeed, patients with CCH may experience within-individual variability in attack severity with attacks of mild to moderate intensity as reported previously by a Danish group [35]. Third, from analyzing the literature, the results may not be completely surprising as previous research efforts may have hinted at reduced AEE in relation to certain CH attacks. Snoer et al. analyzed the presence of symptoms in the pre-ictal, ictal and postictal phase of CH attacks in a prospective, observational questionnaire-based study. Apart from 54% of CH attacks having restlessness as a general symptom, also 30.4% of CH attacks were accompanied by decreased energy levels [3]. The authors also found that the most frequent general symptom in the pre-ictal phase was a sense of restlessness (22.2% of CH attacks), but also that 13.2% of CH attacks had decreased energy levels and 5.6% of CH attacks had fatigue as a pre-ictal symptom [4]. Snoer et al. have also provided good evidence that general symptoms within the pre-ictal phase of CH attacks occurred at a median of 20 min prior to the CH attack, in line with our findings of the most significant drop in AEE between 30 min before onset and onset. Furthermore, this Danish study also found "decreased energy levels" and/or "fatigue" as possible pre-ictal and ictal symptoms in their cohort. Furthermore, for the pre-ictal phase, Blau and Engel already reported in 1998 that several patients may feel "tired", "low", "apathetic", "listless", "withdrawn", "quiet" or "ill" in the pre-ictal phase [36].

We have no doubt that the pain from CH attacks can indeed be excruciating, may lead to a sense of restlessness or agitation, and therefore result in a tendency to pace or move intensely in contrast to the ictal behavior of patients during migraine attacks. This has been documented for many years by patients, clinical experts and

researchers. We are however convinced of the quality of our measured data points, the registrations in the headache applications performed by participants, and the data analysis. Our data, however, suggests that, in real-world settings with unrestricted access to highly efficacious acute treatment for CH attacks, caution is needed when interpreting increased AEE calculated from wrist-worn actigraphy as this may lead to incorrect alarms, misleading results, or over-detection [37, 38]. In the age of rapid access to actigraphy devices, artificial intelligence, and machine learning algorithms, this feature should be addressed carefully when designing data-driven CH detection models for real-world use.

7.4.1 Limitations

Limitations to our study should be addressed. First, the low number of patients in our study and the low number of CH attacks registered may have introduced a sampling error and make it hard to draw definitive conclusions. Generalization to other patients with CH is therefore not possible at the moment, especially patients with the episodic form who were not included in this study. The results are preliminary and part of a larger research effort into the detection of behavioral changes of headache patients before, during, and after CH attacks. Second, our study design utilized streaming data, potentially resulting in fewer headaches, fulfilling the headache data ratio requirement due to missing data. Böttcher et al. observed that using the Empatica E4® in a non-streaming mode generally yields a higher data ratio, but streaming results in a lower user burden as patients do not have to manually download the data from the device and send it to the cloud [19]. Third, we cannot extrapolate our findings immediately to patients with ECH as only CH attacks from patients with longstanding CCH were investigated. Fourth, the study of untreated, and thus more severe CH attacks, most likely yields different data during the headache phase of the CH attack, and preventive treatment may have its influence too. Our real-world observational study design allowed patients to use all types of CH treatments, both acute and preventive, at their discretion as it was deemed unethical to restrict the use of highly effective treatments at this stage of the research. Fifth, we did not collect any momentary assessments of participants' behavior during the CH attacks. Therefore, any potential causal factor for hypoactivity (e.g. oxygen mask holding) relies on indirect evidence retrieved from the actigraphy data and user input into the smartphone application.

7.5 Conclusion

Our findings show that wrist actigraphy for the detection of CH attacks in ambulatory environments with no restrictions on acute treatment may measure reduced activity during CH attacks, in contrast to our initial assumption that patients would show increased AEE due to movement during CH attacks. Our data may provide a method-

ological and hypothesis-generating basis for future real-world actigraphy studies in a larger sample of patients with CH. Further research is required to determine the use of wrist actigraphy for the analysis of movement during CH attacks and as a digital biomarker for CH.

References

[1] M Fischera, M Marziniak, IA Gralow, and S Evers. "The incidence and prevalence of cluster headache: a meta-analysis of population-based studies". In: *Cephalalgia* 28.6 (2008), pp. 614–618.

[2] "Headache Classification Committee of the International Headache Society (IHS) The International Classification of Headache Disorders, 3rd edition". en. In: *Cephalalgia* 38.1 (Jan. 2018), pp. 1–211. ISSN: 0333-1024, 1468-2982. DOI: 10.1177/0333102417738202. URL: http://journals.sagepub.com/doi/10.1177/0333102417738202 (visited on 01/31/2024).

[3] Agneta Snoer, Nunu Lund, Rasmus Beske, Andreas Hagedorn, Rigmor Højland Jensen, and Mads Barloese. "Cluster headache beyond the pain phase: A prospective study of 500 attacks". en. In: *Neurology* 91.9 (Aug. 2018), e822–e831. ISSN: 0028-3878, 1526-632X. DOI: 10.1212/01.wnl.0000542491.92981.03. URL: https://www.neurology.org/lookup/doi/10.1212/01.wnl.0000542491.92981.03 (visited on 06/16/2023).

[4] Agneta Snoer, Nunu Lund, Rasmus Beske, Rigmor Jensen, and Mads Barloese. "Pre-attack signs and symptoms in cluster headache: Characteristics and time profile". en. In: *Cephalalgia* 38.6 (May 2018), pp. 1128–1137. ISSN: 0333-1024, 1468-2982. DOI: 10.1177/0333102417726498. URL: http://journals.sagepub.com/doi/10.1177/0333102417726498 (visited on 06/16/2023).

[5] Karl Ekbom. "A clinical comparison of cluster headache and migraine". In: *Acta Neurologica Scandinavica* (1970), pp. 1+.

[6] J.N. Blau. "Behaviour during a cluster headache". en. In: *The Lancet* 342.8873 (Sept. 1993), pp. 723–725. ISSN: 01406736. DOI: 10.1016/0140-6736(93)91713-V. URL: https://linkinghub.elsevier.com/retrieve/pii/014067369391713V (visited on 06/02/2023).

[7] David W Dodick, RD Brown Jr, JW Britton, and J Huston III. "Nonaneurysmal thunderclap headache with diffuse, multifocal, segmental, and reversible vasospasm". In: *Cephalalgia* 19.2 (1999). Publisher: Wiley Online Library, pp. 118–123.

[8] Arne May. "Cluster headache: pathogenesis, diagnosis, and management". en. In: *The Lancet* 366.9488 (Sept. 2005), pp. 843–855. ISSN: 01406736. DOI: 10.1016/S0140-6736(05)67217-0. URL: https://linkinghub.elsevier.com/retrieve/pii/S0140673605672170 (visited on 06/16/2023).

[9] Pasquale Montagna, Giulia Pierangeli, and Pietro Cortelli. "The Primary Headaches as a Reflection of Genetic Darwinian Adaptive Behavioral Responses". en. In: *Headache: The Journal of Head and Face Pain* 50.2 (Feb. 2010), pp. 273–289. ISSN: 00178748, 15264610. DOI: 10.1111/j.1526-4610.2009.01584.x. URL: https://onlinelibrary.wiley.com/doi/10.1111/j.1526-4610.2009.01584.x (visited on 06/16/2023).

[10] Yumna Saeed, Phyllis C. Zee, and Sabra M. Abbott. "Clinical neurophysiology of circadian rhythm sleep–wake disorders". en. In: *Handbook of Clinical Neurology*. Vol. 161. Elsevier, 2019, pp. 369–380. ISBN: 978-0-444-64142-7. DOI: 10.1016/B978-0-444-64142-7.00061-8. URL: https://linkinghub.elsevier.com/retrieve/pii/B9780444641427000618 (visited on 06/16/2023).

[11] Andrew P. Hills, Najat Mokhtar, and Nuala M. Byrne. "Assessment of Physical Activity and Energy Expenditure: An Overview of Objective Measures". In: *Frontiers in Nutrition* 1 (June 2014). ISSN: 2296-861X. DOI: 10.3389/fnut.2014.00005. URL: http://journal.frontiersin.org/article/10.3389/fnut.2014.00005/abstract (visited on 06/16/2023).

[12] Cameron McCarthy, Nikhilesh Pradhan, Calum Redpath, and Andy Adler. "Validation of the Empatica E4 wristband". In: *2016 IEEE EMBS International Student Conference (ISC)*. May 2016, pp. 1–4. DOI: 10.1109/EMBSISC.2016.7508621.

[13] Mathias De Brouwer et al. "mBrain: towards the continuous follow-up and headache classification of primary headache disorder patients". en. In: *BMC Medical Informatics and Decision Making* 22.1 (Dec. 2022), p. 87. ISSN: 1472-6947. DOI: 10.1186/s12911-022-01813-w. URL: https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/s12911-022-01813-w (visited on 04/08/2022).

[14] Jonas Van Der Donckt, Mathias De Brouwer, Pieter Moens, Marija Stojchevska, Bram Steenwinckel, Stef Pletinck, Nicolas Vandenbussche, Annelis Goris, Koen Paemeleire, Femke Ongenae, et al. "From self-reporting to monitoring for improved migraine management". In: *Engineer meets Physician (EmP)*. 2022.

[15] Jiawei Bai, Chongzhi Di, Luo Xiao, Kelly R. Evenson, Andrea Z. LaCroix, Ciprian M. Crainiceanu, and David M. Buchner. "An Activity Index for Raw Accelerometry Data and Its Comparison with Other Activity Metrics". en. In: *PLOS ONE* 11.8 (Aug. 2016). Ed. by Jaroslaw Harezlak. ZSCC: NoCitationData[s0], e0160644. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0160644. URL: https://dx.plos.org/10.1371/journal.pone.0160644 (visited on 12/16/2021).

[16] SciPy 1.0 Contributors et al. "SciPy 1.0: fundamental algorithms for scientific computing in Python". en. In: *Nature Methods* 17.3 (Mar. 2020), pp. 352–352. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/s41592-020-0772-5. URL: https://www.nature.com/articles/s41592-020-0772-5 (visited on 06/16/2023).

[17] Jonas Van Der Donckt, Jeroen Van Der Donckt, Emiel Deprost, and Sofie Van Hoecke. "Plotly-Resampler: Effective Visual Analytics for Large Time Series". In: *2022 IEEE Visualization and Visual Analytics (VIS)*. Oklahoma City, OK, USA: IEEE, Oct. 2022, pp. 21–25. ISBN: 978-1-66548-812-9. DOI: 10.1109/VIS54862.2022.00013. URL: https://ieeexplore.ieee.org/document/9973221/ (visited on 06/16/2023).

[18] Ann M. Berger, Kimberly K. Wielgus, Stacey Young-McCaughan, Patricia Fischer, Lynne Farr, and Kathryn A. Lee. "Methodological Challenges When Using Actigraphy in Research". en. In: *Journal of Pain and Symptom Management* 36.2 (Aug. 2008), pp. 191–199. ISSN: 08853924. DOI: 10.1016/j.jpainsymman.2007.10.008. URL: https://linkinghub.elsevier.com/retrieve/pii/S0885392408001127 (visited on 06/22/2023).

[19] Sebastian Böttcher et al. "Data quality evaluation in wearable monitoring". en. In: *Scientific Reports* 12.1 (Dec. 2022), p. 21412. ISSN: 2045-2322. DOI: 10.1038/s41598-022-25949-x. URL: https://www.nature.com/articles/s41598-022-25949-x (visited on 12/18/2022).

[20] Charles R. Harris et al. "Array programming with NumPy". en. In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/s41586-020-2649-2. URL: https://www.nature.com/articles/s41586-020-2649-2 (visited on 06/16/2023).

[21] Jonas Van Der Donckt, Jeroen Van Der Donckt, Emiel Deprost, and Sofie Van Hoecke. "tsflex: Flexible time series processing & feature extraction". en. In: *SoftwareX* 17 (Jan. 2022). ZSCC: 0000000, p. 100971. ISSN: 23527110. DOI: 10.1016/j.softx.2021.100971. URL: https://linkinghub.elsevier.com/retrieve/pii/S2352711021001904 (visited on 03/03/2022).

[22] G Della Marca, C Vollono, M Rubino, A Capuano, G Di Trapani, and P Mariotti. "A Sleep Study in Cluster Headache". en. In: *Cephalalgia* 26.3 (Mar. 2006), pp. 290–294. ISSN: 0333-1024, 1468-2982. DOI: 10.1111/j.1468-2982.2005.01037.x. URL: http://journals.sagepub.com/doi/10.1111/j.1468-2982.2005.01037.x (visited on 06/16/2023).

[23] Nunu Lt Lund, Agneta Henriette Snoer, Poul Jørgen Jennum, Rigmor Højland Jensen, and Mads Christian J Barloese. "Sleep in cluster headache revisited: Results from a controlled actigraphic study". en. In: *Cephalalgia* 39.6 (May 2019), pp. 742–749. ISSN: 0333-1024, 1468-2982. DOI: 10.1177/0333102418815506. URL: http://journals.sagepub.com/doi/10.1177/0333102418815506 (visited on 06/16/2023).

[24] Paola Torelli and Gian Camillo Manzoni. "Behavior during cluster headache". en. In: *Current Pain and Headache Reports* 9.2 (Mar. 2005), pp. 113–119. ISSN: 1531-3433, 1534-3081. DOI: 10.1007/s11916-005-0048-x. URL: http://link.springer.com/10.1007/s11916-005-0048-x (visited on 04/26/2023).

[25] Paola Torelli and Gian Camillo Manzoni. "Pain and behaviour in cluster headache. A prospective study and review of the literature". en. In: *Functional Neurology* (2003).

[26] Anish Bahra, Arne May, and Peter J Goadsby. "Cluster headache: a prospective clinical study with diagnostic implications". In: *Neurology* 58.3 (2002). Publisher: AAN Enterprises, pp. 354–361.

[27] Lee Kudrow. *Cluster headache: mechanisms and management*. Oxford University Press, 1980.

[28] Gian Camillo Manzoni, Mario Giovanni Terzano, Giorgio Bono, Giuseppe Micieli, Nicola Martucci, and Giuseppe Nappi. "Cluster Headache — Clinical Findings in 180 Patients". en. In: *Cephalalgia* 3.1 (Mar. 1983), pp. 21–30. ISSN: 0333-1024, 1468-2982. DOI: 10.1046/j.1468-2982.1983.0301021.x. URL: http://journals.sagepub.com/doi/10.1046/j.1468-2982.1983.0301021.x (visited on 04/24/2023).

[29] Ninan T Mathew. "Cluster headache". In: *Seminars in neurology*. Vol. 17. Issue: 04. \copyright 1997 by Thieme Medical Publishers, Inc., 1997, pp. 313–323.

[30] Anne Luise H. Vollesen, Agneta Snoer, Rasmus P. Beske, Song Guo, Jan Hoffmann, Rigmor H. Jensen, and Messoud Ashina. "Effect of Infusion of Calcitonin Gene-Related Peptide on Cluster Headache Attacks: A Randomized Clinical Trial". en. In: *JAMA Neurology* 75.10 (Oct. 2018), p. 1187. ISSN: 2168-6149. DOI: 10.1001/jamaneurol.2018.1675. URL: http://archneur.jamanetwork.com/article.aspx?doi=10.1001/jamaneurol.2018.1675 (visited on 06/16/2023).

[31] Diana Y Wei and Peter J Goadsby. "Comprehensive clinical phenotyping of nitroglycerin infusion induced cluster headache attacks". en. In: *Cephalalgia* 41.8 (July 2021), pp. 913–933. ISSN: 0333-1024, 1468-2982. DOI: 10.1177/0333102421989617. URL: http://journals.sagepub.com/doi/10.1177/0333102421989617 (visited on 06/16/2023).

[32] Noboru Imai, Nobuyasu Yagi, Ryou Kuroda, Takashi Konishi, Masahiro Serizawa, and Masahiro Kobari. "Clinical profile of cluster headaches in Japan: Low prevalence of chronic cluster headache, and uncoupling of sense and behaviour of restlessness". en. In: *Cephalalgia* 31.5 (Apr. 2011), pp. 628–633. ISSN: 0333-1024, 1468-2982. DOI: 10.1177/0333102410391486. URL: http://journals.sagepub.com/doi/10.1177/0333102410391486 (visited on 06/16/2023).

[33] Chien-An Ko, Guan-Yu Lin, Chi-Hsin Ting, Yueh-Feng Sung, Jiunn-Tay Lee, Chia-Kuang Tsai, Chia-Lin Tsai, Yu-Kai Lin, Tsung-Han Ho, and Fu-Chi Yang. "Clinical Features of Cluster Headache: A Hospital-Based Study in Taiwan". In: *Frontiers in Neurology* 12 (Apr. 2021), p. 636888. ISSN: 1664-2295. DOI: 10.3389/fneur.2021.636888. URL: https://www.frontiersin.org/articles/10.3389/fneur.2021.636888/full (visited on 06/16/2023).

[34] Heui-Soo Moon, Jeong Wook Park, Kwang-Soo Lee, Chin-Sang Chung, Byung-Kun Kim, Jae-Moon Kim, Jong-Hee Sohn, Min Kyung Chu, Kyungmi Oh, and Soo-Jin Cho. "Clinical Features of Cluster Headache Patients in Korea". en. In: *Journal of Korean Medical Science* 32.3 (2017), p. 502. ISSN: 1011-8934, 1598-6357. DOI: 10.3346/jkms.2017.32.3.502. URL: https://jkms.org/DOIx.php?id=10.3346/jkms.2017.32.3.502 (visited on 06/16/2023).

[35] Andreas Hagedorn, Agneta Snoer, Rigmor Jensen, Bryan Haddock, and Mads Barloese. "The spectrum of cluster headache: A case report of 4600 attacks". en. In: *Cephalalgia* 39.9 (Aug. 2019), pp. 1134–1142. ISSN: 0333-1024, 1468-2982. DOI: 10.1177/0333102419833081. URL: http://journals.sagepub.com/doi/10.1177/0333102419833081 (visited on 02/23/2024).

[36] Jn Blau and Ho Engel. “Premonitory and Prodromal Symptoms in Cluster Headache”. en. In: *Cephalalgia* 18.2 (Mar. 1998), pp. 91–93. ISSN: 0333-1024, 1468-2982. DOI: 10.1046/j.1468-2982.1998.1802091.x. URL: http://journals.sagepub.com/doi/10.1046/j.1468-2982.1998.1802091.x (visited on 06/16/2023).

[37] Anna S. Cohen, Brian Burns, and Peter J. Goadsby. “High-Flow Oxygen for Treatment of Cluster Headache: A Randomized Trial”. en. In: *JAMA* 302.22 (Dec. 2009), p. 2451. ISSN: 0098-7484. DOI: 10.1001/jama.2009.1855. URL: http://jama.jamanetwork.com/article.aspx?doi=10.1001/jama.2009.1855 (visited on 06/16/2023).

[38] The Sumatriptan Cluster Headache Study Group*. “Treatment of Acute Cluster Headache with Sumatriptan”. en. In: *New England Journal of Medicine* 325.5 (Aug. 1991), pp. 322–326. ISSN: 0028-4793, 1533-4406. DOI: 10.1056/NEJM199108013250505. URL: http://www.nejm.org/doi/10.1056/NEJM199108013250505 (visited on 06/16/2023).

8

Analysis of Free-living Daytime Movement in Patients with Migraine with Access to Acute Treatment

While Chapter 7 focused on movement patterns during cluster headache attacks, this chapter examines a second use case, i.e., headache events within the migraine subgroup of the mBrain dataset. Unlike cluster headaches, which typically last 1–3 hours, migraine attacks persist for 4–72 hours, requiring a more nuanced analytical approach. To ensure meaningful comparisons, this chapter applies data availability criteria and focuses on shorter time intervals rather than the full headache duration.

With a slightly larger sample size, this chapter explores confounding factors that may influence movement behavior, including acute medication use and effectiveness, perceived headache intensity, and the presence of movement-related symptoms. The results confirm a significant reduction in movement during the ictal phase, particularly in patients reporting movement sensitivity. Additionally, movement suppression was more pronounced when acute treatments were ineffective, highlighting the interplay between treatment response and physical activity levels.

My contributions are achieved by collaborating with a PhD student from the neurology department (Nicolas Vandenbussche, MD), and can be summarized as follows:

- Formulating hypotheses regarding movement changes across migraine phases.
- Designing and systematically documenting the data processing and analysis

methodology.

- Interpreting results and hypothesizing underlying causes, particularly in relation to treatment efficacy and headache severity.
- Proposing how the methodology can be adapted for other event-related wearable monitoring studies.

Analysis of Free-living Daytime Movement in Patients with Migraine with Access to Acute Treatment

Jonas Van Der Donckt[1], Nicolas Vandenbussche [1], Mathias De Brouwer, Bram Steenwinckel, Marija Stojchevska, Koen Paemeleire, Femke Ongenae, and Sofie Van Hoecke

Published in "Journal of Headache and Pain, Vol. 26, 2025, Article number 33"

Structured Abstract

Background Motion can exacerbate headache during a migraine attack, potentially leading to avoidance of routine physical activity. Advances in wrist-worn actigraphy facilitate objectively analyzing how headache episodes affect physical activity in everyday settings. The primary hypothesis was hypoactivity during daytime headache events. Secondary hypotheses are hypoactivity during the prodromal and postdromal hours closest to the headache event.

Methods During a 90-day prospective observational study, participants diagnosed with migraine wore an actigraphy device on their non-dominant wrist during daily life and work, while also logging migraine-related data in a dedicated smartphone application. There were no restrictions on use of acute and preventive headache treatments. Data from the wrist-worn accelerometer were used to (i) calculate activity energy expenditure, and (ii) predict types of human activities. These metrics were used to compare daytime prodromal, ictal, and postdromal phases of headache events with time-matched intervals during non-headache periods.

Results A significant reduction in daytime physical activity was observed during the ictal phase of headache attacks, as evidenced by decreases in both activity energy expenditure and human activity recognition prediction metrics. A reduction in movement was also observed during evening hours (18:00–24:00) on headache days. However, no significant physical activity changes were noted in the prodromal and postdromal phases. Reduced physical activity was more pronounced during the ictal phase when acute treatments were ineffective.

Conclusion This study is the first to examine the impact of headache on physical activity levels during daytime headache events by assessing changes in daily activities and activity energy expenditure in individuals with migraine, within their habitual environments and without restrictions on acute medication use. Our findings confirm reduced movement during the ictal phase of migraine attacks, supporting the primary hypothesis. Wrist-worn actigraphy further indicated that this reduction is more pronounced when patients experience movement sensitivity. Evening hypoactivity is also

[1] Contributed equally

observed on headache days. Furthermore, attacks with ineffective acute treatment or moderate-to-high intensity were associated with more pronounced reductions in movement. In contrast, our data did not support the secondary hypothesis that physical activity would decrease during daytime prodromal and postdromal periods.

Trial Registration NCT04983186 (www.ClinicalTrials.gov).

8.1 Background

Migraine is a chronic neurological disorder characterized by recurring headache attacks of moderate to severe intensity, often accompanied by symptoms such as photophobia, phonophobia, nausea or vomiting [1]. Based on scientific knowledge of the disorder, the diagnostic criteria for migraine of the international classification of headache disorders, third edition (ICHD-3), stipulate increased mechanosensitivity as one of the accompanying symptoms of untreated migraine attacks. This sensitivity often results in pain exacerbation or avoidance of routine physical activity [1, 2].

Nosological studies investigating migraine have consistently shown that patients exhibit heightened sensitivity to movements throughout the migraine attack [3, 4]. A study based on questionnaires by Pavao Martins et al. reported that trying to sleep and lying down were among the most common reported coping strategies by patients undergoing migraine attacks [5]. Furthermore, physical activity (PA) is frequently identified as a major precipitating or aggravating factor of migraine attacks [6, 7]. Patients with migraine may also experience kinesiophobia [8]. Even during the interictal periods, studies found that patients with migraine tend to be significantly less physically active than controls and that they reported a significantly lower realizable level of activity [9]. The Nord-Trøndelag health survey (HUNT) questionnaire study found individuals with migraine and non-migraine headaches to be less physically active compared to those without headaches, possibly due to avoidance behavior [10, 11]. Additionally, this HUNT study observed that the frequency of headaches had a greater impact on PA levels than the type of headache. Conversely, several clinical trials suggest that physical exercise may be beneficial for migraine management, demonstrating a reduction in both the frequency and severity of migraine attacks [12, 13, 14]. Pathophysiologically, the heightened sensitivity to movement in migraine sufferers may stem from facilitated activation of nociceptive fibers [15]. Physiological studies from animal experiments show that sensitization of meningeal afferents contributes to intracranial mechanical hypersensitivity, which also explains the clinical observations of exaggerated intracranial mechanosensitivity in humans (e.g., worsening of the pain by coughing, breath-holding, or sudden head movement) [16, 17].

Previous studies already have employed various forms of actigraphy for the study of migraine, with most findings indicating reduced PA during migraine attacks [18, 19, 20]. However, these studies often face limitations such as short observation periods,

inconsistent methodologies, or insufficient sample sizes, which may impact the generalizability and robustness of their conclusions. For instance, Tulen et al. investigated in total eight migraine episodes of moderate to severe intensity across six female participants, using accelerometers temporarily attached to the trunk and upper legs during an intervention at the onset of each migraine episode. Their findings consistently showed decreased body movement during migraine attacks [18]. Furthermore, the study emphasized that factors such as attack severity, acute treatment efficacy, and the time of day could influence behavioral aspects, including the time spent in various body positions, dynamic activities, and the number of postural transitions. Measurements were conducted in the participants' habitual settings, with a two-day headache-free period serving as baseline [18]. The intervention of approaching patients during migraine episodes, however, could influence their behavior or physical activity patterns, potentially confounding the study results. Similarly, Rogers et al. analyzed changes in daily, free-living PA during a seven-day monitoring period across university participants, utilizing a migraine cohort of 28 individuals, and 35 non-headache controls. Using pedometers to approximate PA through step counts, they found that individuals with migraine exhibited lower PA levels compared to non-headache controls, noting decreases in the number of steps taken, even on headache-free days [19]. A digital evening questionnaire was utilized to identify headache days. A seven-day monitoring period may however be insufficient to comprehensively assess the influence of migraines on PA levels, as it may fail to capture multiple migraine episodes, their distinct phases (prodrome, attack, postdrome), or the natural variability in PA patterns across different days and contexts, limiting the generalizability of the findings. Lastly, Kikuchi et al. examined the relationship between tension-type headache (TTH) intensity and PA over a seven-day period using momentary headache intensity measures and wrist-worn actigraphy in a cohort of 31 patients with PA [20]. Their findings revealed a significant negative association between intensity of headache and PA levels.

In summary, many studies have examined PA levels in patients with primary headache disorders, but short monitoring periods and inconsistent findings limit their overall conclusions. For instance, while Rogers et al. observed decreased PA on non-headache days compared to headache days, Tulen et al. found a consistent reduction in PA for their 8 investigated headache episodes. To address these limitations, longitudinal monitoring is needed to enable more granular analyses, such as examining symptom-related effects, medication effectiveness, or headache intensity. Notably, no studies to date have evaluated actigraphy as a tool for quantifying movement-related symptoms like agitation or motion sensitivity in migraine patients.

This study utilizes longitudinal wrist-worn actigraphy to assess PA through two approaches: (i) activity energy expenditure (AEE) calculated from the accelerometer readings, and (ii) human activity recognition predictions generated by a machine learning (ML) model. The actigraphy data is combined with ecological momentary assessment (EMA) data collected via a dedicated smartphone application, enabling par-

ticipants to log headache events and their characteristics in real-time. This integration enhances scalability and supports effective longitudinal monitoring with high temporal granularity of headache events [21, 22]. Wrist-worn actigraphy devices have a high acceptance rate by users due to its ease of wear and non-intrusive design, making it well-suited for longitudinal monitoring of PA levels [23]. However, actigraphy devices worn on non-limb locations, such as the hip or chest, are less susceptible to noise and may correlate more strongly with calorimetry, resulting in a trade-off between accuracy and user compliance [24].

The primary aim of this observational study is to analyze PA levels in patients with chronic or episodic migraine, as defined by ICHD-3, using wrist-worn accelerometers. The primary hypothesis is that longitudinal monitored wrist-worn actigraphy data shows reduced PA during the ictal phase of headache events compared to non-migraine periods within the same individuals. Secondary hypotheses propose that PA descreases in the prodromal phase in close relation to the onset of the headache event, and reduced PA in the postdromal phase. This analysis leverages longitudinal actigraphy recordings (±90 days) and accompanying EMA registrations to capture detailed headache event characteristics. PA changes are assessed by (i) analyzing specific daytime headache events, (ii) performing time-of-day-based analyses on migraine days, and (iii) examining prodromal and postdromal phases of daytime headache events. This work explores the potential of wrist-worn actigraphy as a foundation for developing a digital biomarker for migraine attacks.

8.2 Methods

This analysis was part of the mBrain21 study, a comprehensive and longitudinal research initiative as detailed in De Brouwer et al. [22]. The goal of the mBrain21 study was to provide a profound understanding of migraine manifestations within ambulatory environments. To achieve this, study participants were equipped with wrist-worn wearable devices, specifically the Empatica E4®, for on average 90 days, to gather relevant physiological data. The physiological modalities acquired via the E4 device include skin temperature (4Hz), skin conductance (4Hz), acceleration (32Hz), and blood volume pulse (64Hz) from which the heart rate is derived. The Empatica devices were connected via Bluetooth to a dedicated headache diary smartphone application we developed. Through this application, shown in Figure 8.1, participants could log their headache occurrences, characteristics (intensity, location, medication usage and effectiveness), and associated symptoms consistent with ICHD-3 criteria. In addition, specific medication usage, responses to daily morning and evening questionnaires, and behavioral actions including food intake moments are also questioned [22]. Meanwhile, other behaviors, such as physical activities, sleep, and stress-related events, were automatically inferred from the wearable and smartphone sensor data. This integrated setup enabled continuous, real-time, and objective data collection under free-living

conditions over an extended period.

8.2.1 Participants

Study participants were eligible for the study if they were aged 18 to 65, had a migraine history of more than one year as diagnosed by neurologists specializing in headache disorders (NV, KP), and met the ICHD-3 criteria for episodic or chronic migraine. Additional requirements included experiencing at least five days per month free from headaches or related symptoms, migraine onset before age 50, and owning a smartphone with Android Operating System version 8.0 or higher.

Exclusion criteria encompassed any diagnosed headache disorder other than TTH, medication-overuse headaches (as defined by ICHD-3 8.2 and its subsections), daily headaches without pain-free moments, presence of other chronic pain syndromes, significant medical comorbidity that could affect the study outcomes, opioid or barbiturate use, illicit drug or alcohol abuse, or current/planned pregnancy. Participants could also not simultaneously participate in any other academic or commercial medical study.

8.2.2 Study Design

Participants were recruited between July 2021 and August 2023. The study involved two in-hospital visits: an initial baseline visit and a final study visit after 90 days.

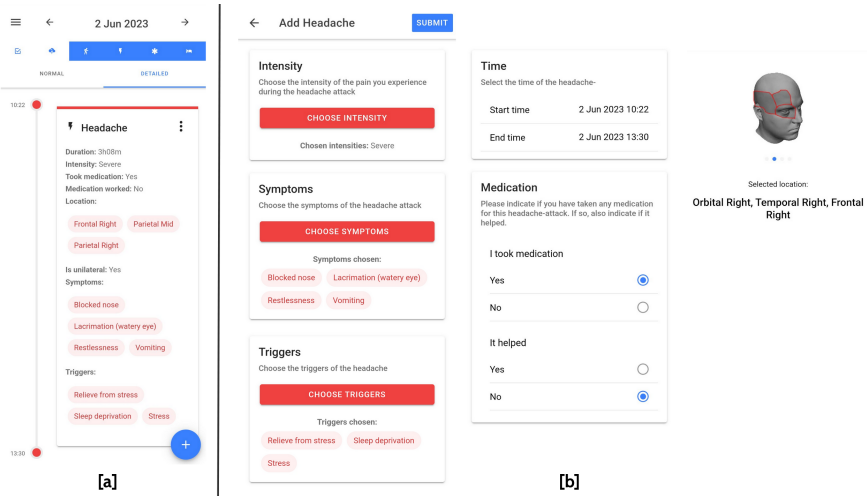


Figure 8.1: Headache registration interface of the mBrain21 phone application. Panel [a] shows the overview of headache events registered in the timeline view of the application. Panel [b] visualizes the headache registry module with different steps to be completed.

During the baseline visit, all study participants were interviewed by a physician researcher and neurologist with expertise in the field of clinical headache disorders (NV). Participants received instructions on how to wear the Empatica E4® actigraph on their non-dominant wrist, connecting the device to their personal smartphone via Bluetooth, and utilizing the accompanying smartphone application for registering headache events and completing daily morning and evening questionnaires.

During the study period, participants were instructed to wear the device as much as possible (daytime and nighttime), to ensure consistent data collection and minimal disruption to their daily routines. They were also advised to charge the wearable at least once a day, ideally in the evening, before bedtime. Participants were encouraged to perform their regular daily activities during the measurement period. However, they were asked to take off the Empatica E4 device when engaging in activities involving water (e.g., showering and swimming), heat (e.g., barbecuing), or cold (e.g., working in freezers).

8.2.3 Data Processing

8.2.3.1 Wrist-Worn Accelerometer Data Processing

In this section, we describe the methodology, illustrated in Figure 8.2, for converting raw accelerometer data from the Empatica device (measured in gravitational (g) units) into delta features for our analysis.

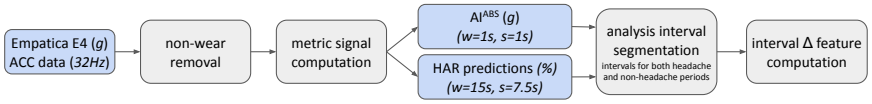


Figure 8.2: Flowchart of the wearable data processing pipeline. *Abbreviations: g = gravitational unit, ACC = accelerometer, Hz = Hertz; w = window, s = stride, AI^{ABS} = absolute activity index, Δ = delta.*

The first processing step involves identifying and eliminating periods of wearable non-wear. These are intervals where the wrist-worn device is not being worn, but continues to record data. Addressing non-wear periods is a recognized challenge in real-world actigraphy research [25]. To address this, a custom algorithm was developed that identifies non-wear periods by analyzing the device's movement, skin temperature, and skin conductance signals, and combines the signal quality indices (SQIs) of these individual signals to create an on-body status signal. Further details can be found in Van Der Donckt et al. (2024) [26]. Figure 8.3 provides a visual example of the resulting output of the employed non-wear detection algorithm applied to the physiological signals captured by the wearable.

After excluding non-wear intervals, the remaining accelerometer data is transformed into two distinct signal groups. The first group consists of the Absolute Activity Index

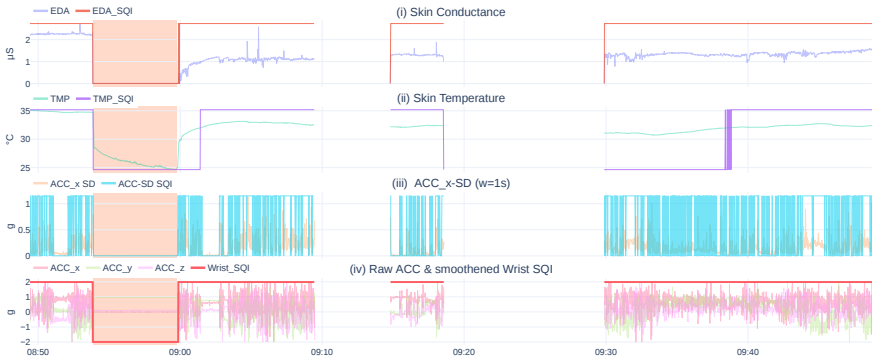


Figure 8.3: Visual overview of the non-wear detection algorithm used on an Empatica E4 excerpt, derived from Van Der Donckt et al. [26]. The red-shaded area in each subplot highlights a manually labeled non-wear interval. Subplots (i) and (ii) display signal-specific signal quality indices (SQIs) for the skin conductance ("EDA") and temperature ("TMP"), respectively. Subplot (iii) shows the standard deviation of the ACC x-axis and the corresponding ACC-SD SQI. Subplot (iv) presents the raw three-axis accelerometer data along with the resulting "Wrist_SQI", which combines the signal-specific SQIs from the above subplots. A low Wrist_SQI value between 08:55 and 09:00 denotes non-wear. *Abbreviations: ACC = accelerometer; SD = standard deviation; SQI = signal quality index, EDA = electrodermal activity, TMP = skin temperature.*

(AI^{ABS}), calculated using a 1-second window and 1-second stride, as recommended by Bai et al. [27]. This signal approximates activity energy expenditure (AEE). For a visual overview and further details on the AI^{ABS} transformation, we refer to Figure 8.4 and the work of Vandenbussche et al. [28]. The second signal group is derived from a human activity recognition (HAR) ML model, which uses a 15-second window and 7.5-second stride – the default configuration for this model. The utilized HAR ML model was specifically developed for the mBrain21 study, and was trained on an in-house dataset, as described in Van Der Donckt et al. (2022) [29]. At each stride step (7.5 seconds), the model predicts the probabilities of six distinct activity types: ["Lying", "Sitting", "Standing", "Walking", "Running", "Cycling"], see subplot b (III) of Figure 8.4.

The HAR analysis in this work primarily focuses on two key activity classifications: "Lying" and "Walking". These activities were chosen for several reasons (i) the HAR ML demonstrates a high validation accuracy for "Walking", and previous research has identified "Lying" as a common behavioral response during migraine headaches [5], (ii) "Walking" is more commonly performed by the general population throughout the day (unlike "Cycling" and "Running"), and (iii) activities such as "Standing" and "Sitting" may cover a wide spectrum of movement intensities and behaviors resulting in varying AEE, which may obscure the results [30]. As such, we hypothesize that differences in movement behavior are most likely to be observed in the "Lying" and "Walking" activities. Additionally, we introduce a "Movement Ratio" signal which aggregates the

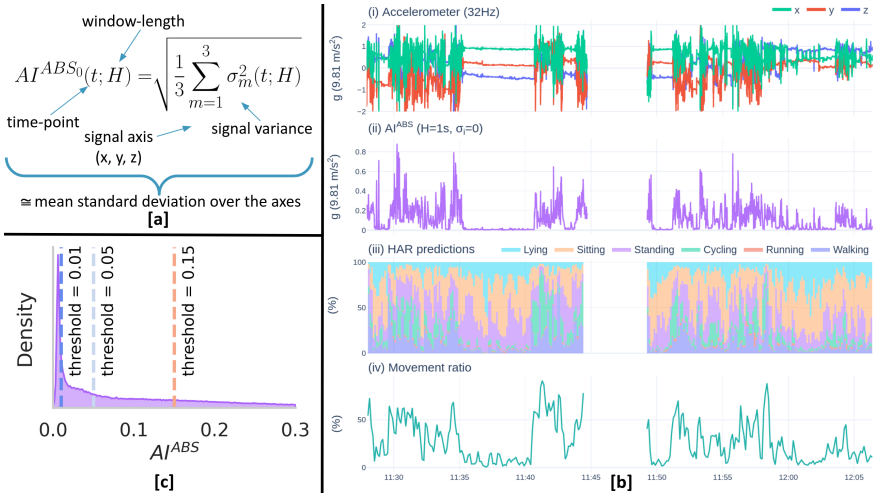


Figure 8.4: Detailed overview of AI^{ABS} computation, its distribution characteristics, and HAR predictions with the derived movement ratio. To calculate the AI^{ABS}, the simplified equation given in panel [a] was used, which assumes a systematic noise-variance σ_i of zero. In panel [b] an Empatica E4 accelerometer excerpt of a study participant (i) is transformed into AI^{ABS} values (with window-length $\mathcal{H}$ = 1 second) (ii). Panel [c] displays the distribution of the AI^{ABS} values of subplot (ii) alongside the threshold values, whose ratio features are utilized for the eventual analysis. In addition, panel [b] demonstrates in subplot (iii) the HAR ML activity prediction probabilities of each class. Lastly, subplot (iv) visualizes the derived "movement ratio" signal from these HAR ML predictions. *Abbreviations:* AI^{ABS} *= absolute activity index;* σ *= variance window length; Hz = Hertz, HAR = human activity recognition; ML = machine learning; t = time point;* σ_m *= signal variance of wrist acceleration over axis x, y or z.*

prediction probabilities for "Walking," "Running," and "Cycling" at each time step, serving as a comprehensive measure of all global body movement activities. Figure 8.4, panel [b], displays the HAR prediction signals and the "Movement Ratio" signal in subplots (iii) and (iv), respectively.

8.2.3.2 Eligibility Criteria for Analysis of Headaches and Corresponding Non-headache Periods

To ensure accurate analysis of the registered headache periods, we utilized the entry time metadata—the timestamp when the event was logged in the mobile application—to identify potential reporting biases. Headache events were flagged for a high probability of recall bias if they were logged more than 24 hours after the reported end time. Similarly, events were flagged for a high probability of predictive bias if their final update occurred more than two hours before the reported end time. Headache records that were flagged for any of the two criteria were excluded from all subsequent analyses. Table 8.5 shows the number of remaining bias-free headache events after this exclusion

process.

For the intervals deemed eligible for analysis, a signal data availability threshold of 95% was applied, which was determined using the methodology outlined in Van Der Donckt et al. (2024) [26]. This threshold strikes a pragmatic balance between metric stability (affected by missing data) and sample retention. Supplementary Figure 1 provides an overview of the distribution spread of signal metric features across various data retention ratios.

Non-headache periods were matched to each bias-free headache period based on specific criteria. To factor out any implicitness, non-headache periods were selected only from days explicitly reported as headache-free in the following morning's questionnaire. These non-headache periods (i) intersect with the time interval of the corresponding headache period, (ii) must contain wearable data (regardless of the amount as they will be concatenated further on and the above outlined availability ratio criteria will be applied on that), (iii) occur on the same type of day (i.e., weekday or weekend) as the headache period, and (iv) fall within 14 days before or after of the headache period. This 14-day proximity threshold was chosen to increase the likelihood of similar time-of-day behavior patterns, as movement patterns are typically more consistent on days close to each other [31]. Furthermore, to minimize the impact of prodromal and postdromal symptoms, non-headache periods are required to be at least 24 hours distant from the end of previous headaches and the beginning of a future headache. Daytime periods are defined as the interval between 8h30 and 22h30.

We are aware that prodromal or postdromal phases may overlap with our 24-hour criterion for non-headache period eligibility, as some studies report prodromal durations of up to 72 hours [31]. Conversely, Kelman L. (2004) [32] found that only 13.2% of 893 migraine patients experienced prodromal durations lasting over 12 hours. Blau J.N. (1980) [33] noted prodromal phases lasting up to 24 hours in a study of 50 participants. Regarding postdrome duration, Kelman L. (2005) [34] observed that 88% of postdromes in 827 headache clinic patients lasted less than 24 hours, and Blau J.N. (1991) [35] reported an average postdrome duration of 18 hours across 40 patients with migraine. Given these findings, this duration threshold was chosen to ensure a substantial number of closely occurring non-headache periods, making it a pragmatic choice for our study.

Additionally, headache and non-headache periods overlapping with Belgian public holidays were filtered out to avoid the potential impact of atypical behavior during holidays.

When multiple eligible non-headache periods were identified for a single headache period, the metric signals from these intervals were concatenated to create a single non-headache metric distribution. Note that the 95% data availability ratio was applied to the concatenation of the eligible non-headache intervals rather than to each individual interval.

From this concatenated distribution, non-headache features were computed and

paired with corresponding features derived from the accompanying headache attacks. Table 8.1 summarizes the features that are computed for each interval. Supplemental Figure 2 illustrates the empirically determined AI^{ABS} threshold ratio features, presenting the AI^{ABS} distribution and the threshold ratios across the different activities as predicted by the HAR model.

Table 8.1: Overview of utilized features for each metric signal and their descriptions.

Metric signal	Feature name	Description
AI^{ABS}	BTR-0.01(*)	Below Threshold Ratio, using a threshold AI^{ABS} value of 0.01. The proportion of the AI^{ABS} signal interval distribution that lies below an AI^{ABS} intensity value of 0.01. This metric range allows us to observe changes in low intensity AEE regions.
	TR-0.05-0.15(*)	Between Threshold Ratio, using a lower and upper threshold AI^{ABS} value of 0.05 and 0.15, respectively. This metric range allows us to observe changes in moderate to slightly intense AEE regions.
	ATR-0.15(*)	Above Threshold Ratio, with a threshold AI^{ABS} value of 0.15. The proportion of the signal interval distribution that lies above an AI^{ABS} value of 0.15. This metric range allows us to observe changes in high-intensity AEE regions.
Lying **Walking** **Movement ratio**	mean	The activity's average prediction probability for the interval of interest.

(*) Given the right-skewed distribution of the AI^{ABS} signal (Figure 8.4, panel c), the utilization of threshold ratio-based features facilitates the computation of robust features by effectively accommodating the distribution's asymmetry.
Abbreviations: AEE = activity energy expenditure; AI^{ABS} = absolute activity index; ATR = above threshold ratio; BTR = below threshold ratio

Lastly, feature deltas (Δ) were computed for each feature by subtracting the corresponding non-headache interval feature from those of the headache interval. Consequently, a positive Δ indicates a higher feature value during the headache period, while a negative Δ indicates a lower value.

Data imputation was not applied; instead, features were computed directly from the metric signals, which may include small bouts of missing data (fewer than 5%). This approach was chosen to avoid further complicating the methodology, especially considering the assumptions required for identifying suitable periods for imputation and aggregation in the context of headache events.

Table 8.2 provides an overview of all the applied eligibility criteria for headache and non-headache period selection.

Table 8.2: Overview of headache and non-headache interval eligibility criteria.

	Headache intervals	Non-headache intervals
Event validity criteria	(i) occur during daytime periods: between 8h30 and 22h30 (ii) Predictive and recall bias filters: retain registrations with increased temporal specificity; Predictive bias: last event interaction must occur later than 2h before the reported headache end time. Recall bias: event must be registered <24h after the reported headache end time. (iii) no overlap with a Belgian public holiday	(i) occur at the same time of day as the corresponding headache interval (i.e., the same daytime period) (ii) occur at the same type of day as the headache interval (weekday or weekend) (iii) participants explicitly indicated that no headaches occurred on the interval date period through a morning questionnaire response.(*) (iv) non-headache interval boundaries must be >=24h distant from headache intervals and be <14 days from the headache pair (v) no overlap with a Belgian public holiday
Wearable data validity	(iv) >=95% data availability for the analysis interval	(vi) After stacking all eligible non-headache periods, >=95% data availability for the analysis interval
As all our analyses are focused on daytime periods, each interval lies within a single day, alleviating the need to check for multiple overlapping dates.		

8.2.4 Ethics Approval and Participants' Consent

The study was approved by the Ethics Committee of University Hospital Ghent (BC-10031). The study was preregistered at clinicaltrials.gov (NCT04983186). All participants gave informed consent for the collection, analysis, and publication of their data.

8.2.5 Analysis and Statistics

Demographic and participant-specific data, including age, sex, duration of headache syndrome, and headache treatment regimens, are provided descriptively as proportions and means with standard deviations (SD). The compliance rate with the daily questionnaire, which serves as an indicator of daily app engagement, is expressed as the percentage of days an individual participant filled out either the morning or evening questionnaire. This rate is reported using the median, along with the first (Q1) and third quartile (Q3). Registered headache episodes are described with average duration (in hours and minutes) and attack intensity, both accompanied by their respective SDs. Lastly, the proportion of attacks treated with acute therapy and the proportion of these acute-treated attacks that were successfully managed are described.

Table 8.3: Summary of criteria applied to different analyses.

	(1) Full ictal phase	(2) time-of-day intervals	(3) onset intervals	(4) offset intervals
Headache interval criteria				
(i) daytime criteria	✓	**X**	✓*	✓*
(ii) headache registration bias: Predictive and recall bias filter	✓	✓	✓	✓
(iii) no Belgian holiday	✓	✓	✓	✓
(iv) >95% data availability	✓	✓*	✓*	✓*
Non-headache interval criteria				
(i) same time as headache interval	✓	✓*	✓*	✓*
(ii) same type of day as headache day	✓	✓	✓	✓
(iii) explicit indication of non-headache day	✓	✓	✓	✓
(iv) interval >24h distant to any headache; <14 day distant to headache event pair	✓	✓*	✓*	✓*
(v) no Belgian holiday	✓	✓	✓	✓
(vi) >95% data after stacking	✓	✓*	✓*	✓*

Four main analyses were conducted, each targeting different intervals of the eligible headache events. Table 8.3 provides an overview of the criteria applied to each analysis.

The first analysis examines movement differences during headache attacks, focusing on the entire ictal phase. Specifically, AI^{ABS} threshold ratios and average HAR prediction values were calculated for daytime headache intervals and compared with corresponding non-headache daytime data. Importantly, this analysis only included full headache intervals that did not overlap with nighttime hours (22:00 to 10:30), as also assessing movement differences during nighttime periods might skew the results. Consequently, considering the typical duration of (untreated) migraine attacks (i.e.

4–72 hours) and the frequent onset of headaches in the early morning (Supplemental Figure 3), a limited number of headache instances are retained for this analysis (N=32 out of 505 bias-free headache events). In addition, this full daytime headache duration analysis also examines the impact of acute treatment, reported movement sensitivity symptoms, and headache intensity on changes in PA. Acute treatment use and effectiveness was determined by the input of the participant within the headache registration view of the application, as shown in panel [b] of Figure 8.1. These variables are visually represented using color hues in three supplementary subplots. Movement sensitivity is identified when either "motion sensitivity" or "pain aggravation during routine activity" was reported.

The second analysis categorizes days as headache or non-headache days based on participants' morning questionnaire responses, which queries whether the previous day was headache-free or not. Eligible headache days were required to also contain the onset of at least one bias-free headache event. Days, marked as headache free by the morning questionnaire, were excluded for analysis if they overlap with a registered headache event or fall on the day before or after a headache event's start or end. No further filtering of the headache days was performed, meaning headache days could include one or multiple headache events, with onset times ranging from early morning or late evening. Each day was segmented into eight two-hour intervals: morning (8–10h), pre-noon (10–12h), noon (12–14h), afternoon (14–16h), early evening (16–18h), evening (18h–20h), late evening (20–22h), and night (22h–24h). For each interval on a headache day, corresponding non-headache intervals are identified using the criteria listed in Table 8.2, followed by metric feature Δ computation. This independent evaluation leads to a varying number of attack pairs across intervals. Subgroup analyses (e.g., based on acute treatment) were not performed due to the complexity of attributing specific headache events to entire days, such as determining the minimum overlap between an event and a headache day or accounting for multiple events occurring on the same day.

The third and fourth analyses investigate movement differences during specific time intervals around the onset and end of headaches, respectively. These analyses focus on five one-hour intervals: from three hours before headache onset (prodromal phase) to the second hour of the headache (ictal phase), and from the last two hours of the headache (ictal phase) to three hours after its end (postdromal phase). . These intervals were chosen in an attempt to capture the most pronounced changes in physical activity, hypothesizing that participants might adjust their behavior, such as increasing activity, during the premonitory phase in anticipation of a headache. Expanding these intervals to encompass the full prodromal and postdromal phases would reduce data availability due to strict inclusion criteria (>95% data coverage and alignment with daytime activity. Similar to the previous analyses, delta features for these periods were computed by subtracting features of non-headache intervals from those of corresponding headache intervals. The number of events considered for each interval may vary, as both headache

and non-headache interval pairs must meet the ≥95% data ratio requirement and fall within daytime hours (8h30–22h30). For intervals within the ictal phase (i.e., the one-hour intervals up to 2 hours after headache onset or before its end), only those fully contained within the ictal phase of the corresponding headache were included. This excludes intervals of shorter headaches (< 2 hours) where this condition cannot be met. By independently applying the ≥95% data ratio and daytime criteria to each 1-hour interval pair (as detailed in Table 8.3), we retain approximately 2.5 times more pairs than in our first full headache duration analysis (N=32), while maintaining the same eligibility criteria for headache and non-headache pairs. In alignment with the first analysis, the second, third, and fourth sub-analyses also examine the presence of motion sensitivity symptoms, acute treatment, and headache intensity.

Statistical testing was performed across all four analyses on the paired signal metric features of headache and their corresponding non-headache intervals. Statistical significance is marked with asterisks (*) in the visualizations. Normality testing, conducted using D'Agostino and Pearson's normality test [36], did not reject the null hypothesis of the samples originating from a normal distribution for the different sample sets. Consequently, we employed the paired sample t-test to determine whether the Δ feature values are symmetrically distributed around zero, considering a two-tailed distribution as the alternative hypothesis. Subsequently, Bonferroni correction was applied for each subplot family to adjust for multiple testing. Reflecting the exploratory nature of this study, both unadjusted and adjusted significance values are presented in the visualizations. Additionally, due to this exploratory nature and the absence of prior data, we did not perform a formal sample size calculation before starting the study.

8.2.6 Data Analysis Software and Visualization Tools

All data processing and analysis were conducted using Python version 3.9. Exploratory data analysis of the raw wearable data was performed with Plotly-Resampler [37]. During this exploratory analysis phase, we verified that daytime headaches (adhering to the 22.00–8.30 nighttime filter) did not overlap with participants' typical sleep times, as observed during manual sleep period annotation (see Supplemental Table 1 for sleep period statistics). Statistical testing was conducted using the SciPy library, and the seaborn toolkit was utilized for scientific visualization [38, 39]. To efficiently compute the AIABS, vectorized numPy functions were leveraged through the `tsflex` library [40, 41].

Table 8.4: Demographics and baseline characteristics of participants, n=27.

Age, mean (SD)	37 (13)
Sex, n (%)	Female 21 (78%) / Male 6 (22%)
Average migraine days per month, mean (SD)	7.7 (5.5)
Duration of headache syndrome in years, mean (SD)	18 (11)
Current use of acute treatment, n (%)	26 (96.3%)
Current use of preventive treatment, n (%)	15 (55.6%)
Days in study, median (Q1-Q3)	91 (83-97)
Days with at least 1 daily questionnaire completed (i.e., compliance rate, %), median (Q1-Q3)	76% (37.2-91.6 %)

8.3 Results

8.3.1 Baseline Characteristics

In the study, 27 participants out of 30 enrolled participants registered at least one headache event, reporting a total of 572 attacks. The median frequency was 15 attacks per participant, with an interquartile range of 5.5 to 34. After adjusting for headache registration timing bias (i.e., headache criteria (ii) of Table 8.3), 505 attacks remained for analysis. Table 8.4 outlines the demographic and baseline characteristics of the 27 participants who reported headaches, while Table 8.5 details the characteristics of these headaches.

Table 8.5: Description of headache events. Remark that upright values indicate average values across the entire population, whereas cursive values indicate the mean of descriptive statistics of per-participant values.

		n	Headache event duration in hours and minutes, mean (SD)	Mean intensity (SD)(*)	Acute treatment use (%)	Acute treatment effectiveness (%)
All events		572	7h09 (8h35) 8h01 (6h22)	2.96 (0.76) 2.91 (0.43)	72.0% 72.5%	69.9% 73.8%
Bias free events		505	6h42 (8h16) 7h14 (6h01)	2.96 (0.75) 2.87 (0.45)	71.7% 71.2%	71% 73.7%
Movement sensitivity	No movement sensitivity	404	6h24 (8h13) 7h24 (6h23)	2.87 (0.61) 2.76 (0.45)	69.3% 68.8%	70.0% 69.1%
	Movement sensitivity	96	7h55 (8h31) 8h57 (6h31)	3.35 (0.75) 3.47 (0.53)	83.3% 80.8%	73.5% 78.2%
Acute treatment	No treatment	143	5h55 (5h44) 6h42 (5h32)	2.42 (0.56) 2.30 (0.30)	0%	/
	Effective treatment	257	6h23 (7h41) 8h12 (7h32)	3.13 (0.76) 3.14 (0.45)	100%	100%
	No effective treatment	105	8h34 (11h42) 9h24 (9h17)	3.24 (0.69) 3.02 (0.51)	100%	0%
Intensity group	Below moderate	146	5h53 (6h59) 7h24 (11h40)	1.99 (0.08) 2.00 (0.01)	41.1% 48.1%	78.3% 74.3%
	Moderate	241	6h44 (9h15) 7h31 (6h56)	3 (0) 3 (0)	78.0% 79.8%	70.7% 72.9%
	Above moderate	118	7h37 (7h32) 7h57 (4h44)	4.05 (0.22) 4.02 (0.05)	96.6% 96.8%	67.6% 73.1%

Note (*): Intensity levels: 1 = no pain, 2 = light pain, 3 = moderate pain, 4 = severe pain, 5 = very severe pain.
Abbreviations: n, N = number; SD = standard deviation.

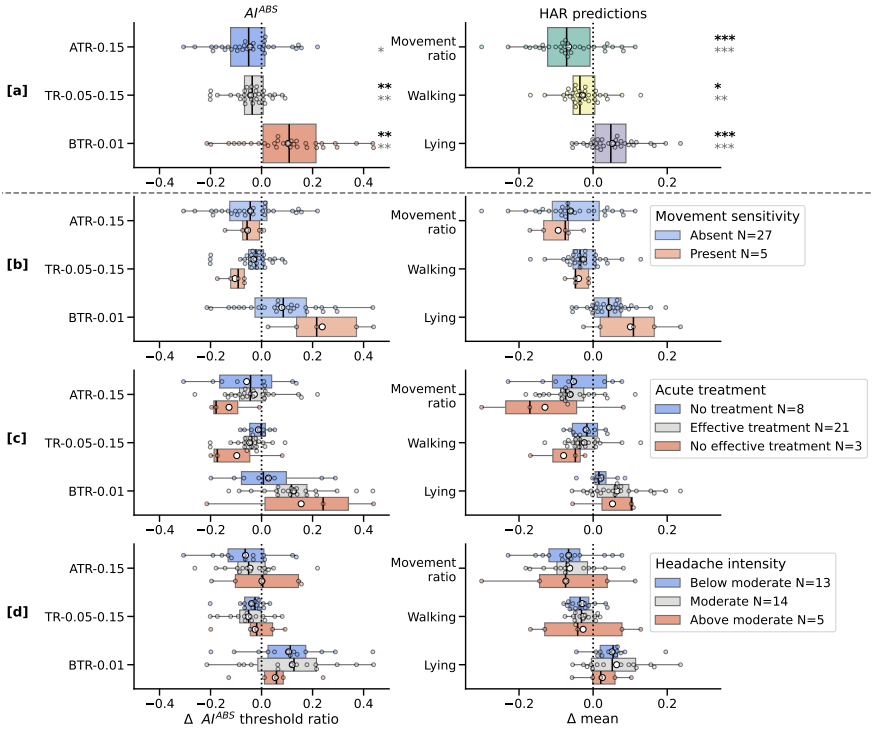


Figure 8.5: Analysis of daytime full headache duration feature deltas, subtracting the values from a corresponding non-headache period from those of the headache period (32 headache events across 9 patients). Each subplot row presents the same data, differentiated by color-coding for various aspects. The first row [a] is color-coded by metric signal feature, row [b] indicates the presence of the movement sensitivity symptom during headache registration, row [c] shows whether acute treatment was used and whether it was effective, and row [d] is color-coded based on experienced headache intensity. Large o-shaped markers in the box plots signify the mean values. Row [a] presents both uncorrected and corrected p-values, displayed with dim gray and bold black asterisks (*), respectively. No statistical testing was performed for rows [b], [c], and [d] due to reduced sample size for the subgroups. Note: P-values: **** = p < 0.0001, *** = p < 0.001, ** = p < 0.01, * = p < 0.05. Abbreviations: AI^{ABS}: absolute activity index; ATR = above threshold ratio; BTR = below threshold ratio; HAR = human activity recognition; TR = threshold ratio; Δ = feature delta.

8.3.2 Analysis 1: Full Headache Duration Analysis

Figure 8.5 presents the feature deltas in full daytime headaches across 32 headache events from 9 participants. Subplot row [a], which does not perform any subgrouping, shows an increase in low-intensity activity periods (AI^{ABS} below threshold 0.01: p < 0.01 post-correction) and a decrease in moderate to high intensity periods (AI^{ABS} between 0.05 and 0.15: p < 0.01 post-correction) during headache events. This observation is further substantiated by the HAR predictions (row [a], second column),

indicating a significant increase in average lying probabilities ($p < 0.001$ post-correction) and a decrease in walking and movement ratio probabilities ($p < 0.05$ and $p < 0.001$ post-correction, respectively). The second subplot row [b] suggests that headache episodes associated with movement sensitivity may correlate with a further reduction in movement, though this observation is based on a small sample size (N=5, 3 participants), limiting its conclusiveness. The third row [c] implies that ineffective acute treatment could result in less intense movements, but this inference is drawn from an even smaller sample (N=3, 1 participant). Subplot row [d] does not exhibit a clear trend in relation to headache intensity, and the small sample size (N=5, 3 participants) for "above moderate" intensity precludes any firm conclusions.

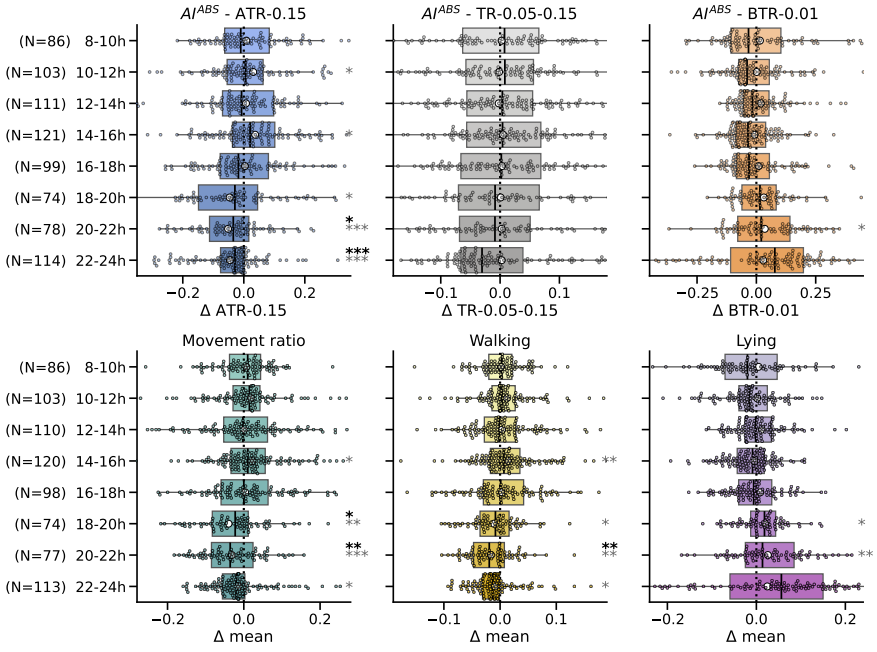


Figure 8.6: Time-of-day interval analysis for AI^{ABS} (row 1) and HAR prediction (row 2) feature Δs, subtracting the values from a corresponding non-headache day from those of the headache day. Both subplot rows utilize identical interval data. Large o-shaped markers in the box plots signify the mean values. The (N=#) prefix for each y-axis label indicates the number of attack pairs for the corresponding interval. Uncorrected and corrected p-values are represented with dim gray and bold black asterisks, respectively. Note: P-values: **** = $p < 0.0001$, *** = $p < 0.001$, ** = $p < 0.01$, * = $p < 0.05$. Abbreviations: AI^{ABS}: absolute activity index; ATR = above threshold ratio; BTR = below threshold ratio; HAR = human activity recognition; TR = threshold ratio; Δ = feature delta.

8.3.3 Analysis 2: Time-of-day Based Analysis

Figure 8.6 examines differences in PA across eight two-hour intervals using six signal metric features. From morning to late afternoon (i.e., 8-10h, 10-12h, 12-14h, 14-16h, 16h-18h), no consistent changes in movement are observed. However, during the early evening interval, ("18-20h") a significant decrease is noted in movement ratio probabilities ($p < 0.01$ post-correction). The late evening interval of "20-22h" shows significant reductions in both movement ratio and walking metrics ($p < 0.01$ post-correction). Visually, the "18-20h", "20-22h", and "22-24h" intervals display a consistent trend of decreased movement in AIABS (row 1) and HAR probability (row 2) features, though not always statistically significant.

8.3.4 Analysis 3: Fixed Intervals Relative to Headache Onset

Figures 8.7 and 8.8 display the feature deltas for 1-hour intervals around headache onset, using AI^{ABS} and HAR prediction signal metrics, respectively.

In subplot row [a] of both figures, no clear trends are observed in the prodromal phase, aside from a slight, non-significant increase in lying probabilities during the "-2h to -1h" and "-1h to onset" intervals. During the first hour of the ictal phase ("onset to 1h"), a significant decrease in movement intensity is observed (AI^{ABS} below threshold 0.01: $p < 0.05$ post-correction). This decrease is also significant for the movement ratio and lying HAR predictions ($p < 0.05$ after correction for both) during the same interval, as shown in Figure 8.8. No significant changes are observed for the "1h to 2h" interval of the ictal phase in either figure.

Subplot row [b] categorizes headache event pairs based on the presence of motion sensitivity as reported symptom. In the movement sensitivity subgroup, both AI^{ABS} (Figure 8.7) and HAR predictions (Figure 8.8) suggest increased movement during the prodromal phase (i.e., "-3h to -2h", "-2h to -1h", and "-1h to onset"), followed by decreased movement during the ictal phase. This decrease during the headache phase aligns with findings from the daytime ictal interval analysis in Figure 8.5. Supplemental Figure 5 provides additional analysis for the motion sensitivity subgroup, showing non-corrected statistical significance for increased movement during the prodromal "-1h to onset" phase.

In subplot row [c] of both figures, medication effectiveness is color-coded. No distinct trends are observed during the prodromal phase, except for a slight reduction in movement ratio for non-effectively treated attacks. However, during the "onset to 1h" interval of the ictal phase, non-effectively treated attacks show a significant decrease in movement intensity, as detailed by Supplemental Figure 6.

Lastly, subplot row [d] in Figure 8.7 and 8.8 shows an increase in movement for above-moderate intensity attacks during the "-3h to -2h" and "-2h to -1h" intervals. This is supported by a significant increase of AI^{ABS} (above threshold 0.15: $p < 0.05$ post-correction) features for the "-2h to -1h" interval, as depicted by Supplementary Figure 7.

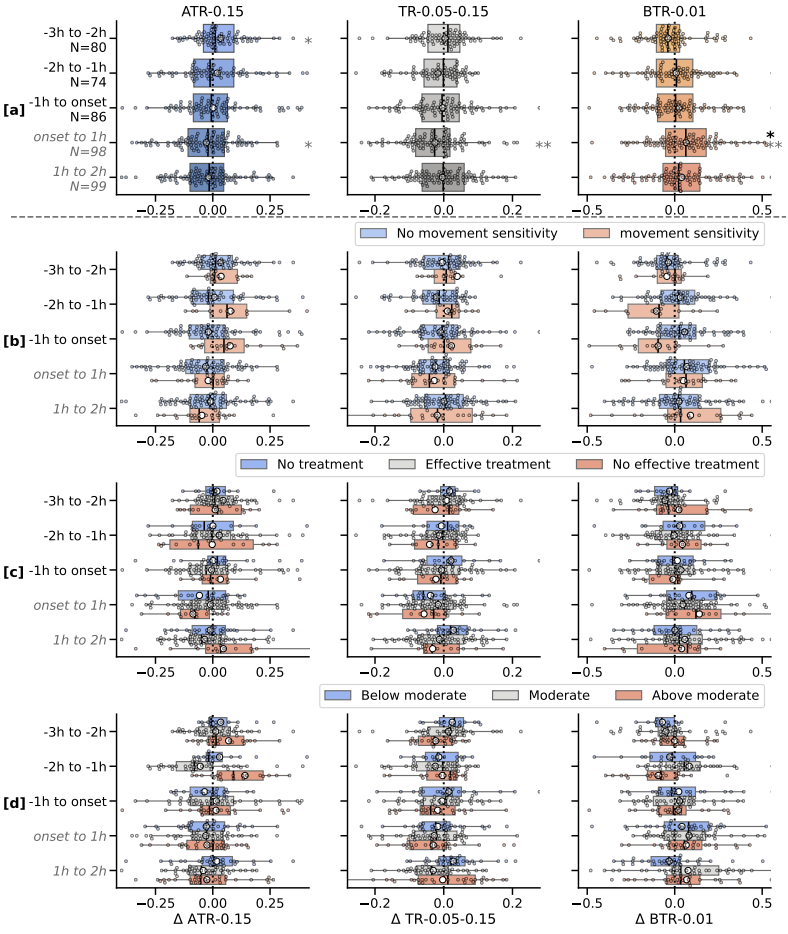


Figure 8.7: Daytime AIABS Δ plots for intervals relative to headache onset, subtracting the values from a corresponding non-headache period from those of the headache period. Each subplot row displays identical data, color-coded by metric signal feature [a], presence of movement sensitivity as a symptom during headache registration [b], acute treatment use and its effectiveness [c], and headache intensity [d]. Large o-shaped markers in the box plots signify the mean values. The first row presents both uncorrected and corrected p-values, displayed with dim gray and bold black asterisks, respectively. Ictal intervals are marked by gray italic y-axis labels. No statistical testing was performed for rows [b], [c], and [d] due to reduced sample size for the subgroups.

For the ictal phase, below-moderate intensity attacks exhibit a slightly smaller decrease in movement compared to the moderate and above-moderate intensity subgroups.

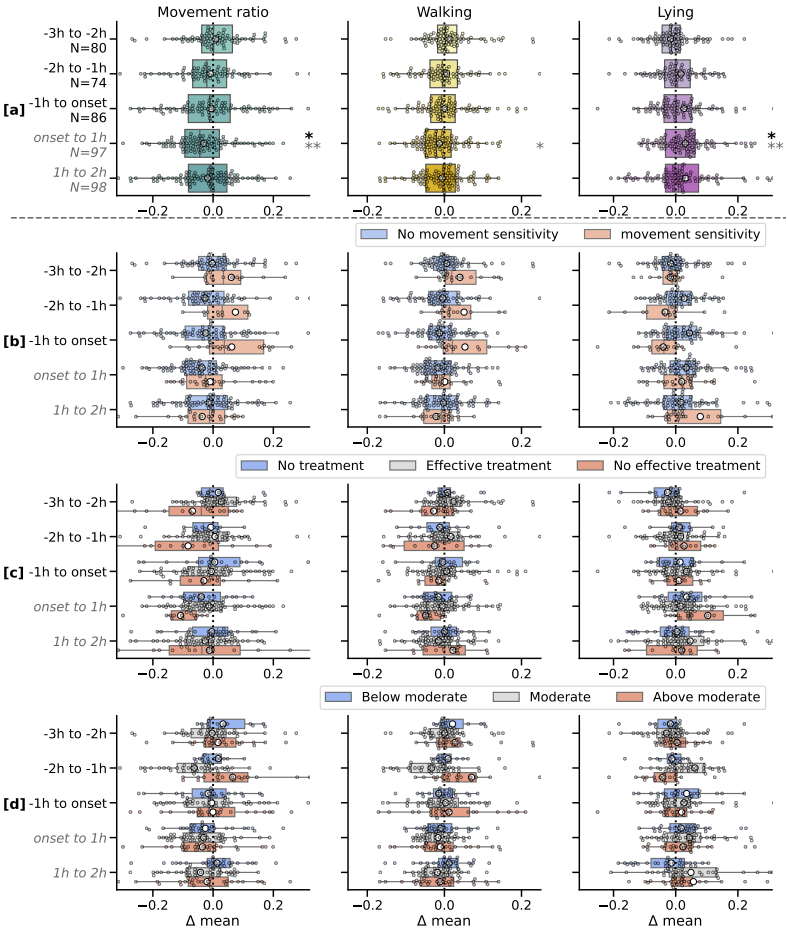


Figure 8.8: Daytime HAR Δ plots for intervals relative to headache onset, subtracting the values from a corresponding non-headache period from those of the headache period. Each subplot row displays identical data, color-coded by [a] metric signal feature, [b] presence of movement sensitivity as a symptom during headache registration, [c] acute treatment use and its effectiveness, and [d] headache intensity. Large o-shaped markers in the box plots signify the mean values. The first row presents both uncorrected and corrected p-values, displayed with dim gray and bold black asterisks, respectively. Ictal intervals are marked with gray italic y-axis labels. No statistical testing was performed for rows [b], [c], and [d] due to reduced sample size for the subgroups.

8.3.5 Analysis 4: Fixed Intervals Relative to Headache End

Figures 8.9 and 8.10 continue the analysis of feature Δs for 1-hour intervals, this time focusing on periods relative to the end of the headache.

Subplot row [a] in both figures highlights a significant reduction in movement during the ictal phase ("-2h to -1h", "-1h to end"). In the postdromal phase, AIABS

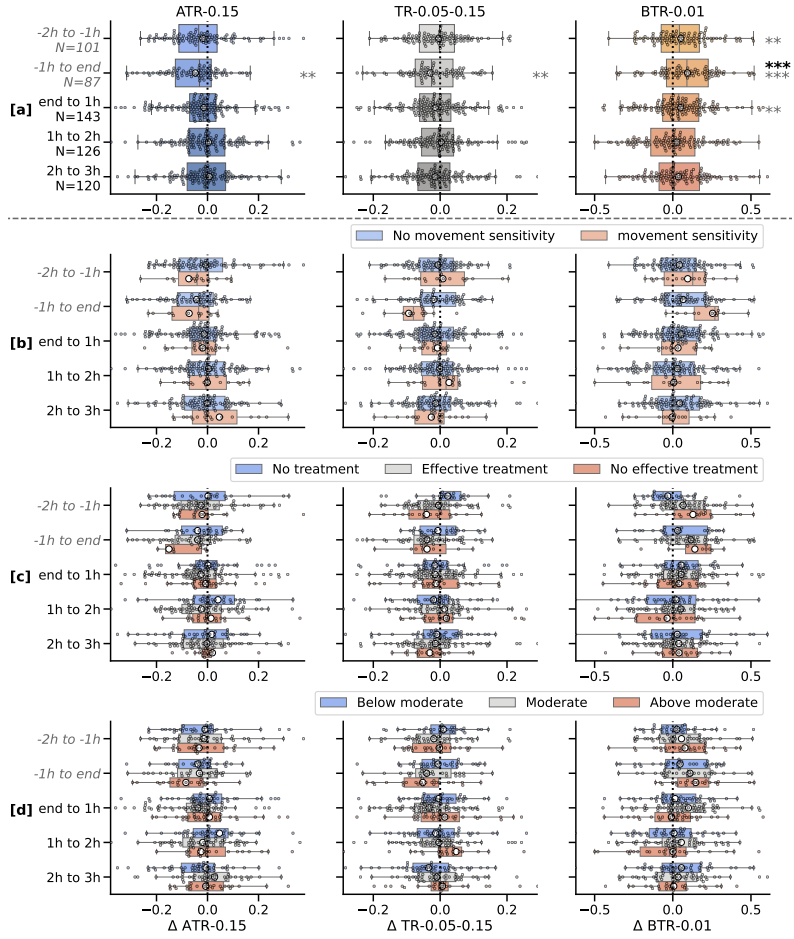


Figure 8.9: Daytime AI^{ABS} Δ plots for intervals relative to headache end, subtracting the values from a corresponding non-headache period from those of the headache period. Each subplot row displays identical data, color-coded by metric signal feature (row [a]), presence of movement sensitivity as a symptom during headache registration (row [b]), acute treatment (row [c]) and headache intensity (row [d]). Large o-shaped markers in the box plots signify the mean values. The first row presents both uncorrected and corrected p-values, displayed with dim gray and bold black asterisks, respectively. Ictal intervals are marked with gray italic y-axis labels. No statistical testing was performed for rows [b], [c], and [d] due to reduced sample size for the subgroups.

features do not indicate significant movement changes, while HAR prediction features indicate a slight decrease in PA. This is reflected by a significant decrease in walking probabilities and an increase in lying probabilities for the “end to 1h” interval ($p < 0.05$ for both post-correction).

Subplot row [b] categorizes the event pairs of [a] based on the presence of mo-

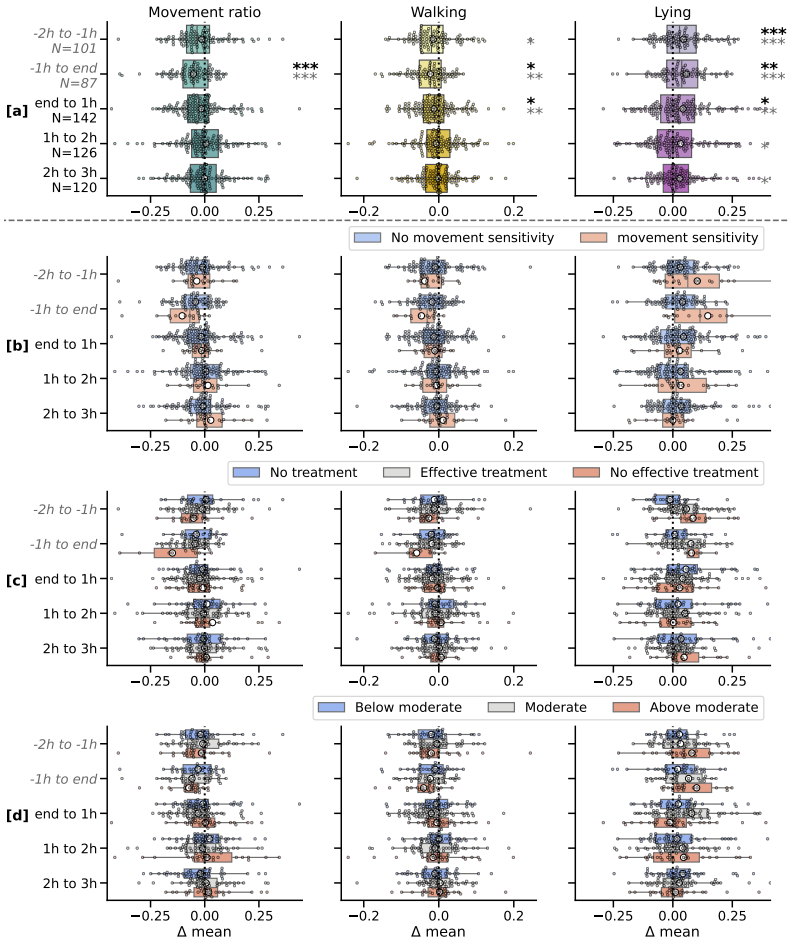


Figure 8.10: Daytime AI^{ABS} Δ plots for intervals relative to headache end, subtracting the values from a corresponding non-headache period from those of the headache period. Each subplot row displays identical data, color-coded by metric signal feature (row [a]), presence of movement sensitivity as a symptom during headache registration (row [b]), acute treatment (row [c]) and headache intensity (row [d]). Large o-shaped markers in the box plots signify the mean values. The first row presents both uncorrected and corrected p-values, displayed with dim gray and bold black asterisks, respectively. Ictal intervals are marked with gray italic y-axis labels. No statistical testing was performed for rows [b], [c], and [d] due to reduced sample size for the subgroups.

tion sensitivity symptoms. In alignment with earlier findings, the motion sensitivity subgroup shows a greater reduction in PA during the ictal phase. Furthermore, this PA reduction for the motion sensitivity subgroup appears to intensify as the phase progresses towards the headache end, as visually highlighted in Supplemental Figure 8. In the postdromal phase, no significant movement differences are observed for either

subgroup, except for a minor increase in movement in the "2h to 3h" interval for the movement sensitivity category.

Subplot Row [c] of Figures 8.9 and 8.10 groups delta features by the effectiveness of acute treatment. The trend remains the same for the subgroup receiving ineffective treatment, showing movement reduction during the ictal phase, with a greater reduction for the non-effective treated subgroups, but no significant PA changes for the groups during the postdromal phase. Supplemental Figure 9 provides a visualization of the ineffective acute treatment group.

The final subplot row [d] provides an overview of the headache intensity categories, revealing a more pronounced reduction in movement during the ictal phase for the above-moderate intensity subgroup, aligning with prior observations. This trend is further detailed in Supplemental Figure 10. The below-moderate subgroup does not show a decrease during the ictal phase. No discernible movement differences are noted in the postdromal phase across intensity subgroups.

8.4 Discussion

Only a few prior actigraphy studies have examined PA levels in patients with headache disorders in free living ambulatory environments [18, 20]. These studies were often limited by short observation periods or inconsistent methodologies, which affected the robustness and generalizability of their findings. Our study addresses these gaps by conducting a longitudinal analysis of daytime PA, evaluating AEE and movement behaviors in patients with migraine within their natural settings, including their habitual use of acute treatments. Using wrist-worn actigraphy combined with a smartphone-based ecological momentary assessment (EMA), we captured high-resolution headache registration data alongside covariates such as movement sensitivity, acute medication usage and headache intensity over a median duration of 91 days. A dual analytical integrated raw accelerometer data (AI^{ABS}) and ML predictions (movement ratio, walking, lying) derived from a HAR algorithm.

The study results demonstrate high consistency between both AI^{ABS} and HAR prediction signal metrics, suggesting that for the tasks presented in the results section, processed data from HAR algorithms may suffice when raw actigraphy data is not available. This reduces the dependence on specialized actigraphy sensors and raw data, potentially lowering costs, enhancing accessibility, and enabling activity monitoring in broader settings, including resource-constrained environments or retrospective studies utilizing existing HAR data.

More importantly, our findings support the primary hypothesis that the daytime ictal phase of migraine attacks are characterized by a significant reduction in intense movement and an increase in low-intensity activities, such as lying down, see Figure 8.5 [a]. However, our secondary hypotheses regarding hypoactivity during the prodromal and the postdromal phase were not supported by the data (Figure 8.7, 8.8, 8.9, 8.10).

Notably, our interval-based analysis in Figure 8.9 [a] and 8.10 [a] highlighted a progressively declining trend in AEE and movement during the ictal phase, with the largest reduction in the "-1h to end" interval.

Several potential explanations may account for the absence of reduced movement during the prodromal and postdromal phases. One possibility is that participants had difficulty distinguishing the prodromal and postdromal phases from the headache phase, which may have led to misclassification or overlap. Additionally, postdromal fatigue could manifest in forms unrelated to gross motor activity, such as cognitive or emotional fatigue, which would not be captured by physical activity metrics. By reviewing the literature, it can also be concluded that either movement sensitivity is an understudied symptom of the premonitory and postdromal phase, or the symptom by nature of the disorder is very infrequently present during these phases [42, 43].

The time of day interval analysis (Figure 8.6) revealed that participants exhibited reduced movement during evening hours (i.e., 18-20h, 20-22h, and 22-24h) on headache days. This outcome aligns with existing evidence suggesting that fatigue or a worn-out effect leads patients to be less active at the end of a headache day. However, considering that the majority of ictal phases conclude in the evening (Supplemental Figure 3) and the most substantial movement reduction is observed during the "-1h to end" interval (Figure 8.9 and 8.10), these findings likely share a period overlap. The insights from Figure 8.6, coupled with the observation that most migraine attacks end in the evening, suggest a potential focus for future research on evening data to enhance headache day detection.

In the subgroup analyses, we discovered that wrist-worn activity devices can detect changes in activity levels for headache events characterized by movement sensitivity (Figure 8.5 [b]). Specifically, during the ictal phase, headaches characterized by motion sensitivity demonstrate a more pronounced decrease in activity compared to those without this symptom (row [b] of Figures 8.9, 8.10). Intriguingly, headaches associated with either motion sensitivity or an above moderate intensity exhibit increased activity during the prodromal phase (row [b] of Figures 8.7, 8.8). Interpretation of this finding is challenging, but several hypotheses arise: 1) a subconscious increase in movement during the prodromal phase, 2) a deliberate increase in activity destined to complete daily tasks before headache onset, or 3) hyperactivity as a trigger for headache events with movement sensitivity. Notably, episodes without motion sensitivity symptom also exhibited reduced PA during the ictal phase (Figure 8.5 [b]). This hypoactivity could be attributed to other migraine-related symptoms, such as fatigue or cognitive disturbances, or it might result from patients' deliberate avoidance of stimulus-rich environments.

Although pain may contribute to reduced activity, we found that the majority of successfully treated headaches still led to reduced activity levels in the ictal phase (row [c] of Figures 8.9-8.10, intervals "-2h to end" and "-1h to end"). However, attacks of below moderate intensity show less significant reductions in PA during the

ictal phase (row [d] of Figures 8.7-8.10). Analysis of the headache intensity levels in Table 8.5 reveals an average intensity of 3.13 (suggesting moderate pain) for the effective treatment group, compared to an average intensity 1.99 (suggesting light pain) for the below moderate subgroup. This suggests that while treatment effectiveness may aid in alleviating the pain (momentarily), perceived headache intensity may be a more primary driver for activity level changes. Additionally, significantly higher headache intensity levels are observed for events associated with movement sensitivity (Mann-Whitney u-test on bias-free headache events, $p<0.001$, $N=505$). Specifically, events with movement sensitivity had an average intensity of 3.35 (moderate-to-severe pain) compared to 2.87 (light-to-moderate pain) for those without this symptom. This heightened intensity may contribute to the increased hypoactivity observed in the motion sensitivity group during the ictal phase.

Headache attacks that were not treated effectively showed a statistically supported reduction in movement during the ictal phase (row [c] of Figure 8.7-8.10), particularly in the intervals immediately preceding the end of the ictal phase (e.g., "-2h to end" and "-1h to end", see Supplemental Figure 9 - "Lying"). While similar trends are visually apparent across other metric signals, the small sample size for this subgroup limits broader generalization. This trend of decreasing PA when nearing the end of the ictal phase may correspond with the buildup of headache pain throughout the ictal phase [15].

Certain interesting sub-analyses, such as grouping attacks based on ICHD-3 migraine criteria, were deliberately not performed. The ICHD-3 diagnostic criteria focus on untreated attacks, which we confound by allowing acute and preventive treatments. Additionally, our focus on daytime intervals and the 95% data availability threshold further reduces the number of attacks qualifying as migraines. Therefore, we refrained from conducting this analysis.

As migraine is a disorder of sensory processing—particularly involving mechanosensitive pain fibers of the trigeminovascular system [44, 45]—the reduced PA observed during the various phases of a migraine attack may stem from several overlapping factors. First example, patients may experience movement sensitivity, which is the aggravation of pain due to otherwise non-painful activity or routine activities. Second, they may withdraw themselves from physically or cognitively demanding tasks to minimize external stimuli. Additionally, rest and sleep are therapeutic for migraine relief, leading to spontaneous hypoactivity as an adaptive behavior. This may be reflected in the hypoactivity seen during the time blocks between 18.00 to 22.00, when fatigue is already building during the evening but is exacerbated by the ictal or postdromal effects of headache attacks (Figure 8.6 and 8.7).

Key strengths of our study include the consistency observed between the AI^{ABS} and HAR analyses, offering diverse methodological approaches towards actigraphy analysis in headache disorders. Moreover, the study's rigorous methodology—characterized by precise definitions of headache episodes and strict data validity criteria—aims to

improve the reliability of our findings, while addressing the inherent challenges of monitoring migraines in real-world contexts. The outlined intra-subject analysis framework is widely applicable to other event-based wearable monitoring studies, such as those investigating epilepsy or mood disorders, as well as studies exploring disorder-related factors like the impact of preventive treatments (see Supplemental Figure 11 for preliminary analysis). These studies help evaluate whether certain symptoms or disorders do have an impact on daily life quality or behavior (such as activity levels). Moreover, with large sample sizes, these studies could analyze extensive datastreams to determine the feasibility of detecting these events in real-world conditions.

Limitations to the study should also be addressed. First, the most important limitation to the study is its exploratory nature with no pre-specified sample size. Given the low sample size and the rigorous preprocessing done to access the most robust parts of the dataset, thereby losing multiple data points during the process (e.g. due to missing data, non-wear periods, connectivity problems etc.), the results of this study should be interpreted as hypothesis-generating rather than directly inferential. Additionally, the cohort was limited to Android users with sufficient technical skills, introducing potential selection bias. Moreover, the study design and inclusion criteria implicitly resulted in the majority of participants being white-collar workers. Second, throughout the data processing methodology, we established our own set of rules and thresholds to ensure data validity and determine eligibility for headache and non-headache intervals. The majority of these values were validated during exploratory analysis and optimized to balance data availability with theoretical validity. While these custom rules were tailored to the study's specific needs, they should be considered when interpreting the results. Third, our non-headache selection procedure allows pairing overlapping non-headache intervals for multiple headache periods. Fourth, it has been demonstrated that wrist-worn actigraphy devices are less representative of global body movements and, consequently, AEE [24]. Although we partially mitigate this limitation by employing a dual metric signal approach, for which the HAR ML predictions estimate global body activities, no direct extrapolation of the results presented above can be made in regard to accelerometers placed anywhere else on the body. Fifth, although our analysis tried to minimize potential recall bias, the start and end times of headache events were collected through self-reporting in the smartphone application, meaning no real-time registration (e.g. via button or command) was utilized. We acknowledge the fact that pain from migraine attacks builds up over time, and so a precise starting point is often obscured. Therefore, the exact start of a headache event is hard to determine with a high temporal accuracy. Moreover, the EMA headache registration requires participants to report a single headache intensity level. This can be problematic as headache intensity may change throughout the ictal phase or due to medication use. Differences in interpretation, with some reporting their average intensity while others record their peak intensity, could further complicate the data. Furthermore, timestamps for the exact time of intake of acute treatment were also unavailable, making analyses

on the effect of acute medication on AEE currently impossible. Additionally, the study lacked information whether participants experienced episodic or chronic migraine, possibly confounding the results. Last, we need to consider the effect of wearable wear bias, implying that participants may tend to wear the wearable less during certain types of activity, potentially skewing the findings. Specifically, since the Empatica E4 device is not waterproof, it cannot be worn during water-related activities, resulting in an incomplete capture of overall behavior.

8.5 Conclusion

This study is one of the first and largest exploratory investigations to objectively analyze daytime movement behavior in patients with migraine within their habitual environments. Our findings demonstrate the feasibility of wrist-worn actigraphy to detect changes in physical activity (PA) during headache events. Specifically, the results support our primary hypothesis of reduced PA during the ictal phase, particularly in cases of unsuccessfully treated headaches and those with reported movement sensitivity. Additionally, we observed decreased PA during evening hours on headache days compared to non-headache days. However, our secondary hypothesis of reduced activity during the prodromal and postdromal periods was not supported by the data. Overall, this study highlights a robust methodological framework for analyzing objective movement data and underscores the potential of wrist-worn actigraphy as a promising digital biomarker for migraine attacks.

Additional Information

Data Availability Statement

The data supporting the findings of this study are not publicly available due to ethical and participant privacy concerns. Data sharing is restricted to comply with the confidentiality agreements made with participants. Researchers requiring access to specific aspects of the data may contact the corresponding author for further discussion. The majority of code and methodology align with examples provided in the authors' analysis methodology paper: github.com/predict-idlab/data-quality-challenges-wearables.

Supplementary Information

All supplementary information is available at GitHub.

References

[1] Marcel Arnold. "Headache Classification Committee of the International Headache Society (IHS) The International Classification of Headache Disorders, 3rd edition". en. In: *Cephalalgia* 38.1 (Jan. 2018), pp. 1–211. ISSN: 0333-1024, 1468-2982. DOI: 10.1177/0333102417738202. URL: http://journals.sagepub.com/doi/10.1177/0333102417738202 (visited on 01/31/2024).

[2] Birthe Krogh Rasmussen, Rigmor Jensen, and Jes Olesen. "A Population-Based Analysis of the Diagnostic Criteria of the International Headache Society". en. In: *Cephalalgia* 11.3 (July 1991), pp. 129–134. ISSN: 0333-1024, 1468-2982. DOI: 10.1046/j.1468-2982.1991.1103129.x. URL: http://journals.sagepub.com/doi/10.1046/j.1468-2982.1991.1103129.x (visited on 01/31/2024).

[3] Jolanta Vanagaite Vingen, Trond Sand, and Lars Jacob Stovner. "Sensitivity to Various Stimuli in Primary Headaches: A Questionnaire Study". en. In: *Headache: The Journal of Head and Face Pain* 39.8 (Sept. 1999), pp. 552–558. ISSN: 0017-8748, 1526-4610. DOI: 10.1046/j.1526-4610.1999.3908552.x. URL: https://headachejournal.onlinelibrary.wiley.com/doi/10.1046/j.1526-4610.1999.3908552.x (visited on 01/31/2024).

[4] Isabel Pavão Martins, Raquel Gil Gouveia, and Elsa Parreira. "Kinesiophobia in Migraine". en. In: *The Journal of Pain* 7.6 (June 2006), pp. 445–451. ISSN: 15265900. DOI: 10.1016/j.jpain.2006.01.449. URL: https://linkinghub.elsevier.com/retrieve/pii/S1526590006005347 (visited on 01/31/2024).

[5] Isabel Pavão Martins and Elsa Parreira. "Behavioral Response to Headache: A Comparison Between Migraine and Tension‐type Headache". en. In: *Headache: The Journal of Head and Face Pain* 41.6 (June 2001), pp. 546–553. ISSN: 0017-8748, 1526-4610. DOI: 10.1046/j.1526-4610.2001.041006546.x. URL: https://headachejournal.onlinelibrary.wiley.com/doi/10.1046/j.1526-4610.2001.041006546.x (visited on 01/31/2024).

[6] Egilius L.H. Spierings, Anniek H. Ranke, and Peter C. Honkoop. "Precipitating and Aggravating Factors of Migraine Versus Tension‐type Headache". en. In: *Headache: The Journal of Head and Face Pain* 41.6 (June 2001), pp. 554–558. ISSN: 0017-8748, 1526-4610. DOI: 10.1046/j.1526-4610.2001.041006554.x. URL: https://headachejournal.onlinelibrary.wiley.com/doi/10.1046/j.1526-4610.2001.041006554.x (visited on 01/31/2024).

[7] L Kelman. "The Triggers or Precipitants of the Acute Migraine Attack". en. In: *Cephalalgia* 27.5 (May 2007), pp. 394–402. ISSN: 0333-1024, 1468-2982. DOI: 10.1111/j.1468-2982.2007.01303.x. URL: http://journals.sagepub.com/doi/10.1111/j.1468-2982.2007.01303.x (visited on 01/31/2024).

[8] Mariana Tedeschi Benatto, Débora Bevilaqua-Grossi, Gabriela Ferreira Carvalho, Marcela Mendes Bragatto, Carina Ferreira Pinheiro, Samuel Straceri Lodovichi, Fabíola Dach, César Fernández-de-las-Peñas, and Lidiane Lima Florencio. "Kinesiophobia Is Associated with Migraine". en. In: *Pain Medicine* 20.4 (Apr. 2019), pp. 846–851. ISSN: 1526-2375, 1526-4637. DOI: 10.1093/pm/pny206. URL:

https://academic.oup.com/painmedicine/article/20/4/846/5193811 (visited on 01/31/2024).

[9] Dl Stronks, Jhm Tulen, Jbj Bussmann, Ljmm Mulder, and J Passchier. "Interictal Daily Functioning in Migraine". en. In: *Cephalalgia* 24.4 (Apr. 2004), pp. 271–279. ISSN: 0333-1024, 1468-2982. DOI: 10.1111/j.1468-2982.2004.00661.x. URL: http://journals.sagepub.com/doi/10.1111/j.1468-2982.2004.00661.x (visited on 01/15/2024).

[10] E Varkey, K Hagen, J-A Zwart, and M Linde. "Physical Activity and Headache: Results from the Nord-Trøndelag Health Study (HUNT)". en. In: *Cephalalgia* 28.12 (Dec. 2008), pp. 1292–1297. ISSN: 0333-1024, 1468-2982. DOI: 10.1111/j.1468-2982.2008.01678.x. URL: http://journals.sagepub.com/doi/10.1111/j.1468-2982.2008.01678.x (visited on 01/31/2024).

[11] L. Robberstad, G. Dyb, K. Hagen, L.J. Stovner, T.L. Holmen, and J.-A. Zwart. "An unfavorable lifestyle and recurrent headaches among adolescents: The HUNT Study". en. In: *Neurology* 75.8 (Aug. 2010), pp. 712–717. ISSN: 0028-3878, 1526-632X. DOI: 10.1212/WNL.0b013e3181eee244. URL: https://www.neurology.org/doi/10.1212/WNL.0b013e3181eee244 (visited on 01/31/2024).

[12] S Osün Narin, L Pinar, D Erbas, V Oztürk, and F Idiman. "The effects of exercise and exercise-related changes in blood nitric oxide level on migraine headache". en. In: *Clinical Rehabilitation* 17.6 (Sept. 2003), pp. 624–630. ISSN: 0269-2155, 1477-0873. DOI: 10.1191/0269215503cr657oa. URL: http://journals.sagepub.com/doi/10.1191/0269215503cr657oa (visited on 01/31/2024).

[13] Lp Queiroz, Mfp Peres, Ej Piovesan, F Kowacs, Mc Ciciarelli, Ja Souza, and E Zukerman. "A Nationwide Population-Based Study of Migraine in Brazil". en. In: *Cephalalgia* 29.6 (June 2009), pp. 642–649. ISSN: 0333-1024, 1468-2982. DOI: 10.1111/j.1468-2982.2008.01782.x. URL: http://journals.sagepub.com/doi/10.1111/j.1468-2982.2008.01782.x (visited on 01/31/2024).

[14] European Headache Federation School of Advanced Studies (EHF-SAS) et al. "The association between migraine and physical exercise". en. In: *The Journal of Headache and Pain* 19.1 (Dec. 2018), p. 83. ISSN: 1129-2369, 1129-2377. DOI: 10.1186/s10194-018-0902-y. URL: https://thejournalofheadacheandpain.biomedcentral.com/articles/10.1186/s10194-018-0902-y (visited on 01/31/2024).

[15] Koen Paemeleire, Nicolas Vandenbussche, and Richard Stark. "Migraine without aura". en. In: *Handbook of Clinical Neurology*. Vol. 198. Elsevier, 2023, pp. 151–167. ISBN: 978-0-12-823356-6. DOI: 10.1016/B978-0-12-823356-6.00007-X. URL: https://linkinghub.elsevier.com/retrieve/pii/B978012823356600007X (visited on 08/08/2024).

[16] A. M. Strassman, S. A. Raymond, and R. Burstein. "Sensitization of meningeal sensory neurons and the origin of headaches". en. In: *Nature* 384.6609 (Dec. 1996), pp. 560–564. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/384560a0. URL: https://www.nature.com/articles/384560a0 (visited on 01/31/2024).

[17] Carolyn Bernstein and Rami Burstein. "Sensitization of the Trigeminovascular Pathway: Perspective and Implications to Migraine Pathophysiology". en. In: *Journal of Clinical Neurology* 8.2 (2012), p. 89. ISSN: 1738-6586, 2005-5013. DOI: 10.3988/jcn.2012.8.2.89. URL: https://www.thejcn.com/DOIx.php?id=10.3988/jcn.2012.8.2.89 (visited on 01/31/2024).

[18] J. H.M. Tulen, D. L. Stronks, J. B.J. Bussmann, L. Pepplinkhuizen, and J. Passchier. "Towards an objective quantitative assessment of daily functioning in migraine: a feasibility study". en. In: *Pain* 86.1 (May 2000), pp. 139–149. ISSN: 0304-3959. DOI: 10.1016/S0304-3959(00)00235-9. URL: https://journals.lww.com/00006396-200005010-00019 (visited on 02/03/2023).

[19] Daniel G. Rogers, Dale S. Bond, John P. Bentley, and Todd A. Smitherman. "Objectively Measured Physical Activity in Migraine as a Function of Headache Activity". en. In: *Headache: The Journal of Head and Face Pain* 60.9 (Oct. 2020), pp. 1930–1938. ISSN: 0017-8748, 1526-4610. DOI: 10.1111/head.13921. URL: https://headachejournal.onlinelibrary.wiley.com/doi/10.1111/head.13921 (visited on 01/31/2024).

[20] H Kikuchi, K Yoshiuchi, K Ohashi, Y Yamamoto, and A Akabayashi. "Tension-Type Headache and Physical Activity: An Actigraphic Study". en. In: *Cephalalgia* 27.11 (Nov. 2007), pp. 1236–1243. ISSN: 0333-1024, 1468-2982. DOI: 10.1111/j.1468-2982.2007.01436.x. URL: http://journals.sagepub.com/doi/10.1111/j.1468-2982.2007.01436.x (visited on 01/31/2024).

[21] Marcella May, Doerte U. Junghaenel, Masakatsu Ono, Arthur A. Stone, and Stefan Schneider. "Ecological Momentary Assessment Methodology in Chronic Pain Research: A Systematic Review". en. In: *The Journal of Pain* 19.7 (July 2018), pp. 699–716. ISSN: 15265900. DOI: 10.1016/j.jpain.2018.01.006. URL: https://linkinghub.elsevier.com/retrieve/pii/S1526590018300300 (visited on 01/31/2024).

[22] Mathias De Brouwer et al. "mBrain: towards the continuous follow-up and headache classification of primary headache disorder patients". en. In: *BMC Medical Informatics and Decision Making* 22.1 (Dec. 2022), p. 87. ISSN: 1472-6947. DOI: 10.1186/s12911-022-01813-w. URL: https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/s12911-022-01813-w (visited on 06/16/2023).

[23] Nicola D. Ridgers and Stuart Fairclough. "Assessing free-living physical activity using accelerometry: Practical issues for researchers and practitioners". en. In: *European Journal of Sport Science* 11.3 (May 2011), pp. 205–213. ISSN: 1746-1391, 1536-7290. DOI: 10.1080/17461391.2010.501116. URL: http://www.tandfonline.com/doi/abs/10.1080/17461391.2010.501116 (visited on 01/31/2024).

[24] Hans Van Remoortel et al. "Validity of Six Activity Monitors in Chronic Obstructive Pulmonary Disease: A Comparison with Indirect Calorimetry". en. In: *PLoS ONE* 7.6 (June 2012). Ed. by Marco Idzko, e39198. ISSN: 1932-6203.

doi: 10.1371/journal.pone.0039198. url: https://dx.plos.org/10.1371/journal.pone.0039198 (visited on 04/26/2023).

[25] Ann M. Berger, Kimberly K. Wielgus, Stacey Young-McCaughan, Patricia Fischer, Lynne Farr, and Kathryn A. Lee. "Methodological Challenges When Using Actigraphy in Research". en. In: *Journal of Pain and Symptom Management* 36.2 (Aug. 2008), pp. 191–199. issn: 08853924. doi: 10.1016/j.jpainsymman.2007.10.008. url: https://linkinghub.elsevier.com/retrieve/pii/S0885392408001127 (visited on 06/22/2023).

[26] Jonas Van Der Donckt, Nicolas Vandenbussche, Jeroen Van Der Donckt, Stephanie Chen, Marija Stojchevska, Mathias De Brouwer, Bram Steenwinckel, Koen Paemeleire, Femke Ongenae, and Sofie Van Hoecke. "Mitigating data quality challenges in ambulatory wrist-worn wearable monitoring through analytical and practical approaches". en. In: *Scientific Reports* 14.1 (July 2024), p. 17545. issn: 2045-2322. doi: 10.1038/s41598-024-67767-3. url: https://www.nature.com/articles/s41598-024-67767-3 (visited on 08/07/2024).

[27] Jiawei Bai, Chongzhi Di, Luo Xiao, Kelly R. Evenson, Andrea Z. LaCroix, Ciprian M. Crainiceanu, and David M. Buchner. "An Activity Index for Raw Accelerometry Data and Its Comparison with Other Activity Metrics". en. In: *PLOS ONE* 11.8 (Aug. 2016). Ed. by Jaroslaw Harezlak. ZSCC: NoCitationData[s0], e0160644. issn: 1932-6203. doi: 10.1371/journal.pone.0160644. url: https://dx.plos.org/10.1371/journal.pone.0160644 (visited on 12/16/2021).

[28] Nicolas Vandenbussche, Jonas Van Der Donckt, Mathias De Brouwer, Bram Steenwinckel, Marija Stojchevska, Femke Ongenae, Sofie Van Hoecke, and Koen Paemeleire. "Patients with chronic cluster headache may show reduced activity energy expenditure on ambulatory wrist actigraphy recordings during daytime attacks". en. In: *Brain and Behavior* 14.1 (Jan. 2024), e3360. issn: 2162-3279, 2162-3279. doi: 10.1002/brb3.3360. url: https://onlinelibrary.wiley.com/doi/10.1002/brb3.3360 (visited on 01/17/2024).

[29] Van Der Donckt, Jonas and De Brouwer, Mathias and Moens, Pieter and Stojchevska, Marija and Steenwinckel, Bram and Pletinck, Stef and Vandenbussche, Nicolas and Goris, Annelis and Paemeleire, Koen and Ongenae, Femke and Van Hoecke, Sofie. "From self-reporting to monitoring for improved migraine management". eng. In: *EmP : 1st RADar conference on Engineer meets Physician, Proceedings*. Place: Roeselare, Belgium. 2022, p. 5.

[30] Mary E. Rosenberger, William L. Haskell, Fahd Albinali, Selene Mota, Jason Nawyn, and Stephen Intille. "Estimating Activity and Sedentary Behavior from an Accelerometer on the Hip or Wrist". en. In: *Medicine & Science in Sports & Exercise* 45.5 (May 2013), pp. 964–975. issn: 0195-9131. doi: 10.1249/MSS.0b013e31827f0d9c. url: https://journals.lww.com/00005768-201305000-00021 (visited on 04/02/2024).

[31] Neeraj Garg, Sanjay K. Dhurandher, Petros Nicopolitidis, and J. S. Lather. "Efficient mobility prediction scheme for pervasive networks". en. In: *International Journal of Communication Systems* 31.6 (Apr. 2018), e3520. ISSN: 1074-5351, 1099-1131. DOI: 10.1002/dac.3520. URL: https://onlinelibrary.wiley.com/doi/10.1002/dac.3520 (visited on 01/17/2024).

[32] Leslie Kelman. "The Premonitory Symptoms (Prodrome): A Tertiary Care Study of 893 Migraineurs". en. In: *Headache: The Journal of Head and Face Pain* 44.9 (Oct. 2004), pp. 865–872. ISSN: 0017-8748, 1526-4610. DOI: 10.1111/j.1526-4610.2004.04168.x. URL: https://headachejournal.onlinelibrary.wiley.com/doi/10.1111/j.1526-4610.2004.04168.x (visited on 01/17/2024).

[33] J N Blau. "Migraine prodromes separated from the aura: complete migraine." en. In: *BMJ* 281.6241 (Sept. 1980), pp. 658–660. ISSN: 0959-8138, 1468-5833. DOI: 10.1136/bmj.281.6241.658. URL: https://www.bmj.com/lookup/doi/10.1136/bmj.281.6241.658 (visited on 01/17/2024).

[34] L Kelman. "The Postdrome of the Acute Migraine Attack". en. In: *Cephalalgia* 26.2 (Feb. 2006), pp. 214–220. ISSN: 0333-1024, 1468-2982. DOI: 10.1111/j.1468-2982.2005.01026.x. URL: http://journals.sagepub.com/doi/10.1111/j.1468-2982.2005.01026.x (visited on 01/17/2024).

[35] Jn Blau. "Migraine Postdromes: Symptoms After Attacks". en. In: *Cephalalgia* 11.5 (Nov. 1991), pp. 228–231. ISSN: 0333-1024, 1468-2982. DOI: 10.1046/j.1468-2982.1991.1105229.x. URL: http://journals.sagepub.com/doi/10.1046/j.1468-2982.1991.1105229.x (visited on 01/17/2024).

[36] Ralph D'Agostino and E. S. Pearson. "Tests for departure from normality. Empirical results for the distributions of b_2 and $\sqrt{b_1}$". en. In: *Biometrika* 60.3 (1973), pp. 613–622. ISSN: 0006-3444, 1464-3510. DOI: 10.1093/biomet/60.3.613. URL: https://academic.oup.com/biomet/article-lookup/doi/10.1093/biomet/60.3.613 (visited on 01/22/2024).

[37] Jonas Van Der Donckt, Jeroen Van Der Donckt, Emiel Deprost, and Sofie Van Hoecke. "Plotly-Resampler: Effective Visual Analytics for Large Time Series". In: *2022 IEEE Visualization and Visual Analytics (VIS)*. Oklahoma City, OK, USA: IEEE, Oct. 2022, pp. 21–25. ISBN: 978-1-66548-812-9. DOI: 10.1109/VIS54862.2022.00013. URL: https://ieeexplore.ieee.org/document/9973221/ (visited on 06/16/2023).

[38] SciPy 1.0 Contributors et al. "SciPy 1.0: fundamental algorithms for scientific computing in Python". en. In: *Nature Methods* 17.3 (Mar. 2020), pp. 352–352. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/s41592-020-0772-5. URL: https://www.nature.com/articles/s41592-020-0772-5 (visited on 06/16/2023).

[39] Michael Waskom. "seaborn: statistical data visualization". In: *Journal of Open Source Software* 6.60 (Apr. 2021), p. 3021. ISSN: 2475-9066. DOI: 10.21105/joss.03021. URL: https://joss.theoj.org/papers/10.21105/joss.03021 (visited on 01/17/2024).

[40] Jonas Van Der Donckt, Jeroen Van Der Donckt, Emiel Deprost, and Sofie Van Hoecke. "tsflex: Flexible time series processing & feature extraction". en. In: *SoftwareX* 17 (Jan. 2022). ZSCC: 0000000, p. 100971. ISSN: 23527110. DOI: 10.1016/j.softx.2021.100971. URL: https://linkinghub.elsevier.com/retrieve/pii/S2352711021001904 (visited on 03/03/2022).

[41] Charles R. Harris et al. "Array programming with NumPy". en. In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/s41586-020-2649-2. URL: https://www.nature.com/articles/s41586-020-2649-2 (visited on 06/16/2023).

[42] Nazia Karsan, Abigail Peréz-Rodríguez, Karthik Nagaraj, Pyari R Bose, and Peter J Goadsby. "The migraine postdrome: Spontaneous and triggered phenotypes". en. In: *Cephalalgia* 41.6 (May 2021), pp. 721–730. ISSN: 0333-1024, 1468-2982. DOI: 10.1177/0333102420975401. URL: https://journals.sagepub.com/doi/10.1177/0333102420975401 (visited on 01/23/2025).

[43] Todd J. Schwedt, Richard B. Lipton, Peter J. Goadsby, Chia-Chun Chiang, Brad C. Klein, Cory Hussar, Chengcheng Liu, Sung Yun Yu, Michelle Finnegan, and Joel M. Trugman. "Characterizing Prodrome (Premonitory Phase) in Migraine: Results From the PRODROME Trial Screening Period". en. In: *Neurology Clinical Practice* 15.1 (Feb. 2025), e200359. ISSN: 2163-0402, 2163-0933. DOI: 10.1212/CPJ.0000000000200359. URL: https://www.neurology.org/doi/10.1212/CPJ.0000000000200359 (visited on 01/23/2025).

[44] Peter J. Goadsby, Philip R. Holland, Margarida Martins-Oliveira, Jan Hoffmann, Christoph Schankin, and Simon Akerman. "Pathophysiology of Migraine: A Disorder of Sensory Processing". en. In: *Physiological Reviews* 97.2 (Apr. 2017), pp. 553–622. ISSN: 0031-9333, 1522-1210. DOI: 10.1152/physrev.00034.2015. URL: https://www.physiology.org/doi/10.1152/physrev.00034.2015 (visited on 02/02/2024).

[45] Adriana Della Pietra, Laura Gómez Dabó, Petr Mikulenka, Christian Espinoza-Vinces, Doga Vuralli, Isil Baytekin, Paolo Martelletti, Rashid Giniatullin, and On behalf of the School of Advanced Studies of the European Headache Federation (EHF-SAS). "Mechanosensitive receptors in migraine: a systematic review". en. In: *The Journal of Headache and Pain* 25.1 (Jan. 2024), p. 6. ISSN: 1129-2377. DOI: 10.1186/s10194-023-01710-1. URL: https://thejournalofheadacheandpain.biomedcentral.com/articles/10.1186/s10194-023-01710-1 (visited on 01/10/2025).

9 Conclusion

"The truth lies in the code."

–Andrej Karpathy

This dissertation has focused on advancing large-scale time series data analytics, addressing challenges from scalable visualization (descriptive and prescriptive analytics) to (re)evaluating the effectiveness of traditional machine learning for time series data, with a particular emphasis on wearable data acquired in real-world environments. This final chapter summarizes the impact of my work and discusses directions for future research and development.

9.1 Summary and Discussion of the Research Findings

RG1: Facilitating scalable time series visualization
The research presented in Chapters 2 and 3 established a foundation for systematically comparing and analyzing line chart data-aggregation methods. The flexible interface of Plotly-Resampler allows users to employ optimized aggregation algorithms, as well as experiment with and implement novel approaches, directly supporting **RG1-A** and **RG1-B**.

Our findings indicated that achieving pixel-perfect aggregation using the M4 algorithm might be challenging in practice, as it requires a complex interplay of various visualization aspects that are not always easy to control. Additionally, we highlighted the critical role of vertical extrema in shape-preserving subsampling algorithms. This

observation is further explored in Appendix A, where we introduced an efficient search space reduction method for the largest triangle three buckets (LTTB) algorithm. This optimization significantly improves performance, achieving a 30× speed-up (with parallelization) compared to standard LTTB, thereby realizing **RG1-C**. Finally, we introduced a methodology for assessing visual stability, providing a foundation for evaluating the effectiveness of data aggregation methods in streaming contexts.

Collectively, these contributions significantly advance the field of scalable time series line chart visualization, thereby achieving RG1.

RG2: Improving traditional time series machine learning
Effectively assessing the potential of traditional time series machine learning requires well-crafted feature engineering and sensible signal processing. To support this, Appendix B introduced `tsflex`, a Python library designed for efficient feature extraction from heterogeneous time series data. Unlike many existing tools, `tsflex` makes no implicit assumptions about data regularity or continuity, enabling robust processing even when data contains gaps. It also provides native support for multi-window feature extraction, a capability that has been largely overlooked in prior work. Through benchmarking, we demonstrated that `tsflex` offers superior efficiency compared to existing libraries, thereby achieving **RG2-A**.

Chapters 4 and 5 presented technical write-ups of two machine learning competitions, in which we secured first place in both by conducting extensive data analyses and introducing novel data transformation methodologies.

The SHL Challenge (Chapter 4) involved classifying transportation modes using 5-second windows of smartphone movement data $\in \{\text{Acc, Gyr, Mag}\}$, with one modality missing at random. Our analysis revealed two key challenges: (1) a distribution shift between the train and test datasets, and (2) increased variability due to the unknown smartphone location during testing. To address these challenges, we designed a machine learning pipeline that focuses on magnitude- and rotation-invariant feature aggregation. To ensure magnitude invariance, we either excluded magnitude-sensitive features or normalized signals prior to feature computation. For orientation invariance, we initially experimented with the signal magnitude vector (SMV) aggregation, but it underperformed compared to rotation-aware features. This led to the development of novel rotation-invariant aggregation techniques that summarize x, y, z signals into statistical representations using three configurations: `stat2` (mean, std), `stat3` (mean, std, skew), and `sort` (min, mid, max). Each configuration outperformed both SMV and rotation-aware methods. Our final pipeline achieved a macro F1 score of 79.1% on the test set, outperforming the second-place solution, a CNN-based approach, by $\pm$ 10% absolute (69.4%). These results are detailed in the challenge organizers' summary paper [1].

The WEAR [2] Challenge (Chapter 5) required classifying 18 workout activities—including transient phases—using continuous data from four wearables (one per limb).

Due to the continuous, unsegmented nature of the recordings, we leveraged our `tsflex` toolkit to extract multi-resolution features.

Exploratory analysis revealed inconsistencies in wearable orientations. To realize rotation invariance, we introduced two novel augmentations:

1. Left-right swapping (LR-swapping): Expands the feature matrix by including all possible left-right swap combinations for the upper and lower limbs ({"no swap", "upper LR-swap", "lower LR-swap", "upper & lower LR-swap"}.

2. Upper-lower limb pairing (UL-pairing): Reduces feature dimensionality by pairing one upper-limb wearable with one lower-limb wearable, generating combinations like "left-arm & left-leg", "left-arm & right-leg", etc..

Both augmentation strategies were also applied to the validation and test set, and final predictions were aggregated across combinations, effectively functioning as a mini-ensemble strategy.

Our final pipeline utilized UL-pairing, and achieved a macro F1 score of 91.5%. Postprocessing further improved this score to 97%. Notably, postprocessed errors only occurred between activities and the transient phases (and thus never between distinct activities). This suggests that the remaining prediction errors were primarily due to annotation inconsistencies in activity start and end times. This classical ML pipeline significantly outperformed SotA DL approaches, such as (shallow) DeepConvLSTM [3] or Attend-and-Discriminate [4] models, which achieved a macro F1 score of 75.8% and 80.6%, respectively.

In summary, our results confirm the success and competitiveness of traditional machine-learning approaches, thereby fulfilling **RG2-B**. Across both competitions, our classical machine learning pipelines substantially outperformed deep learning approaches while requiring substantially less training time and offering greater interpretability, making them more elegant solutions.

RG3: Advancing the analysis of real-world wearable data

Chapter 6 provided actionable guidelines for improving wearable and EMA data quality in real-world monitoring studies, relying strongly on visualization-based methodologies. These methods aid in tracking participant compliance and identifying biases in wearable usage patterns. Additionally, the chapter introduced an improved non-wear detection algorithm and a methodology to assess metric variability in the presence of missing data, collectively realizing **RG3-A**. Several of these recommendations were applied and proven relevant in Chapter 7, and 8.

In Chapter 7, we investigated movement behavior during the ictal phase of cluster headache attacks. While the initial hypothesis suggested increased movement during attacks, our analysis revealed the opposite—a reduction in movement. This finding challenges clinical expectations and suggests that real-world movement patterns during

attacks may be influenced by external factors, such as the use of preventive or acute medications to manage symptoms.

In Chapter 8, we extended this analysis to the daytime movement of migraine patients, employing a dual analytical approach where both AEE and HAR machine learning (ML) predictions were utilized as input signals for our analysis. Our results indicated a reduction in daytime movement during the ictal phase, aligning with clinical insights on the mechanosensitivity of migraine attacks [5]. Additionally, we observed a decrease in evening movement on headache days, which may be attributed to increased fatigue or greater behavioral flexibility in the evening, as most daily obligations occur earlier in the day.

Together, these chapters underscore the potential of wearable devices for real-world monitoring of behavioral patterns during headache events, contributing to **RG3-B**. However, further research with larger cohorts is needed to fully understand the complex factors influencing movement behavior in headache disorders.

9.2 Impact and Use Cases

With my background as an industrial engineer, I prioritize practical impact and effectiveness just as much as—if not more than—academic contributions. Below, I outline the tangible impact of our open-source toolkits, datasets, and experiments, along with key insights we have gained.

- Chapter 2 resulted in an open-source repository that contains the codebase of Plotly-Resampler, licensed under MIT, making the code easily accessible to the community. By distributing the package through PyPI, users can easily install it. Furthermore, integration efforts with other frameworks, such as `Nixtla/statsforecast` and `pycaret`, have driven wider adoption, as evidenced by having 1000+ GitHub stars and 10M+ installs, leading to more mature code.

  Initially, Plotly-Resampler was a monolithic package that included implementations for all data-aggregation algorithms. However, through the analysis of data-aggregation methods in Chapter 3 and the development of algorithmic improvements in Appendix A, we restructured its architecture. Concretely, we introduced a high-level interface in Plotly-Resampler, while migrating the core data-aggregation codebase into a new package: `tsdownsample` [6]. This package provides highly CPU-optimized implementations of downsampling algorithms and extends the usage of data-aggregation methods to other visualization toolkits beyond `plotly.js`. Notably, Anaconda has adopted tsdownsample for use in Holoviews [1].

  Based on GitHub issues and online meetings, we know that Plotly-Resampler is

[1] https://github.com/holoviz/holoviews/pull/6059

actively used worldwide by oceanographers, battery-testing companies, electrical aviation startups (BETA technologies), epilepsy researchers (Byteflies), car manufacturers (Mercedes), and stock traders (Blackrock).

- The work presented in Chapters 6, 7 and 8 was carried out in the scope of the imec.AAA context-aware health monitoring project [2]. This multidisciplinary collaboration with the neurology department of Ghent University hospital contributed towards obtaining real-world clinical insights into cluster headache and migraine.
  To promote reproducibility and practical application, a portion of the *mBrain* dataset was made openly accessible, and the code related to the data quality improvements was open-sourced. This facilitates the adoption of our proposed countermeasures by other researchers and practitioners.
  Additionally, the EMA-registration *mBrain* application, further developed by colleagues, was made available on the Google Play Store [3]. This digital application serves as an accessible headache diary for a broad audience, enabling retrospective analyses to investigate covariates such as the similarity of headache characteristics, for which I conducted a first analysis within the *mBrain* study [7].
  Beyond my own work, other researchers have also benefited from the collective efforts of the *mBrain* study and dataset. For instance, in De Brouwer et al. [8], the whole headache application and study setup are described in detail, facilitating the reproducibility of the paradigm. In Stojchevska et al. [9], contextual data streams of the smartphone were utilized to analyze smartphone behavior changes in the proximity of headaches.
  Last, the outcomes and progress of the *mBrain* study have also been disseminated broadly. Findings were shared with both the scientific community and the general public through an article in the EOS magazine, presentations at national and international conferences, and public outreach events such as *"Wetenschapscafé"* and *"Kraks @ Krook"*.

- Chapter 4 and 5 leverage Plotly-Resampler to visualize the data, as well as the open-source MIT-licensed `tsflex` toolkit, which is outlined in Appendix B. `tsflex` has demonstrated its value across various research projects, from near-real-time ML HAR models in the *mBrain* study to applications beyond healthcare, such as predictive maintenance. Additionally, `tsflex` has played a pivotal role in a high-impact publication in which we underscored the relevance of traditional ML [10]. With these contributions, I hope to influence the scientific community to not only focus on DL but also see traditional ML as a worthy competitor.

[2] https://predict.idlab.ugent.be/projects/mbrain/
[3] https://play.google.com/store/apps/details?id=be.ugent.idlab.predict.headacheconnect

9.3 Future Work

This dissertation addressed key challenges in large-scale line chart visualization, real-world wearable monitoring studies, and applied machine learning on time series data. Along the way, new questions and opportunities for further exploration emerged. This section highlights potential directions for future research and development.

9.3.1 Scalable Time Series Visualization

Dynamic aggregation of other data types

While line charts are well-suited for time series analytics, the most effective visualization depends on the specific task and characteristics of the data. In some cases, alternative visual formats may be more appropriate. For example, time-frequency plots can highlight spectral dynamics, event timelines can emphasize discrete occurrences, and weekly bar charts may better reveal periodic trends [11]. Identifying when to use these alternatives remains an open question. Future research should aim to establish practical guidelines for selecting appropriate aggregation strategies and visualization types based on the use case.

Moreover, for financial time series data, such as stock prices, candlestick charts can be more informative [12]. These charts aggregate data into "candles," each representing four key values: open, high, low, and close. Since these four values align with the M4 aggregation method [13], a proof-of-concept for dynamic candlestick aggregation was implemented in Plotly-Resampler[4]. This is also demonstrated in Figure 9.1 (left top). However, this preliminary implementation has limitations. Currently, the bucket size is determined by the ratio between the front-end view and the number of displayed candlesticks, sometimes resulting in non-standard intervals (e.g., 5.12 hours). Such irregular bucket sizes may hinder interpretability. Future work should address this by ensuring bucket sizes remain standardized and meaningful.

Similarly, uniform shape-preserving subsampling algorithms typically use fixed-width buckets, which align well with linear x-axes. However, certain scenarios—such as visualizing the power spectral density (PSD) of a signal—benefit from logarithmic axis scaling, requiring buckets that span equal ranges in log space rather than in linear space. To address this, I developed a preliminary implementation of the `logLTTB` algorithm, which uses *log*-scaled equidistant buckets for data sampling. However, this approach remains unoptimized as the corresponding bucket sizes in real space vary significantly, posing additional computational challenges. As shown in the right column of Figure 9.1, the `logLTTB` algorithm prioritizes low-frequency ranges when sampling PSD data, contrasting with standard LTTB sampling. Future optimizations should refine this approach to improve consistency and efficiency.

[4] https://github.com/predict-idlab/plotly-resampler/pull/289

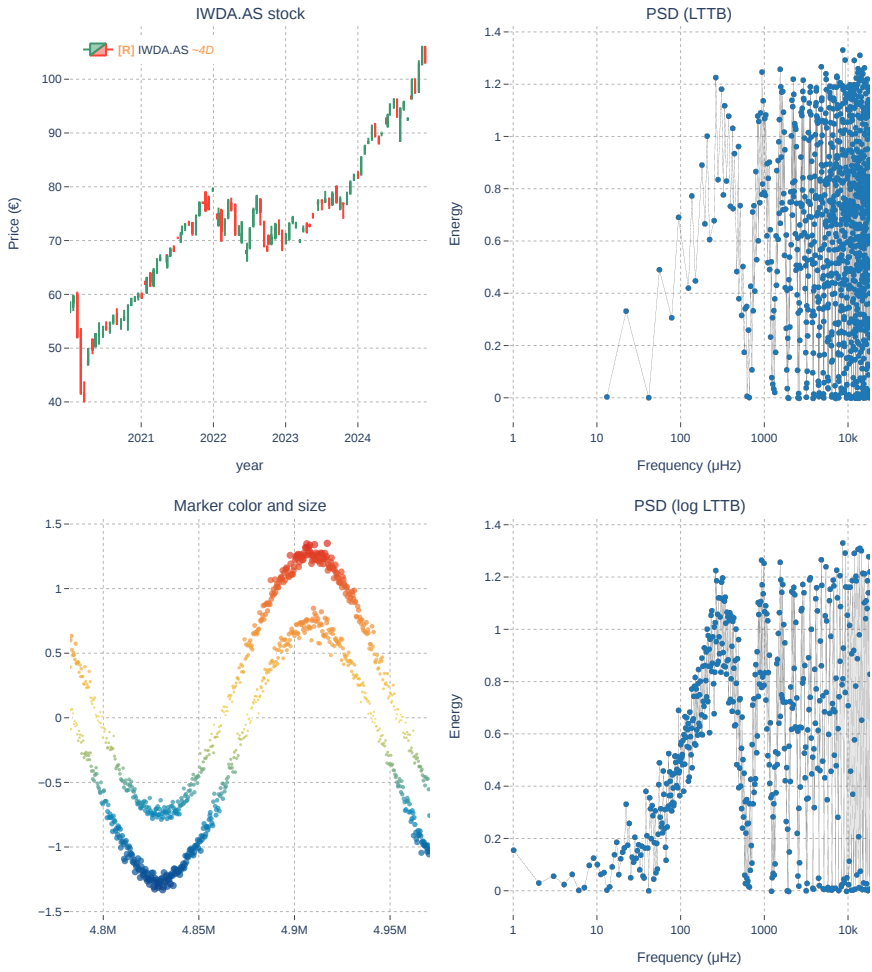


Figure 9.1: Illustration of several specialized time series visualization types in Plotly-Resampler. The upper-left subplot presents the IWDA.AS ticker, also depicted in Figure 1.2, using a candlestick plot. The lower-left subplot shows a scatter plot of a noisy sine signal, with marker size and color encoding the amplitude. The subplots in the second column depict a power spectral density (PSD) transformation of a signal, plotted on a logarithmic x-axis. Note how the `logLTTB` method in the lower right subplot more effectively captures low-frequency variations, as the utilized log-buckets align with the visualization's logarithmic x-axis.

Beyond aggregation methods, other line-chart properties, such as marker color and size, may enhance visualization effectiveness by emphasizing specific data aspects or encoding additional variables, such as a sensor's health index [14]. Although prior studies have examined the perceptual impact of these properties individually, their combined effect with data aggregation algorithms remains unexplored. Future research could

investigate how integrating subsampling techniques with marker attribute adjustments (e.g., color and size) affects visualization utility and interpretability.

Finally, certain time series signal transformations cannot be represented using line charts. For example, the short-time Fourier transform (STFT) is best visualized as a 2D-heatmap. Similarly to line charts, scalable 2D-heatmap visualizations can be achieved through data aggregation techniques—specifically, rasterization-based methods, such as mean-bucket aggregation. In preliminary work, a proof-of-concept implementation was created to perform 2D-heatmap aggregation in Plotly-Resampler[5]. However, achieving an efficient implementation requires further exploration of optimal strategies, including trade-offs between heatmap representations, static images, data decimation, and 2D aggregation algorithms.

Generalize MinMax-preselection

In Appendix A, we introduced how MinMax-preselection enables computationally efficient bucket search-space reduction for subselecting data points with LTTB. Building on the observation in Chapter 3 that vertical extrema are crucial, we hypothesize that MinMax-preselection benefits not only LTTB, but also other shape-preserving algorithms. Future research could explore how MinMax-preselection can be applied to other existing algorithms, such as Ramer–Douglas–Peucker (RDP) or Visvalingam–Whyatt (VW).

Additionally, the relevance of MinMax-preselection even applies to newly proposed shape-preserving subsampling algorithms. In September 2024, Li et al. introduced feature-preserving compensated sampling (FPCS) [15], a novel subsampling method. Upon reviewing and implementing their approach, we observed that the retained datapoints tend to be a subset of MinMax-preselected datapoints. Since FPCS is not parallelizable, incorporating MinMax-preselection could improve its efficiency. Figure 9.2 compares the visual representativity of the (MinMax)FPCS algorithm with other algorithms. These first analysis results show that (MinMax)FPCS appears to be more data-efficient than other algorithms, achieving higher visual representativeness at lower n_{out}. These findings highlight the potential of MinMax-preselection as a general strategy to improve both performance and efficiency in shape-preserving subsampling methods.

[5] https://github.com/predict-idlab/plotly-resampler/issues/169

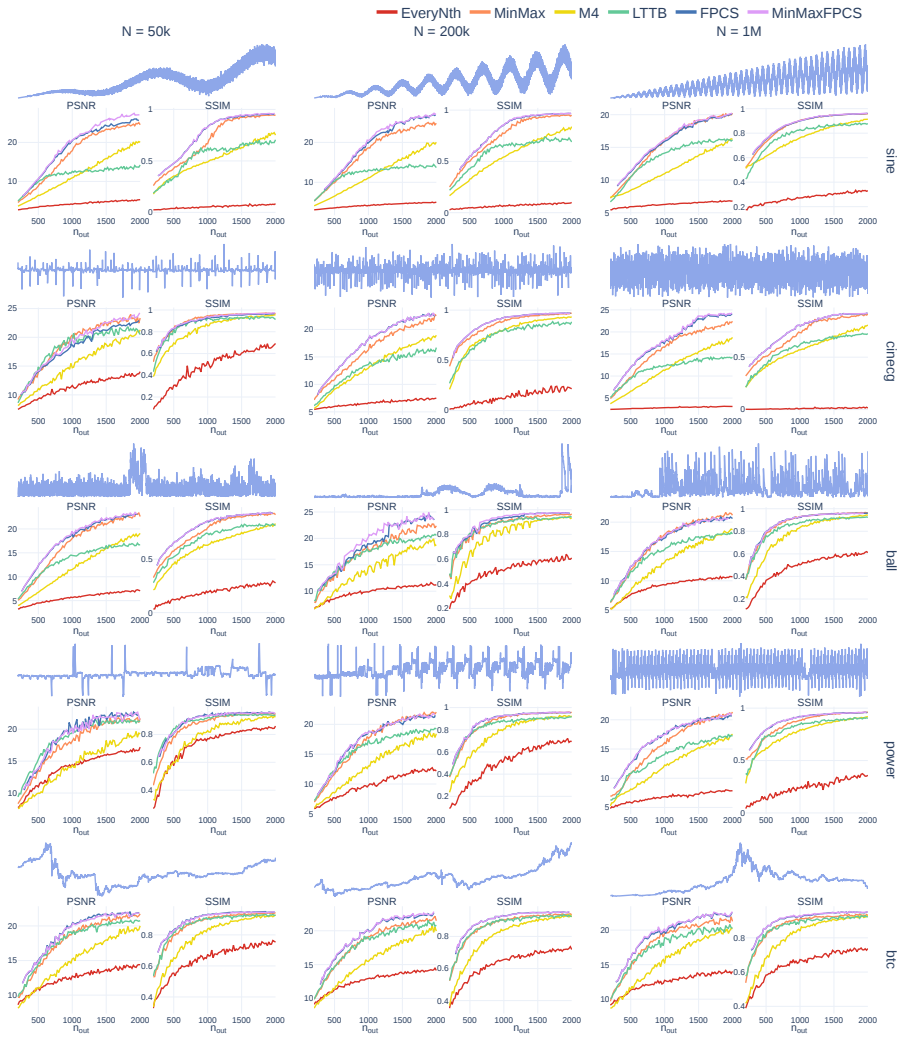


Figure 9.2: Assessing visual representativeness of (`MinMax`)FPCS for various time series templates. Each row displays a distinct time series dataset, with columns indicating the template size. All image templates, including the blue reference templates which are depicted above the metric subplots, were generated using Plotly's default settings (linear interpolation, line-width of 2 pixels). The metric subplots reveal trends in aggregation algorithm performance as n_{out} (x-axis) increases (range [200, 2000]). More information about these trends, metrics, conv-mask scaling, and templates can be found in Chapter 3. SSIM denotes conv-mask scaled structural similarity, while PSNR represents conv-mask scaled peak signal to noise ratio.

9.3.2 Wearable Data Analytics

Traditional machine learning for HAR

In Chapters 4 and 5, the first place was secured in two machine learning competitions, both focused on wearable-based human activity recognition (HAR). This success led me to hypothesize that traditional machine learning methods—though already validated in our prior work within the sleep domain [10]—have been largely overlooked in favor of more complex deep learning approaches in broader HAR applications.

To explore this hypothesis, a preliminary benchmark was conducted, comparing our multi-resolution feature approach to the work of Bock et al., who demonstrated that a Shallow DeepConvLSTM network achieves (near) state-of-the-art (SoTA) performance across five different datasets [21]. Specifically, I computed multi-resolution features using window sizes $w \in 1s,\ 2s,\ 4s,\ 8s,\ 16s$ with a stride of 0.5s. This evaluation was performed on both the processed data provided by Bock et al. [3] and the raw data, following the same validation procedure as in their study.

Table 9.1 presents our preliminary results. On raw data, our approach significantly outperforms the Shallow DeepConvLSTM baseline. On processed data, our method also performs better in most cases, except for the Opportunity dataset, where it performs slightly worse. These findings suggest that traditional machine learning remains highly competitive for human activity recognition. However, given the exploratory nature of these experiments, definitive conclusions cannot be drawn yet. Future work should further investigate the efficacy of multi-resolution features for HAR to establish their broader applicability and robustness.

Investigating rotation-invariant aggregations

In both ML challenges, we observed inconsistencies in device orientation, highlighting the need for rotation-invariant processing within the ML pipeline. As an initial approach, we applied the signal magnitude vector (SMV) transformation to remove rotation information. However, in both ML competitions, this method underperformed compared to rotation-aware feature engineering.

Despite its frequent use in the literature, SMV proves to be a suboptimal approach,

Table 9.1: Comparison of multi-resolution features (and feature shifting) against the best results from Bock et al. [3].

	(Best F1 score) Bock et al. [3]			**Mine (using processed data from [3])**			**Mine (using raw data)**		
Dataset	**Pr.**	**Rc.**	**F1**	**Pr.**	**Rc.**	**F1**	**Pr.**	**Rc.**	**F1**
Opportunity [16]	66.6%	47.2%	51.0%	52.0%	53.0%	49.0%	68.9%	53.9%	55.7%
HHAR [17]	50.7%	50.8%	46.7%	70.0%	67.2%	67.5%	69.8%	67.3%	67.5%
Wetlab [18]	40.0%	45.4%	39.2%	55.0%	51.3%	52.2%	51.0%	48.9%	49.3%
RWHAR [19]	77.6%	76.3%	74.4%	84.9%	84.4%	84.6%	90.0%	89.3%	89.6%
SBHAR* [20]	72.6%	76.2%	71.6%	88.4%	92.2%	90.2%	89.1%	92.9%	90.9%

often overlooking alternative transformations that could more effectively achieve rotation invariance. We hypothesize that integrating more effective rotation-invariant aggregation methods—either at the signal level (as demonstrated in Chapter 5) or at the feature level (as in Chapter 4)—could improve performance in HAR tasks. Future work should investigate these methods to enhance the robustness of wearable-based ML models.

Physiological data analysis

My research on wearable devices has primarily focused on accelerometer data due to its objectivity and resilience to noise. However, as discussed in Section 1.3, wearable devices often capture multiple physiological signals, such as heart rate, skin temperature, and skin conductance. Analyzing these multimodal signals introduces additional complexity, particularly due to inter-individual variability.

To address this, intra-subject comparisons and the use of linear mixed (effect) models (LMMs) offer promising strategies for reducing variability at the group level. Nonetheless, analyzing within-subject changes may still require accounting for potential non-linear relationships in physiological measurements—such as the non-linear nature of heart rate [22]. A complementary, data-driven strategy involves incorporating contextual information such as an individual's fitness level or resting heart rate, or adopting personalized modeling approaches to enhance interpretability and accuracy. Several of these countermeasures are currently being further explored by colleagues.

Effective visualization remains crucial to ensure that physiological analysis yields reliable and meaningful insights. We also hypothesize that aggregating physiological signals—such as deriving resting heart rate from raw heart rate data—is more informative for assessing long-term health or overall physiological burden than for evaluating short-term or isolated events. As a direction for future research, it would be valuable to explore how aggregated measures like resting heart rate and heart rate variability relate to headache-related covariates within the *mBrain* dataset.

Speech-based Ambulatory Monitoring

Wearable technologies offer promising opportunities for enhancing the real-world monitoring of chronic disorders. In this dissertation, I demonstrated how wearable movement monitoring can reveal behavioral changes associated with headache events, even though interpreting these patterns remains complex due to individual variability and the influence of treatments.

Expanding beyond movement and physiological sensing, incorporating additional modalities can further enrich the contextual understanding of disorders. In particular, audio (speech) signals represent a valuable yet underexplored data source [23]. Speech can provide insights into both environmental context (e.g., social settings, ambient noise) and the user's emotional or physical state. However, as extensively demonstrated

in this dissertation, translating these lab findings to real-world settings is non-trivial. As a first step, I designed a novel speech paradigm aimed at minimizing cognitive wandering, making it suitable for both in-lab and real-world deployment [24]. Ongoing work by colleagues focuses on integrating speech-based acquisition and processing into monitoring platforms while carefully addressing privacy considerations.

Together, these insights open exciting avenues for multimodal monitoring strategies that combine movement, physiological, and acoustic signals, enabling a more comprehensive and personalized understanding of patients' daily experiences.

Generalization of non-wear detection

In Section 6.5.1 of Chapter 6, an improved non-wear detection pipeline was designed for the Empatica E4 wearable, demonstrating greater efficiency and a higher detection rate than the off-wrist detection pipeline proposed by Böttcher et al. [25]. However, both our and Böttcher's pipelines were designed and validated within the same geographical and demographic user group, limiting their generalizability.

Since temperature is a key modality for fine-grained non-wear detection, validating these algorithms across diverse geographical regions and climates is essential. I hypothesize that under varying environmental conditions, a more sophisticated temperature-specific signal quality index (SQI) may be required instead of simple thresholding. For example, incorporating a rate-of-change metric, as used in other studies such as Pagnamenta et al. [26], could improve generalizability. Furthermore, our current analysis was constrained to a single device, further restricting generalizability. Future research should address these limitations by validating across multiple wearable devices and diverse environmental conditions.

9.4 Closing Remark

Wearable technologies are no longer confined to laboratories or short-term studies—they are becoming an integral part of daily life, continuously capturing streams of physiological and behavioral data in real-world contexts. This dissertation has examined the technical, methodological, and practical challenges of analyzing the large-scale time series data generated by these devices. From improving the scalability of visualization tools to re-assessing the value of classical machine learning approaches, the contributions presented here address core obstacles to the broader adoption and understanding of wearable data analytics. While these advances mark meaningful progress, they also underscore the complexity and breadth of this evolving field.

The integration of wearable technologies into everyday environments presents significant opportunities—not only in healthcare, but also in preventive medicine, mental health, human performance, sustainability, and urban planning. In these settings, continuous and context-aware monitoring can uncover insights that were previously

inaccessible. Realizing this potential, however, introduces new challenges: building robust and scalable pipelines for multimodal data under unpredictable conditions; ensuring interoperability across heterogeneous devices and platforms; protecting user privacy through responsible data practices; and designing intuitive, scalable interfaces for a wide range of stakeholders.

This dissertation has directly addressed several of these issues, but many open questions remain. Developing systems that are both reliable and interpretable in the face of noisy, real-world data is a complex undertaking—but it is precisely this complexity that makes the work both important and rewarding. As wearable sensors become more pervasive and data streams increasingly rich, our responsibility extends beyond advancing algorithms. It lies in designing systems that are transparent, resilient, and ultimately worthy of users' trust.

References

[1] Lin Wang, Mathias Ciliberto, Hristijan Gjoreski, Paula Lago, Kazuya Murao, Tsuyoshi Okita, and Daniel Roggen. "Summary of SHL Challenge 2024: Motion Sensor-based Locomotion and Transportation Mode Recognition in Missing Data Scenarios". In: *Companion of the 2024 on ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 2024, pp. 555–562.

[2] Marius Bock, Hilde Kuehne, Kristof Van Laerhoven, and Michael Moeller. "Wear: An outdoor sports dataset for wearable and egocentric activity recognition". In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8.4 (2024), pp. 1–21.

[3] Marius Bock, Alexander Hölzemann, Michael Moeller, and Kristof Van Laerhoven. "Improving deep learning for HAR with shallow LSTMs". In: *Proceedings of the 2021 ACM International Symposium on Wearable Computers*. 2021, pp. 7–12.

[4] Alireza Abedin, Mahsa Ehsanpour, Qinfeng Shi, Hamid Rezatofighi, and Damith C Ranasinghe. "Attend and discriminate: Beyond the state-of-the-art for human activity recognition using wearable sensors". In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5.1 (2021), pp. 1–22.

[5] AM Strassman, SA Raymond, and R Burstein. "Sensitization of meningeal sensory neurons and the origin of headaches". In: *Nature* 384.6609 (1996), pp. 560–564.

[6] Jeroen Van Der Donckt, Jonas Van Der Donckt, and Sofie Van Hoecke. "tsdownsample: high-performance time series downsampling for scalable visualization". In: *SoftwareX* 29 (2025), p. 102045.

[7] Nicolas Vandenbussche, Jonas Van Der Donckt, Mathias De Brouwer, Bram Steenwinckel, Marija Stojchevska, Femke Ongenae, Sofie Van Hoecke, and Koen Paemeleire. "A longitudinal comparative study of headache event symptomatology in patients with migraine between smartphone-based headache diary entries and clinician-led intake interviews". In: *Neurology International* (2025).

[8] Mathias De Brouwer, Nicolas Vandenbussche, Bram Steenwinckel, Marija Stojchevska, Jonas Van Der Donckt, Vic Degraeve, Jasper Vaneessen, Filip De Turck, Bruno Volckaert, Paul Boon, et al. "mBrain: towards the continuous follow-up and headache classification of primary headache disorder patients". In: *BMC medical informatics and decision making* 22.1 (2022), p. 87.

[9] Marija Stojchevska, Jonas Van Der Donckt, Nicolas Vandenbussche, Mathias De Brouwer, Koen Paemeleire, Femke Ongenae, and Sofie Van Hoecke. "Uncovering the potential of smartphones for behavior monitoring during migraine follow-up". In: *BMC Medical Informatics and Decision Making* 25.1 (2025), p. 88.

[10] Jeroen Van Der Donckt, Jonas Van Der Donckt, Emiel Deprost, Nicolas Vandenbussche, Michael Rademaker, Gilles Vandewiele, and Sofie Van Hoecke. "Do not sleep on traditional machine learning: Simple and interpretable techniques are competitive to deep learning for sleep scoring". In: *Biomedical Signal Processing and Control* 81 (2023), p. 104429.

[11] Ben Shneiderman. "The eyes have it: A task by data type taxonomy for information visualizations". In: *The craft of information visualization*. Elsevier, 2003, pp. 364–371.

[12] Sungahn Ko, Isaac Cho, Shehzad Afzal, Calvin Yau, Junghoon Chae, Abish Malik, Kaethe Beck, Yun Jang, William Ribarsky, and David S Ebert. "A survey on visual analysis approaches for financial data". In: *Computer Graphics Forum*. Vol. 35. 3. Wiley Online Library. 2016, pp. 599–617.

[13] Uwe Jugel, Zbigniew Jerzak, Gregor Hackenbroich, and Volker Markl. "M4: a visualization-oriented time series data aggregation". In: *Proceedings of the VLDB Endowment* 7.10 (2014), pp. 797–808.

[14] Ghulam Jilani Quadri and Paul Rosen. "A Survey of Perception-Based Visualization Studies by Task". In: *IEEE Transactions on Visualization and Computer Graphics* 28.12 (Dec. 2022), pp. 5026–5048. ISSN: 1077-2626, 1941-0506, 2160-9306. DOI: 10.1109/TVCG.2021.3098240. URL: https://ieeexplore.ieee.org/document/9492011/ (visited on 02/03/2025).

[15] Hongyan Li, Bo Yang, and Yansong Chua. "FPCS: Feature Preserving Compensated Sampling of Streaming Time Series Data". In: *IEEE Transactions on Visualization and Computer Graphics* (2024).

[16] Daniel Roggen, Alberto Calatroni, Mirco Rossi, Thomas Holleczek, Kilian Förster, Gerhard Tröster, Paul Lukowicz, David Bannach, Gerald Pirkl, Alois Ferscha, et al. "Collecting complex activity datasets in highly rich networked sensor environments". In: *2010 Seventh international conference on networked sensing systems (INSS)*. IEEE. 2010, pp. 233–240.

[17] Allan Stisen, Henrik Blunck, Sourav Bhattacharya, Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind Dey, Tobias Sonne, and Mads Møller Jensen. "Smart devices are different: Assessing and mitigatingmobile sensing heterogeneities for activity recognition". In: *Proceedings of the 13th ACM conference on embedded networked sensor systems*. 2015, pp. 127–140.

[18] Philipp M Scholl, Matthias Wille, and Kristof Van Laerhoven. "Wearables in the wet lab: a laboratory system for capturing and guiding experiments". In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 2015, pp. 589–599.

[19] Timo Sztyler and Heiner Stuckenschmidt. "On-body localization of wearable devices: An investigation of position-aware activity recognition". In: *2016 IEEE international conference on pervasive computing and communications (PerCom)*. IEEE. 2016, pp. 1–9.

[20] Jorge-L Reyes-Ortiz, Luca Oneto, Albert Samà, Xavier Parra, and Davide Anguita. "Transition-aware human activity recognition using smartphones". In: *Neurocomputing* 171 (2016), pp. 754–767.

[21] Marius Bock, Michael Moeller, and Kristof Van Laerhoven. "Temporal Action Localization for Inertial-based Human Activity Recognition". In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8.4 (2024), pp. 1–19.

[22] Buccelletti Francesco, Bocci Maria Grazia, Gilardi Emanuele, Fiore Valentina, Calcinaro Sara, Fragnoli Chiara, Maviglia Riccardo, and Franceschi Francesco. "Linear and nonlinear heart rate variability indexes in clinical practice". In: *Computational and mathematical methods in medicine* 2012.1 (2012), p. 219080.

[23] Taiba Majid Wani, Teddy Surya Gunawan, Syed Asif Ahmad Qadri, Mira Kartiwi, and Eliathamby Ambikairajah. "A comprehensive review of speech emotion recognition systems". In: *IEEE access* 9 (2021), pp. 47795–47814.

[24] Jonas Van Der Donckt, Mitchel Kappen, Vic Degraeve, Kris Demuynck, Marie-Anne Vanderhasselt, and Sofie Van Hoecke. "Ecologically valid speech collection in behavioral research: The Ghent Semi-spontaneous Speech Paradigm (GSSP)". In: *Behavior Research Methods* 56.6 (2024), pp. 5693–5708.

[25] Sebastian Böttcher, Solveig Vieluf, Elisa Bruno, Boney Joseph, Nino Epitashvili, Andrea Biondi, Nicolas Zabler, Martin Glasstetter, Matthias Dümpelmann, Kristof Van Laerhoven, et al. "Data quality evaluation in wearable monitoring". In: *Scientific reports* 12.1 (2022), p. 21412.

[26] Sara Pagnamenta, Karoline Blix Grønvik, Kamiar Aminian, Beatrix Vereijken, and Anisoara Paraschiv-Ionescu. "Putting temperature into the equation: development and validation of algorithms to distinguish non-wearing from inactivity and sleep in wearable sensors". In: *Sensors* 22.3 (2022), p. 1117.

A

MinMaxLTTB: Leveraging MinMax-Preselection to Scale LTTB

In Chapter 3, we observed that selecting vertical extrema plays a crucial role in most shape-preserving subsampling algorithms. Building on this insight, this Appendix introduces the MinMaxLTTB algorithm, a novel subsampling algorithm that first utilizes the efficient MinMax algorithm for preselecting vertical extrema, after which the computationally intensive LTTB algorithm is applied. This computational improvement over LTTB is a significant enhancement in the shape-preserving subsampling domain, especially due to its ease of integration into query engines, thereby realizing **RG1-C**. Given both the high visual representativeness along with its computational efficiency, MinMaxLTTB is currently the default downsampling technique in Plotly-Resampler (see Chapter 2).

While MinMaxLTTB was co-designed with Jeroen Van Der Donckt, my (joint) contributions include:

- Discovering the importance of vertical extrema.
- Implementing MinMaxLTTB and LTTB in Python and C.
- Evaluating MinMaxLTTB's visual representativeness across various preselection ratios using the methodology outlined in Chapter 3.
- Benchmarking the performance of LTTB and MinMaxLTTB.

- Hypothesizing broader applications of MinMax-preselection to other computationally intensive data point selection algorithms.
- Open source our fully reproducible visual representativeness analysis and the benchmarking.

MinMaxLTTB: Leveraging MinMax-Preselection to Scale LTTB

Jeroen Van Der Donckt [1], Jonas Van Der Donckt [1], and Sofie Van Hoecke

Published in proceedings of the "2023 IEEE Conference of Visualization and Visual Analytics. Melbourne (Australia), 2023"

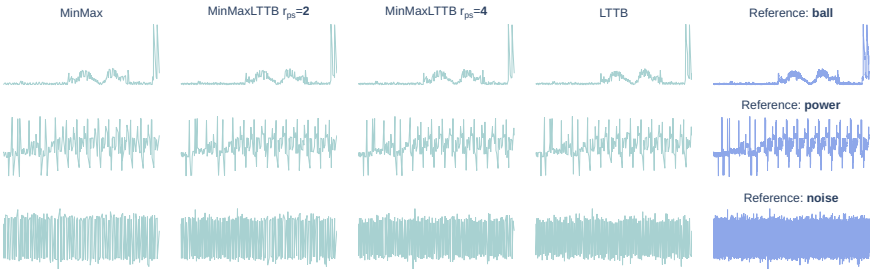


Figure A.1: Visual analysis of MinMax (column 1), `MinMaxLTTB` with preselection ratio $r_{ps} \in \{2, 4\}$ (column 2-3), and LTTB (column 4). Each row corresponds to a unique template (selected from Chapter 3), with the rightmost column presenting the reference images, constructed from 200k data points. The four adjacent aggregations utilize an equal number of output samples ($n_{out} = 200$). Columns 2 and 3 illustrate the effect of the preselection ratio r_{ps} for MinMaxLTTB. Visualization properties used include: Plotly as toolkit, line-width of 2 pixels, and anti-aliasing enabled. A GIF further demonstrates the above visualization for varying values for n_{out}.

Abstract Visualization plays an important role in the analysis and exploration of time series data. To facilitate efficient visualization of large datasets, downsampling has emerged as a well-established approach. This work concentrates on largest triangle three buckets (LTTB), a widely adopted downsampling algorithm for time series data point selection. Specifically, we introduce MinMaxLTTB, a two-step algorithm that significantly improves the scalability of LTTB. MinMaxLTTB consists of the following two steps: (i) the MinMax algorithm preselects a certain ratio of minimum and maximum data points, followed by (ii) applying the LTTB algorithm on only these preselected data points, effectively reducing LTTB's time complexity. The MinMax algorithm is computationally efficient and can be parallelized, enabling efficient data point preselection. Additionally, MinMax demonstrates competitive performance in terms of visual representation, making it also an effective data reduction method. Experimental results demonstrate that MinMaxLTTB outperforms LTTB by more than an order of magnitude in terms of computation time. Furthermore, preselecting a small multiple of the desired output size already yields similar visual representativeness compared to LTTB. In summary, MinMaxLTTB leverages the computational efficiency

[1] Contributed equally

of MinMax to scale LTTB, without compromising on LTTB its favorable visualization properties. The code and experiments associated with this chapter can be found at https://github.com/predict-idlab/MinMaxLTTB.

A.1 Introduction

Visualization is widely recognized as a powerful tool for analyzing and exploring time series data, with line charts proving particularly effective for most tasks [1]. As the volume of time series data continues to expand, there is an increasing need for efficient visualization methods capable of handling large datasets [2, 3, 4]. To address this challenge, downsampling has emerged as a well-established technique that involves either aggregating or selecting a representative subset of the time series [5, 1, 6]. By reducing the number of data points while preserving the overall shape of the time series, downsampling minimizes network latency and improves rendering time, making it a vital component in numerous widely adopted time series databases [7, 8].

This work specifically focuses on the largest triangle three buckets (LTTB) algorithm [9], which is a value preserving aggregation method as it downsamples by selecting data points from the original time series [10, 11]. LTTB has gained widespread adoption in industry, with companies like Uber incorporating it as a downsampling function in their M3 metrics platform [7] and TimeScaleDB offering LTTB as a server-side hyperfunction [8].

Despite its broad use, LTTB exhibits certain computational limitations that restrict its applicability to massive datasets containing billions of data points. Specifically, LTTB involves expensive operations to compute triangular surfaces for each data point and requires a sequential pass over the data, making it unsuitable for parallelization. To overcome these computational challenges, we introduce *MinMaxLTTB*, a two-step approach that (i) employs MinMax-preselection as an initial data reduction step, and (ii) applies LTTB on the preselected data points. In particular, this approach leverages the computational efficiency of the MinMax algorithm to make LTTB more scalable.

Utilizing the visual evaluation framework that we proposed in previous work [12], we evaluate the visual representativeness of the proposed MinMaxLTTB algorithm for various MinMax-preselection ratios and compare with LTTB. These findings provide empirical evidence for choosing an appropriate MinMax-preselection ratio. Furthermore, we assess the runtime performance improvement of MinMaxLTTB over LTTB. Notably, the presented technique is the default downsampling approach in our open-source time series visualization tool Plotly-Resampler [13], which has at the time of writing over 3.5 million installations.

A.2 Related Work

This section provides an overview of related work in the field, focusing on the scalability of value preserving data aggregation algorithms for time series line chart visualization.

A.2.1 Time Series Visualization

Visualization is crucial for exploring time series data, with the human eye frequently being advocated as the ultimate data mining instrument [14]. For the majority of time series analysis tasks, simple visualizations, such as line charts, have proven most effective [1]. Interactive visualization techniques, as emphasized by Shneiderman's visual information-seeking mantra [15], are essential for exploring large time series data by providing an overview, enabling zooming and filtering, and allowing access to details-on-demand [16].

Most interactive approaches render visualizations client-side, often in web-based environments [17, 4]. However, this approach presents two significant challenges when handling large data volumes: (i) considerable network latency due to the transmission of large data volumes, and (ii) poor client-side rendering performance [5]. Both aspects limit responsiveness and interactivity, which, as highlighted above, is crucial for effectively exploring time series data. To overcome these limitations, data aggregation has proven to be an effective approach [18, 2, 6].

A.2.2 Data Aggregation for Time Series Visualization

Data aggregation for time series visualization can be categorized into density-wise data aggregation and downsampling. Density-wise data aggregation employs a shared color-coding to generate an image of the data on the server-side, which is transmitted to and displayed on the client front-end [19]. Downsampling reduces the number of data points that are transmitted to the visualization front-end while aiming to preserve specific characteristics or the overall shape of the data. Consequently, this approach results in reduced latency, enhanced client-side rendering time, and increased responsiveness, as the application is not burdened with rendering all data points [5].

Downsampling can be further differentiated into characteristic and value preserving data aggregation. Characteristic data aggregation aims to emphasize specific properties or trends in the data by employing aggregation operations such as mean, median, or smoothing [20, 21]. Conversely, value preserving data aggregation, also referred to as data point selection, selects data points from the original time series with the objective of preserving its overall shape.

A.2.2.1 Value Preserving Data Aggregation

Numerous value preserving data aggregation algorithms have been proposed in literature [22, 23]. Among these, EveryNth, MinMax, M4 [10], and LTTB [9] are arguably the most prevalent algorithms [12, 8, 7].

Table A.1 presents an overview of the computational properties of these four algorithms. The EveryNth algorithm selects every n^{th} data point in order to construct an output of n_{out} data points, which is an inexpensive and common query operation [11]. MinMax downsampling involves selecting the vertical extrema (min and max) for each bin (i.e., bucket B_i), using the (arg)min and (arg)max operations, which are also highly optimized queries [24]. M4 can be viewed as a combination of EveryNth and MinMax, essentially selecting the vertical (min and max) and horizontal (first and last) extrema for each bucket [10]. Given the highly optimized server-side query operations that these three algorithms capitalize on and the relatively low cost of the operations involved (selecting or comparing data), improving the computational properties of these three algorithms is practically infeasible. It is important to note that in-memory approaches also benefit from the low computational cost and the parallelizability of these algorithms.

LTTB is based on the concept of effective triangular areas, which is often employed in line simplification algorithms [9]. Specifically, LTTB selects in each bucket B_i the data point that forms the largest triangular surface with the previously selected data point and the next bucket's (B_{i+1}) average value. In contrast to the above three algorithms, LTTB involves more expensive operations, including (i) computing the average for each bin, and (ii) calculating and comparing the triangular surface for each data point within a bin. Furthermore, this algorithm requires a sequential pass over the data, making parallelization impossible. Implementing LTTB in database solutions also requires a user-defined function, benefiting less from server-side query optimizations [25]. As such, LTTB is considerably more expensive in both out-of-core (i.e., query) and in-memory contexts.

Given these observations on LTTB's computational properties, the focus of this work is to improve the scalability of LTTB, striving towards the same computational properties as M4 and MinMax.

A.3 MinMaxLTTB

We propose MinMaxLTTB, an enhancement to LTTB that addresses its unfavorable computational properties. This is achieved by building on insights from prior research, which indicated the importance of selecting (alternating) vertical extrema for representative data aggregation [12, 9, 10]. Algorithm 1 presents the MinMaxLTTB algorithm in pseudocode. We refer the reader to github.com/predict-idlab/tsdownsample [26] for a concrete implementation of this algorithm. MinMaxLTTB is realized in a two-step

Table A.1: Overview of time series data point selection algorithms, where N denotes the time series length, and n_{out} represents the aggregation output size. We included the MinMaxLTTB algorithm for comparison.

	(i) Time	(ii) Memory	(iii) Parallelizable	(iv) Output Memory
EveryNth	$O(n_{out})$	$O(1)$	✓	$O(n_{out})$
MinMax	$O(N)$	$O(1)$	✓	$O(n_{out})$
M4 [10]	$O(N)$	$O(1)$	✓	$O(n_{out})$
LTTB [9]	$O(N)$	$O(1)$	×	$O(n_{out})$
MinMaxLTTB	$O(N)$	$O(n_{out})$	✓	$O(n_{out})$

algorithm 1: MinMaxLTTB

Input: $data$: original time series
n_{out}: desired number of points to select
r_{ps}: preselection ratio
Result: $selected\_idx$: index of the selected data points

```
/* 1. Preselect r_ps · n_out data points                                    */
1 preselected_idx ← MinMax(data[1 : -1], r_ps · n_out)
/* Implementation detail: prepare the preselected_data                      */
2 preselected_id ← preselected_idx + 1          // Increment index values with 1
3 preselected_idx.prepend(0)                   // Add first data point its index
4 preselected_idx.append(data.len())            // Add last data point its index
5 preselected_data ← data[preselected_idx]
/* 2. Apply LTTB on the pre-selected data points                            */
6 selected_idx ← LTTB(preselected_data, n_out) return selected_idx
```

process, where we (i) *preselect* vertical extrema by using MinMax, and then (ii) *apply* LTTB on the preselected data points. In addition to the MinMax-preselection, the first and last data point of the original time series are also passed to the LTTB algorithm.

In the first step (line 1), we preselect $r_{ps} \cdot n_{out}$ data points, where $r_{ps} \geq 2$ denotes the integer preselection ratio and n_{out} the number of output data points. Remark that $r_{ps}/2$ indicates the number of LTTB *sub-buckets* for which vertical extrema (i.e., min and max) will be selected, e.g., an r_{ps} of 8 can be interpreted as dividing each LTTB bucket into 4 sub-buckets [2].

In the second step (line 6), the LTTB algorithm is applied on the $r_{ps} \cdot n_{out}$ data points, allowing LTTB to scale with n_{out} instead of the time series length N. Although the MinMax algorithm scales with N, it is relatively inexpensive and easily parallelizable, facilitating efficient data reduction. Furthermore, since extracting the (arg)min and (arg)max is a common query, this technique can be seamlessly integrated into database solutions where it benefits from server-side query optimizations such as caching [25].

[2]Note that MinMaxLTTB with $r_{ps} = 1$ corresponds to MinMax aggregation.

Note that in Algorithm 1, additional operations are necessary to prepare the preselected data for input to the LTTB algorithm. Although these are implementation details, it is important to be aware of these intricacies. In the first step (line 1), all data points except the first and last are processed by the MinMax algorithm. We exclude these points because LTTB will always include them in the output, whereas MinMax might not necessarily select them. As a direct consequence of excluding these points, we increment all preselected indices by 1 (line 2) to ensure correct indexing on the original time series. After this operation, we add the indices of the first and last data point to the preselected indices (lines 3 and 4 respectively), facilitating the complete selection of the preselected data (line 5).

A.3.1 Visual Representativeness and Preselection Ratio

We analyze the visual representativeness of MinMaxLTTB, shown in Figure A.2, for various preselection ratios $r_{ps} \in \{2, 4, 6\}$ employing the methodology proposed in Chapter 3. We refer the reader to this prior work for details on the visual representativeness metrics and the selected time series templates.

A first key observation is that MinMaxLTTB's performance curves are on par with LTTB for low-roughness series such as the btc templates, power $\leq$200k, cinecg-50k, and ball-200k. In these cases, the MinMax-preselection ratio has little influence, possibly explained by the absence of prominent extrema in the templates. For higher roughness series such as the noise templates and cinecg $\geq$200k, MinMaxLTTB even seems to perform (slightly) better, especially for low $r_{ps} \in \{2, 4\}$. This observation can be attributed to LTTB favoring the selection of data points near the left bin-edge. Figure A.3 intuitively demonstrates that extrema near the left bin-edge allow creating larger triangular surfaces (with the selected extremum from the previous bucket), potentially omitting larger extrema that occur in the center or right part of the bin. Since MinMax-preselection with a low r_{ps} preselects fewer options, while only considering the vertical position, LTTB's tendency to select near left-bin edge extrema values is partly mitigated. For example, in Figure A.3, using $r_{ps} = 2$, points (2) and (3) would get preselected for bucket B_i, resulting in MinMaxLTTB selecting the bin-maximum (3) instead of (1). This results in improved visual representativeness (compared to the reference template) as more prominent data points are retained.

Consequently, increasing the preselection ratio makes the MinMaxLTTB performance curves less noisy and shift more towards the LTTB curves, as MinMaxLTTB will have more near left bin-edge preselected extrema. In summary, MinMaxLTTB does not degrade in visual representativeness with respect to visual LTTB, and a low $r_{ps} \in \{4, 6\}$ already results in high visual similarity to LTTB.

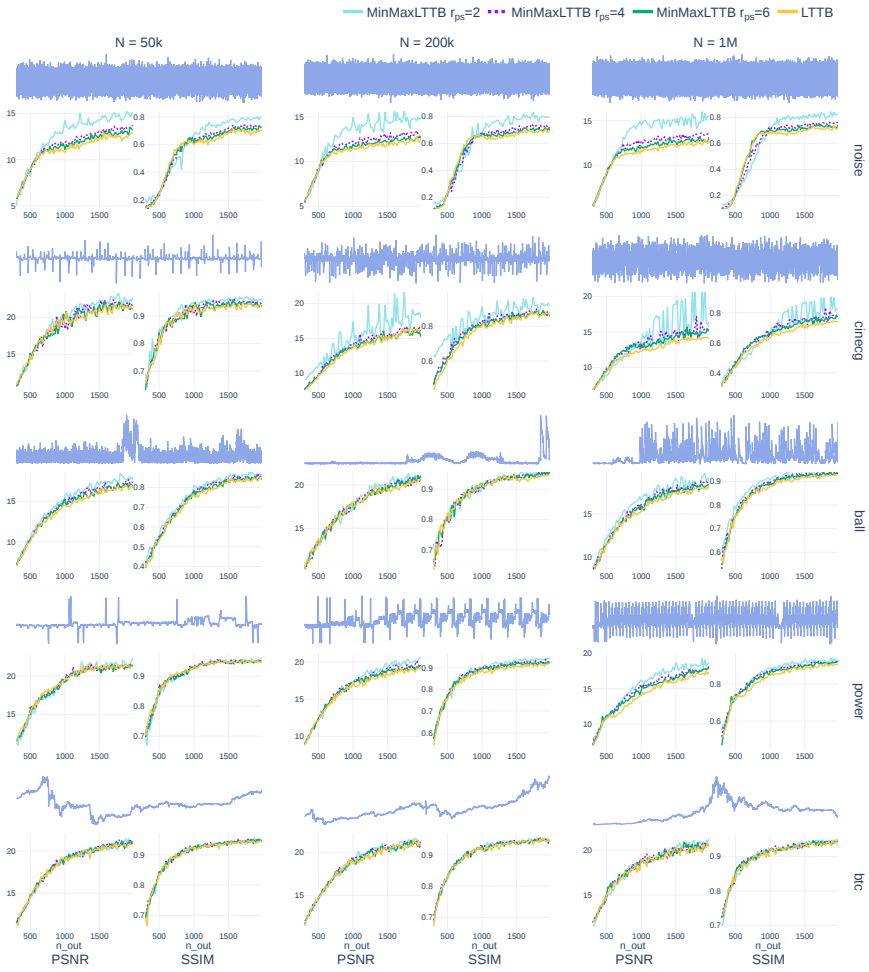


Figure A.2: Assessing visual representativeness of `MinMaxLTTB` with $r_{ps} \in \{2, 4, 6\}$ for various time series templates. Each row displays a distinct time series dataset, with columns indicating the template size. All image templates, including the blue reference templates which are depicted above the metric subplots, were generated using Plotly's default settings (linear interpolation, line-width of 2 pixels). The metric subplots reveal trends in aggregation algorithm performance as n_{out} (x-axis) increases (range [200, 2000]). More information about these trends, metrics, conv-mask scaling, and templates can be found in Chapter 3. SSIM denotes conv-mask scaled structural similarity, while PSNR represents conv-mask scaled Peak Signal to Noise Ratio. A GIF and an HTML animation further demonstrate the visual representativeness of MinMaxLTTB for $r_{ps} \in [1 - 8]$.

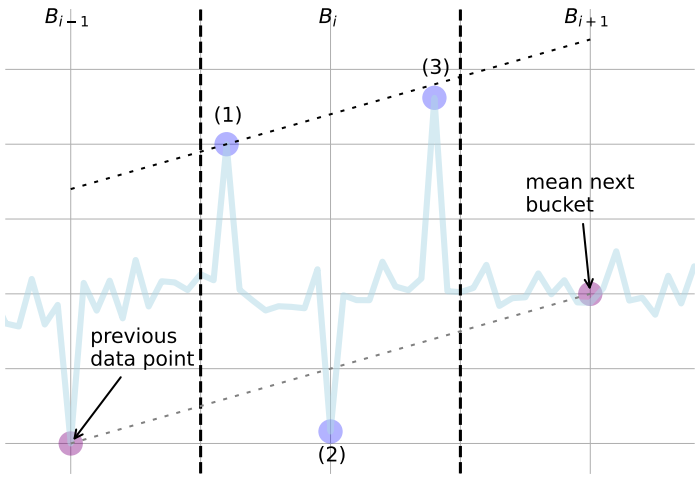


Figure A.3: Illustration of LTTB's tendency to (i) select contrasting extrema values in neighboring buckets and (ii) favor extrema proximity to the left-bin edge. For all points in B_i, triangular areas are calculated using the prior selected point from B_{i-1} and the mean x and y values of B_{i+1}. Points (1) and (3), both opposing the selected extrema of B_{i-1}, compete for the largest triangular surface. Note that extremum (2) would yield a considerably smaller triangle than points (1) and (3). The dashed line passing through point (1) represents the *equisurface* line, with points above it generating larger triangles and those below resulting in smaller ones. This line will always be parallel to the one connecting the previous data point and the mean of the next bucket. Despite being the global extrema of B_i, (3) is not selected as it resides below the equisurface line of (1), illustrating LTTB's tendency to favor data points near the left-bin edge.

A.3.2 Performance

MinMaxLTTB exhibits the same linear time complexity as LTTB (see Table A.1). The memory complexity of MinMaxLTTB is $O(n_{out})$ instead of $O(1)$ because, after the first step, $r_{ps} \cdot n_{out}$ indices are preselected (and thus stored in memory). In the previous section, it was demonstrated that a low r_{ps} is sufficient to achieve comparable visual representativeness. Therefore, the additional memory overhead associated with MinMaxLTTB is almost negligible compared to LTTB, since the memory complexity of constructing the output is $O(n_{out})$ [3].

Although the runtime of both algorithms scales linearly with the time series size (N), the slope of this linear scaling is significantly lower (i.e., better) for MinMaxLTTB. This can intuitively be explained by looking at the operations that are performed (for each data point) by both algorithms. In particular, both algorithms perform comparisons, with LTTB comparing for the largest triangular surface and MinMax comparing for extrema. However, in addition to a comparison, LTTB's operations

[3] Note that this cost is unavoidable for all algorithms, and therefore included in the output memory column of Table A.1.

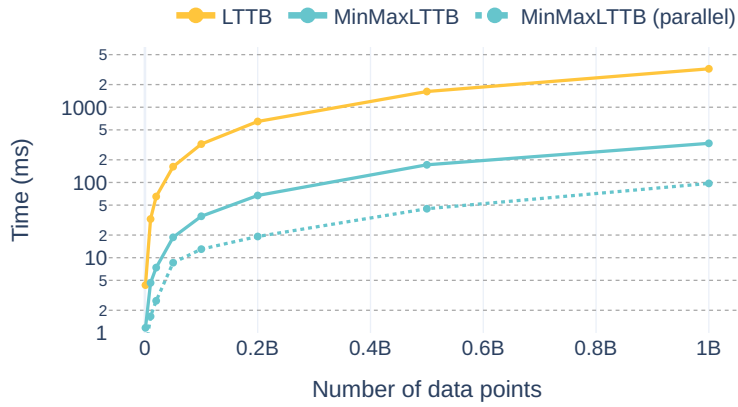


Figure A.4: In-memory performance analysis of LTTB and MinMaxLTTB. For LTTB the C implementation from `plotly-resampler` v0.8.3.2 is used, for MinMaxLTTB the implementation can be found here.

include the calculation of the triangular surfaces (and the bin-wise average), thus requiring more operations than MinMax, resulting in a higher slope for LTTB. In addition to superior linear scaling, MinMaxLTTB also allows for parallelization, which further alleviates the linear scaling to multiple cores.

We evaluate the runtime of MinMaxLTTB and compare with LTTB. To do so, we utilize the in-memory CPU-optimized implementations that are made open source in our tsdownsample package [26]. The benchmarks were executed on a server with an *Intel Xeon E5-2650 v2 (32) @ 3.40GHz* CPU and SAMSUNG M393B1G73QH0-CMA DDR3 1600MT/s RAM, running on the *Ubuntu 18.04.6 LTS x86_64* operating system. Other running processes were limited to a minimum.

Figure A.4 shows the runtime (y-axis) of the LTTB and MinMaxLTTB algorithms for increasing number of data points (x-axis), up to 1 billion data points. We clearly observe the improved scaling of MinMaxLTTB with respect to LTTB. In particular, we notice an order of magnitude (10x) improved runtime for the singe core execution. When applying parallelization on MinMaxLTTB, the performance increases 30x compared to LTTB. This improvement in performance was observed while utilizing 32 threads, which is the number of logical cores on the benchmarking CPU.

A.4 Conclusion

This work proposes MinMaxLTTB, a downsampling technique that mitigates the unfavorable computational properties of LTTB through the use of MinMax-preselection. Our evaluation of MinMaxLTTB's visual representativeness reveals that even a small preselection ratio $r_{ps} \in \{4, 6\}$ yields a high degree of similarity to LTTB. As future work, a case study should further affirm these findings. Performance analysis demon-

strated that MinMaxLTTB's computation time decreases by over an order of magnitude compared to LTTB. This is particularly significant for scalable visualization of big data, as interactive latency greatly influences the rate at which users make observations during exploratory analysis [27]. We further hypothesize that the demonstrated effectiveness of MinMax-preselection could potentially be extended to other computationally demanding data point selection algorithms such as Visvalingam–Whyatt [28], Ramer–Douglas–Peucker [29], and longest line bucket [9]. Finally, the fact that MinMaxLTTB is currently the default aggregation algorithm for already over half a year in a widely utilized visualization tool serves as strong evidence of its efficacy.

References

[1] Wolfgang Aigner, Silvia Miksch, Wolfgang Müller, Heidrun Schumann, and Christian Tominski. "Visualizing time-oriented data—A systematic view". en. In: *Computers & Graphics* 31.3 (June 2007). ZSCC: 0000450, pp. 401–409. ISSN: 00978493. DOI: 10.1016/j.cag.2007.01.030. URL: https://linkinghub.elsevier.com/retrieve/pii/S0097849307000611 (visited on 03/03/2022).

[2] Nikos Bikakis. "Big Data Visualization Tools". en. In: *arXiv:1801.08336 [cs]* (Feb. 2018). ZSCC: 0000075 arXiv: 1801.08336. URL: http://arxiv.org/abs/1801.08336 (visited on 01/26/2022).

[3] Parke Godfrey, Jarek Gryz, Piotr Lasek, and Nasim Razavi. "Interactive visualization of big data". In: *Beyond Databases, Architectures and Structures. Advanced Technologies for Data Mining and Knowledge Discovery: 12th International Conference, BDAS 2016, Ustroń, Poland, May 31-June 3, 2016, Proceedings 11*. Springer. 2016, pp. 3–22.

[4] Rostislav Netek, Jan Brus, and Ondrej Tomecka. "Performance Testing on Marker Clustering and Heatmap Visualization Techniques: A Comparative Study on JavaScript Mapping Libraries". In: *ISPRS International Journal of Geo-Information* 8.8 (2019). ISSN: 2220-9964. DOI: 10.3390/ijgi8080348. URL: https://www.mdpi.com/2220-9964/8/8/348.

[5] Rajeev Agrawal, Anirudh Kadadi, Xiangfeng Dai, and Frederic Andres. "Challenges and opportunities with big data visualization". en. In: *Proceedings of the 7th International Conference on Management of computational and collective intElligence in Digital EcoSystems*. Caraguatatuba Brazil: ACM, Oct. 2015, pp. 169–173. ISBN: 978-1-4503-3480-8. DOI: 10.1145/2857218.2857256. URL: https://dl.acm.org/doi/10.1145/2857218.2857256 (visited on 03/27/2022).

[6] Bum Chul Kwon, Janu Verma, Peter J. Haas, and Cagatay Demiralp. "Sampling for Scalable Visual Analytics". en. In: *IEEE Computer Graphics and Applications* 37.1 (Jan. 2017), pp. 100–108. ISSN: 0272-1716. DOI: 10.1109/MCG.2017.6. URL: http://ieeexplore.ieee.org/document/7819391/ (visited on 03/05/2023).

[7] Benjamin Raskin and Niknunj Aggarwal. *The billion data point challenge: Building a query engine for high cardinality time series data*. uber.com/billion-data-point-challenge. Dec. 2018. URL: https://www.uber.com/en-BE/blog/billion-data-point-challenge/.

[8] Jônatas Davi Paganini. *Downsampling in the database: How data locality can improve data analysis*. www.timescale.com/blog. Feb. 2023. URL: https://www.timescale.com/blog/downsampling-in-the-database-how-data-locality-can-improve-data-analysis/.

[9] Sveinn Steinarsson. "Downsampling Time Series for Visual Representation". en. MA thesis. University of Iceland, 2013. DOI: http://hdl.handle.net/1946/15343. URL: http://hdl.handle.net/1946/15343.

[10] Uwe Jugel, Zbigniew Jerzak, Gregor Hackenbroich, and Volker Markl. "M4: a visualization-oriented time series data aggregation". en. In: *Proceedings of the VLDB Endowment* 7.10 (June 2014), pp. 797–808. ISSN: 2150-8097. DOI: 10.14778/2732951.2732953. URL: https://dl.acm.org/doi/10.14778/2732951.2732953 (visited on 10/14/2022).

[11] Uwe Jugel, Zbigniew Jerzak, Gregor Hackenbroich, and Volker Markl. "VDDA: automatic visualization-driven data aggregation in relational databases". en. In: *The VLDB Journal* 25 (Feb. 2016), pp. 53–77. ISSN: 1066-8888, 0949-877X. DOI: 10.1007/s00778-015-0396-z. URL: http://link.springer.com/10.1007/s00778-015-0396-z (visited on 03/03/2023).

[12] Jonas Van Der Donckt, Jeroen Van Der Donckt, Michael Rademaker, and Sofie Van Hoecke. "Data Point Selection for Line Chart Visualization: Methodological Assessment and Evidence-Based Guidelines". In: *arXiv preprint arXiv:2304.00900* (2023).

[13] Jonas Van Der Donckt, Jeroen Van Der Donckt, Emiel Deprost, and Sofie Van Hoecke. "Plotly-resampler: Effective visual analytics for large time series". In: *2022 IEEE Visualization and Visual Analytics (VIS)*. IEEE. 2022, pp. 21–25. DOI: 10.1109/VIS54862.2022.00013.

[14] Jessica Lin, Eamonn Keogh, and Stefano Lonardi. "Visualizing and Discovering Non-Trivial Patterns in Large Time Series Databases". en. In: *Information Visualization* 4.2 (June 2005). ZSCC: 0000178, pp. 61–82. ISSN: 1473-8716, 1473-8724. DOI: 10.1057/palgrave.ivs.9500089. URL: http://journals.sagepub.com/doi/10.1057/palgrave.ivs.9500089 (visited on 03/11/2022).

[15] Ben Shneiderman. "The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations". en. In: *Proceedings 1996 IEEE symposium on visual languages*. IEEE. 1996, pp. 336–343. DOI: 10.1016/B978-155860915-0/50046-9.

[16] James Walker, Rita Borgo, and Mark W. Jones. "TimeNotes: A Study on Effective Chart Visualization and Interaction Techniques for Time-Series Data". en. In: *IEEE Transactions on Visualization and Computer Graphics* 22.1 (Jan. 2016), pp. 549–558. ISSN: 1077-2626. DOI: 10.1109/TVCG.2015.2467751. URL: http://ieeexplore.ieee.org/document/7192735/ (visited on 03/25/2022).

[17] Enrico G Caldarola and Antonio M Rinaldi. "Big data visualization tools: a survey". In: *Research Gate* (2017).

[18] Sam Kumar, Michael P. Andersen, and David E. Culler. *Mr. Plotter: Unifying Data Reduction Techniques in Storage and Visualization Systems*. en. arXiv:2106.12505 [cs]. June 2021. URL: http://arxiv.org/abs/2106.12505 (visited on 03/05/2023).

[19] Holoviz-community. *Datashader, quickly and accurately render even the largest data*. https://github.com/holoviz/datashader.

[20] Joseph M Hellerstein, Ron Avnur, Andy Chou, Christian Hidber, Chris Olston, Vijayshankar Raman, Tali Roth, and Peter J Haas. "Interactive data analysis: The control project". In: *Computer* 32.8 (1999), pp. 51–59. DOI: 10.1109/2.781635.

[21] Kexin Rong and Peter Bailis. "ASAP: prioritizing attention via time series smoothing". en. In: *Proceedings of the VLDB Endowment* 10.11 (Aug. 2017), pp. 1358–1369. ISSN: 2150-8097. DOI: 10.14778/3137628.3137645. URL: https://dl.acm.org/doi/10.14778/3137628.3137645 (visited on 01/24/2023).

[22] Puleum Bae, Keun-Woo Lim, Woo-Sung Jung, and Young-Bae Ko. "Practical implementation of M4 for web visualization service". In: *Journal of Communications and Networks* 19.4 (2017), pp. 384–391. DOI: 10.1109/JCN.2017.000062.

[23] Amaia Gil, Marco Quartulli, Igor G Olaizola, and Basilio Sierra. "Towards smart data selection from time series using statistical methods". In: *IEEE Access* 9 (2021), pp. 44390–44401. DOI: 10.1109/ACCESS.2021.3066686.

[24] Kanat Tangwongsan, Martin Hirzel, Scott Schneider, and Kun-Lung Wu. "General incremental sliding-window aggregation". In: *Proceedings of the VLDB Endowment* 8.7 (2015), pp. 702–713.

[25] Jon Gjengset, Malte Schwarzkopf, Jonathan Behrens, Lara Timbó Araújo, Martin Ek, Eddie Kohler, M Frans Kaashoek, and Robert Tappan Morris. "Noria: dynamic, partially-stateful data-flow for high-performance web applications." In: *OSDI*. Vol. 18. 2018, pp. 213–231.

[26] Jeroen Van Der Donckt, Jonas Van Der Donckt, and Sofie Van Hoecke. "tsdownsample: high-performance time series downsampling for scalable visualization". In: *arXiv preprint arXiv:2307.05389* (2023).

[27] Zhicheng Liu and Jeffrey Heer. "The effects of interactive latency on exploratory visual analysis". In: *IEEE transactions on visualization and computer graphics* 20.12 (2014), pp. 2122–2131.

[28] Maheswari Visvalingam and James D Whyatt. "Line generalisation by repeated elimination of points". In: *The cartographic journal* 30.1 (1993), pp. 46–51. DOI: 10.1179/000870493786962263.

[29] David H Douglas and Thomas K Peucker. "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature". In: *Cartographica: the international journal for geographic information and geovisualization* 10.2 (1973), pp. 112–122. DOI: 10.3138/FM57-6770-U75U-7727.

B

tsflex - Flexible Time Series Processing & Feature Extraction

Chapter 1 highlighted the heterogeneity of time series data and the importance of processing and feature extraction to standardize and transform multimodal time series into a tabular format. These steps are essential for building successful ML pipelines [1]. This chapter aims to tackle several challenges associated with time series processing and feature extraction by introducing `tsflex`, a high-performance and flexible toolkit, fulfilling **RG2-A**. Notably, `tsflex` is the first toolkit to support multi-window feature extraction.

`tsflex` has been instrumental in various applications, including the machine learning pipeline proposed in our sleep-stage detection paper [2], which demonstrated that, when a well-constructed feature vector is used, linear models can perform on par with most deep learning approaches for sleep stage detection. Additionally, `tsflex` has proven its utility in several Kaggle (data science) competitions [1], and has been a core component in industry collaborations (VLAIO O&O iCosy2, VLAIO O&O Volvo, BILA J&J, imec.icon COSMO, imec.icon nervocity imec.icon CHAI, imec.icon PROTEGO, AAA contextaware). Furthermore, it has supported various research projects and papers [3, 4, 5], even being mentioned on Wikipedia [2].

While `tsflex` is a collaborative achievement with Jeroen Van Der Donckt, my (joint)

[1] In several in-class competitions; Sleep Apnea Detection, Misalignment Fault Detection I, Misalignment Fault Detection II

[2] https://en.wikipedia.org/wiki/Feature_engineering

contributions can be summarized as follows:

- Identifying the requirements for effective time series processing and feature extraction.
- Analyzing existing Python tools to evaluate their suitability for these requirements.
- Developing `tsflex` to address these identified needs, including:
  - Iterative prototyping.
  - Software design & development.
  - Packaging, testing, and documenting the code.
  - Integrations with other tools.
  - Continuous support, upgrades, and bug fixes.
- Benchmarking `tsflex` for runtime and memory efficiency and comparing it to existing solutions.

`tsflex` - Flexible Time Series Processing & Feature Extraction

Jonas Van Der Donckt [3], Jeroen Van Der Donckt [3], Emiel Deprost, and Sofie Van Hoecke

Published in "SoftwareX, Vol. 17, 2022, p. 100971"

Abstract Time series processing and feature extraction are crucial and time-intensive steps in conventional machine learning pipelines. Existing packages are limited in their applicability, as they cannot cope with irregularly-sampled or asynchronous data and make strong assumptions about the data format. Moreover, these packages do not focus on execution speed and memory efficiency, resulting in considerable overhead. We present `tsflex`, a Python toolkit for time series **processing and feature extraction**, that focuses on performance and flexibility, enabling broad applicability. This toolkit leverages window-stride arguments of the same data type as the sequence-index, and maintains the sequence-index through all operations. `tsflex` is **flexible** as it supports (1) multivariate time series, (2) multiple window-stride configurations, and (3) integrates with processing and feature functions from other packages, while (4) making no assumptions about the data sampling regularity, series alignment, and data type. Other functionalities include multiprocessing, detailed execution logging, chunking sequences, and serialization. Benchmarks show that `tsflex` is **faster** and more **memory-efficient** compared to similar packages, while being more permissive and flexible in its utilization. The open-source code can be found at https://github.com/predict-idlab/tsdownsample.

Code Metadata

Code: `github.com/predict-idlab/tsflex`
Documentation: `predict-idlab.github.io/tsflex`
License: `MIT`
Code version: `0.2.3`

B.1 Motivation and Significance

Data-driven modelling and forecasting of time series is a major topic of interest in academic research and industrial applications[6, 7], being a key component in various domains such as climate modelling [8], patient monitoring [9], industrial maintenance [10], and decision-making in finance [11].

[3] Contributed equally

Two traditional steps in machine learning on time series are (pre)processing and feature extraction, often performed in this order. Processing is concerned with cleaning or transforming the raw data, e.g., filtering noise, detrending, clipping outliers, and resampling. Feature extraction aims to extract a set of characteristics, i.e., the features, with the intention of constructing a relevant (lower-dimensional) representation of the data. Both steps are time-consuming and rather complex, yet they are crucial for a successful machine learning pipeline [1].

In many cases the time series measurements might not necessarily be observed at a regular rate or could be unsynchronized [12]. Moreover, the presence or absence of measurements and the varying sampling rate may carry information on its own [13]. Unfortunately, current Python time series packages such as `seglearn` [14], `tsfresh` [15], `TSFEL` [16], and `kats` [17] make strong assumptions about the sampling rate regularity and the alignment of modalities. Furthermore, to the best of our knowledge, no library today supports multiple strided-window feature extraction, varying data types (e.g., handling categorical data), and chunking of (multiple) time series. These observations highlight the need for a flexible processing and feature extraction package. Therefore, we present `tsflex`, a package designed solely concerning these two steps, as it aims to get the fundamentals right. `tsflex` offers, next to custom functions, seamless integration with other data science packages, e.g., processing or feature functions from libraries such as `NumPy` [18], `SciPy` [19], `seglearn` [14], `tsfresh` [15], and `TSFEL` [16], or machine-learning toolkits like `scikit-learn` [20].

`tsflex` can be employed from prototyping machine learning pipelines to deploying real-world time series projects. Currently, we are amongst others using `tsflex` in real-time data pipelines for the *mBrain* study [21]. Here `tsflex` is used for processing and feature extraction of raw sensor data streams in which gaps, irregular sampling rates and large data chunks occur.

The remainder of this chapter is as follows. In Section B.2 we elaborate on the software and its functionality. Next on, Section B.3 provides an illustrative example. Section B.4 stresses the impact of `tsflex` by both positioning our toolkit among existing libraries and benchmarking these libraries against `tsflex`. Finally, we end with a conclusion in Section B.5.

B.2 Software Description

`tsflex` is a Python package that leverages (under the hood) efficient `NumPy` [18] data operations on `pandas` [22] data for (pre)processing and extracting features from time series. We opted for pandas data (either `pd.DataFrame` or `pd.Series`) since this is a convenient format for sequence data, and supports amongst others sequence indexing, integrated column names, and various data types. A direct result of complying with the available `pandas` data types is that `tsflex` allows performing operations on numerical, categorical, boolean, time based, and string-like data.

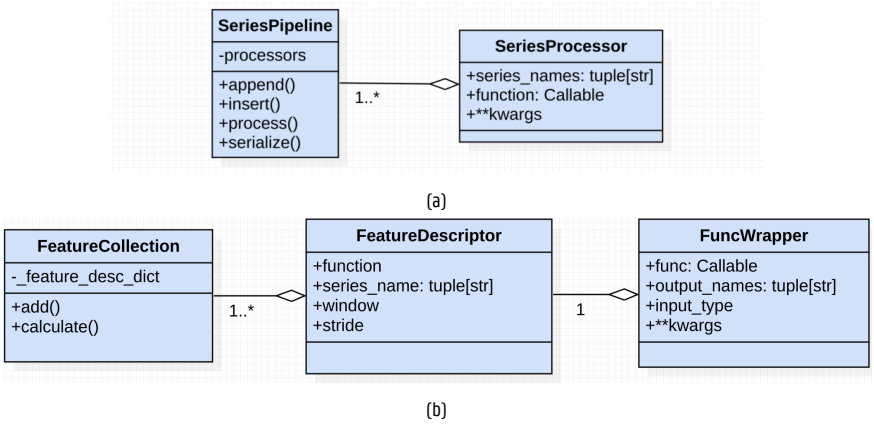


Figure B.1: UML diagram of the (a) `tsflex.processing` and (b) `tsflex.features` submodule.

Users can install `tsflex` by using *pip*; `pip install tsflex`, or *conda*; `conda install -c conda-forge tsflex`. Once installed, our documentation together with various examples should enable the user to apply this toolkit for their purpose.

B.2.1 Software Architecture

`tsflex` consists of two separated entities, i.e., a processing and a feature extraction submodule. The following subsections describe the architecture of both submodules, visually aided by Figure B.1.

Remark that these two submodules work on a different scope. The processing submodule works on full sequences, i.e., full scope, whereas the feature extraction submodule works on strided windows, i.e., restricted scope.

B.2.1.1 Processing Submodule

Figure B.1a depicts the main components of the `tsflex.processing` submodule. The processing functionality is provided by the `SeriesPipeline` which contains one or multiple `SeriesProcessor` steps. The processing steps are applied sequentially on the data that is passed to the processing pipeline. This sequential order is crucial as the processing operations can create new series or update existing ones, which can be used in the succeeding steps, e.g., first applying a filter-processor and in the next steps decomposing that filtered signal. We summarize the objective of each component:

- `SeriesPipeline`: serves as a pipeline, withholding the to-be-sequentially-applied *processing steps*.
- `SeriesProcessor`: an instance of this class describes a *processing step*.

A processing step is defined by a `function` (the *Callable* processing-function), `series_names` (the *name(s)* of the series that should be processed), and `**kwargs` (optional keyword arguments for `function`).

B.2.1.2 Feature Extraction Submodule

Figure B.1b depicts the `tsflex.features` components. The feature extraction functionality is provided by a `FeatureCollection` that contains one or multiple `FeatureDescriptors`. The features are calculated (possibly in parallel) on the data that is passed to the feature collection. We describe the objective of each component:

- `FeatureCollection`: serves as a registry, withholding the to-be-calculated *features*.
- `FeatureDescriptor`: an instance of this class describes a *feature*.

  A feature is defined by a `series_name` (the *name(s)* of the input series on which the feature-function will operate), `function` (the *Callable* feature-function), and `window` and `stride` (the sequence-index based window and stride range).
- `FuncWrapper`: a wrapper around *Callable* functions, intended for advanced feature-function configuration (e.g., customizing feature output-names, passing `**kwargs` to feature functions), and defining the function input data-type (i.e., `numpy.array` or `pandas.Series`).

B.2.2 Software Functionalities

In the sections below, we further detail the processing and feature extraction functionalities, together with other utilities of `tsflex`.

B.2.2.1 Processing

The processing functionality is concerned with either transforming (i.e., replacing) sequences or creating new ones. `tsflex` provides flexible processing by accepting a generic processing function prototype. Such processing functions should take one or multiple sequences as input, followed by optional keyword arguments. This generic processing function prototype enables compatibility with many existing libraries, e.g., `scipy.signal` [19], `statsmodels.tsa` [23][4].

[4] Processing functions can return an arbitrary amount of sequences; `tsflex` supports one-to-one, one-to-many, many-to-one, and many-to-many functions; see https://predict-idlab.github.io/tsflex/processing/index.html#versatile-processing-functions

B.2.2.2 Feature Extraction

The feature extraction functionality is concerned with calculating features on strided-rolling windows. `tsflex` was designed to define the window and stride arguments in the same unit as the sequence-index its data type (e.g., window=*"5min"* and stride=*"30s"* for time-indexed sequences, or window=*300* and stride=*30* for numeric-indexed sequences). As existing libraries define the window and stride in terms of number of samples [14, 15, 16], they implicitly assume that the sampling rate is fixed and there are no gaps. `tsflex`'s *flexibility* is a direct consequence of not making such assumptions; by default, features can be extracted on multivariate time series with varying sampling rates and even gaps[5]. In addition, `tsflex` supports a wide range of feature functions, again enabling compatibility with many existing libraries, e.g., `numpy`, `scipy.stats`, `tsfresh`[6], `seglearn`[6], `tsfel`[6].

B.2.2.3 Other Functionalities

`tsflex` serves various additional functionalities, such as embedded serialization, execution time logging, native support for categorical & time based data, and handling of time series data in chunks. Chunking of sequence data can be performed by calling the `chunk_data` function from the `tsflex.chunking` submodule. Processing & extracting features on chunked data produces lower memory peaks, enabling time series handling in constrained environments (e.g., streaming, edge devices [24]). Additionally, chunking allows parallelizing the sequential processing.

Furthermore, the `FeatureCollection` its `reduce(feat_cols_to_keep)` method returns a new `FeatureCollection` instance, withholding the subset of features that constitute the output column names listed in `feat_cols_to_keep`. Additionally, users can create custom segments over which the features should be calculated by using the `segment_start_idxs` and `segment_end_idxs` parameters of the `FeatureCollection` its `calculate` method. A final noteworthy functionality is the support for vectorized feature functions, i.e., applying a feature function over a 2D—or even 3D in case of multiple input time series—segmented view of the data.

Unit Tests The provided functionalities of `tsflex` are extensively tested through unit testing. For example, these tests assure that the functions should perform view-based operations, that `tsflex` handles categorical and time-based data, and that feature-functions are not allowed to change the view-based input data. Every claim about `tsflex` we make in this chapter is backed by unit testing.

[5] It is the feature-function its responsibility to handle such cases correctly. Note that a feature-function can easily be made robust using the `make_robust` wrapper from `tsflex.features.utils`.

[6] For more details on how `tsflex` integrates with existing libraries, consult our integration notebooks.

B.2.3 Limitations

Currently, there is no agreed standard for time series in Python [25]. The main cause for this disagreement is that each format has its own benefits and disadvantages. An in-depth discussion about this topic is out of scope for this chapter. For `tsflex`, we made the design decision to operate on single-indexed wide/flat data (such as a list of series or a wide-dataframe) whose index represents the sequence-position. In our opinion, this data format is most intuitive to wrangle with, e.g., slicing, visualizing, processing. Therefore, two limitations of `tsflex` are that it (1) does not support long data, nor (2) multi-indexed data (and columns). Remark that a long-dataframe can be transformed into a list of series (that has the same in-memory size). A third limitation is that `tsflex` uses sequence-names as identifiers, resulting in the assumption that each sequence should have a unique name.

B.3 Illustrative Examples

As illustrative example, we provide three snippets containing working code [7].

```
from tsflex.utils.data import load_empatica_data
df_tmp, df_acc, df_ibi = load_empatica_data(["tmp","acc","ibi"])
```

Listing 1: Data loading code.

Listing 1 fetches the data for the examples. In total, three `pd.DataFrames` are loaded, containing multimodal data of different sampling rates. This data is an excerpt of a wrist-worn wearable from the WESAD study [26]. The characteristics of the dataframes are summarized in Table B.1. The `df_tmp` dataframe withholds skin temperature data, `df_acc` withholds accelerometer data along the 3 movement axes, and `df_ibi` contains the interbeat interval (IBI) data, representing the time between two consecutive heartbeats. Remark that IBI data is only available when two consecutive, successfully detected beats took place, making IBI an irregularly sampled series.

Table B.1: Properties of the data used in the examples.

	columns	**shape**	**sampling rate**
df_tmp	[TMP]	(30200, 1)	4.0 Hz
df_acc	[ACC_x, ACC_y, ACC_z]	(241620, 3)	32.0 Hz
df_ibi	[IBI]	(1230, 1)	Irregularly sampled

[7] Online version: https://github.com/predict-idlab/tsflex/blob/main/examples/tsflex_paper.ipynb.

```
import pandas as pd; import numpy as np
from scipy.signal import savgol_filter
from tsflex.processing import SeriesProcessor, SeriesPipeline

# Create the processing functions
def clip_data(sig: pd.Series, min_val=None, max_val=None):
    return np.clip(sig,  a_min=min_val, a_max=max_val)

def smv(*sigs) -> pd.Series:
    sig_prefixes = set(s.name.split('_')[0] for s in sigs)
    res = np.sqrt(np.sum([np.square(s) for s in sigs], axis=0))
    return pd.Series(res, index=sigs[0].index, name="|".join(sig_prefixes)+"_SMV")

# Create the series processors (with their keyword arguments)
tmp_clippper = SeriesProcessor(clip_data, series_names="TMP", max_val=35)
acc_savgol = SeriesProcessor(
    savgol_filter, ["ACC_x", "ACC_y", "ACC_z"], window_length=33, polyorder=2
)
acc_smv = SeriesProcessor(smv, ("ACC_x", "ACC_y", "ACC_z"))

# Create the series pipeline & process the data
series_pipe = SeriesPipeline([tmp_clippper, acc_savgol, acc_smv])
out_data = series_pipe.process([df_acc, df_tmp, df_ibi])
```

Listing 2: Processing example. Continuation of code snippet 1.

B.3.1 Processing

Listing 2 shows how various processing steps are applied on the loaded data. For each processing step a `SeriesProcessor` object is created, which records the series names[8] (i.e., the names of the sequences that should be processed) and the optional keyword arguments. Observe that the `smv` function creates a new series.

B.3.2 Feature Extraction

Listing 3 shows how feature extraction can be performed on the previously processed data. Two `MultipleFeatureDescriptors`[9] are created; the first defines some general statistical and spectral features on the `ACC_SMV` and `TMP` signal for two different windows, and the second defines a robust version[10] of some statistical features (and the number of samples) for the `IBI` signal. Remark that in `general_feats`, `seglearn` feature-functions are imported and wrapped in a convenient manner. These two descriptor objects are enclosed in a feature collection, which is used for extracting (i.e., calculating) the features. The `approve_sparsity` flag enables the user to explicitly acknowledge that there might be sparse data, i.e., irregularly sampled data. Setting this

[8] When a processing function should be applied on multiple series, a list should be passed to the `series_names` argument. When a processing function handles multiple series as input, a tuple (or a list thereof) should be passed to the `series_names` argument.

[9] `MultipleFeatureDescriptors` are a convenient way to define features containing multiple functions, series names, windows, and strides.

[10] It is important to make the feature functions robust as there may be empty windows for the `IBI` data. In such cases, the `make_robust` wrapper avoids that an error is thrown and returns `NaN` instead.

```
from tsflex.features import MultipleFeatureDescriptors, FeatureCollection
from tsflex.features.integrations import seglearn_feature_dict_wrapper
from tsflex.features.utils import make_robust

# Import / create the feature functions
from seglearn.feature_functions import base_features
def area(sig: np.ndarray) -> float:
    return np.sum(np.abs(sig))

# Create the feature descriptors
general_feats = MultipleFeatureDescriptors(
    functions=seglearn_feature_dict_wrapper(base_features()) + [area],
    series_names=["ACC_SMV", "TMP"],
    windows=["5min", "2.5min"],
    strides="2min",
)
ibi_feats = MultipleFeatureDescriptors(
    [make_robust(f) for f in [np.min, np.max, np.mean, np.std]] + [len],
    series_names="IBI",
    windows="5min",
    strides="2min"
)

# Create the feature collection & calculate the features
fc = FeatureCollection([general_feats, ibi_feats])
feat_df = fc.calculate(out_data, return_df=True, approve_sparsity=True)
```

Listing 3: Feature extraction example. Continuation of code snippet 2.

flag avoids warnings that are raised in case of sparsity.

B.4 Impact

We first indicate the impact of `tsflex` by positioning it among other packages. Then, we present `tsflex`'s performance in terms of memory usage and computation time, and compare with related packages. We conclude with some examples and references to notebooks which highlight the cross-domain applicability of `tsflex` for time series.

B.4.1 Functionalities

Irregularly sampled data is ubiquitous. However, most existing time series toolkits assume that either the user segments the data in valid chunks or that the data is regularly sampled. The former induces a significant user burden, whilst the latter is a fairly strong assumption. By employing a `sequence range based` window-stride approach and thus not a *sample based* one, `tsflex` interoperates natively with irregularly sampled sequence data. We position such functionalities of `tsflex` against other related packages in Table B.2. Remark that `tsflex` is the only package that (1) allows defining multiple window-stride combinations, (2) can operate on non-numerical data, and (3) serves time-based chunking functionalities. Moreover, except for `tsfresh`, `tsflex` is the only other library that maintains the index of the data, encouraging index based

analysis of the obtained outputs. We refer to example notebooks for more concrete illustrations of these functionalities.

Table B.2: Comparison of `tsflex` against other relevant packages. The "X-*to*-Y *functions*" in the Properties column with $X, Y \in \{one, many\}$ represent the feature input-to-output relationship; X="one" denotes single-series input, whereas X="many" represents multivariate inputs. When Y="one" a single feature is returned, whilst the Y="many" returns multiple features. More info about these versatile functions can be found here. An online version of this table is shown here.

Properties	**tsflex**	**seglearn**	**tsfresh**	**TSFEL**	**kats**
General					
Time column requirements	Any sortable	Any sorted	Any sortable	Any sorted	Datetime index
Multivariate time series	✓	✓	✓	✓	✓
Unevenly sampled data	✓	✗	✗	✗	✓
Time-column maintenance	✓	✗	✓	✗	✗
Retain output names	✓	✓	✓	✓	✗
Multiprocessing	✓	✗	✓	✓	✗
Operation execution time logging	✓	✗	✗	✗	✗
Chunking (multiple) time series	✓	✗	✗	✗	✗
Feature extraction					
Strided-window definition format	Sequence index range	Sample-based	Sample-based	Sample-based	Na.
Strided-window feature extraction	✓	✓	✓	✓	✗
Multiple stride-window combinations	✓	✗	✗	✗	✗
Custom features	✓	✓	✓	✓	✗
One-to-one functions	✓	✓	✓	✓	✓
One-to-many functions	✓	✓	✓	✓	✓
Many-to-one functions	✓	✓	✗	✗	✗
Many-to-many functions	✓	✗	✗	✗	✗
Categorical data	✓	✗	✗	✗	✗
Data type preservation	✓	✗	✗	✗	✗

B.4.2 Feature Extraction Performance

Considering all Python toolkits, eligible for strided-rolling feature extraction [14, 15, 16], only `seglearn` mentions toolkit-performance by comparing their computation time and model accuracy with other packages. However, for real-world applicability, computational efficiency is of utmost importance. Therefore, we benchmarked `tsflex` its memory usage and runtime against other libraries and made the benchmarking codebase open source at this repository[11] to encourage effortless benchmarking of `tsflex` on other use cases (e.g., edge devices, extremely large datasets, streaming use cases).

[11] We decided to only benchmark feature extraction, as this is the most advanced functionality of `tsflex`. In our experience, the processing functionality is rather straightforward and thus more dependent on the processing functions. However, empirical results indicate that we have a significant efficiency advantage over other existing packages when parallel processing is performed on chunked data.

Profiling is realized by using the `VizTracer` [27] package with the `VizPlugins` add-on. The benchmark dataset is a synthetically generated dataframe consisting of 5 channels and spans 1 hour. Its values have the numerical `numpy.float32` data type. To comply with the assumptions that other toolkits make, each modality is sampled at 1000Hz and does not contain gaps. The toolkit are configured to extract the same features using a window-stride of 30s-10s, respectively. The benchmark process follows these steps for each toolkit-feature-extraction configuration:

1. Each toolkit feature extraction script is called 20 times to average out the memory usage and runtime[12].
2. Script execution:
   (a) Construct the synthetic `pd.DataFrame` benchmark data
   (b) `VizTracer` starts logging
   (c) Create the feature extraction configuration
   (d) Extract and store the features
   (e) `VizTracer` stops logging
   (f) Write the `VizTracer` profile-results to a JSON-file

The profile JSONs were collected on a server with an *Intel Xeon E5-2650 v2 @ 2.60GHz* CPU and SAMSUNG M393B1G73QH0-CMA DDR3 1600MT/s RAM, with *Ubuntu 18.04.5 LTS x86_64* as operating system. Other running processes were limited to a minimum.

Figure B.2 depicts the aggregated JSON-file results, and Table B.3 summarizes the main outcomes of this visualization. We further complement these results with Figure B.3, where the investigated toolkits their performance is compared to `tsfresh` (which can be considered as the industry standard). For this use case, `tsflex` is $\sim 3\times$ faster than its closest competitor in both the sequential and multiprocessing variant. The peak memory usage is of particular interest, as this determines the minimum amount of RAM a system should have. `tsflex` and `TSFEL` apply view-based operations on the data, making them significantly more memory efficient than other packages. Here again, `tsflex` requires $\sim 2.5\times$ less memory than `TSFEL`. Note that `tsfresh` first expands the data into a tsfresh-compatible format before applying feature extraction. This results in a slope in the logarithmic domain from second 15 to second 80-150.

B.4.3 Applicability

`tsflex` is a domain-independent package, enabling broad applicability[13]. For example, this package is already used in multiple research projects such as wearable-based stress

[12] Remark that by recalling the script in separate runs, no caching or memory is shared among executions.
[13] The cross-domain applicability is highlighted by the examples: https://github.com/predict-idlab/tsflex/tree/main/examples

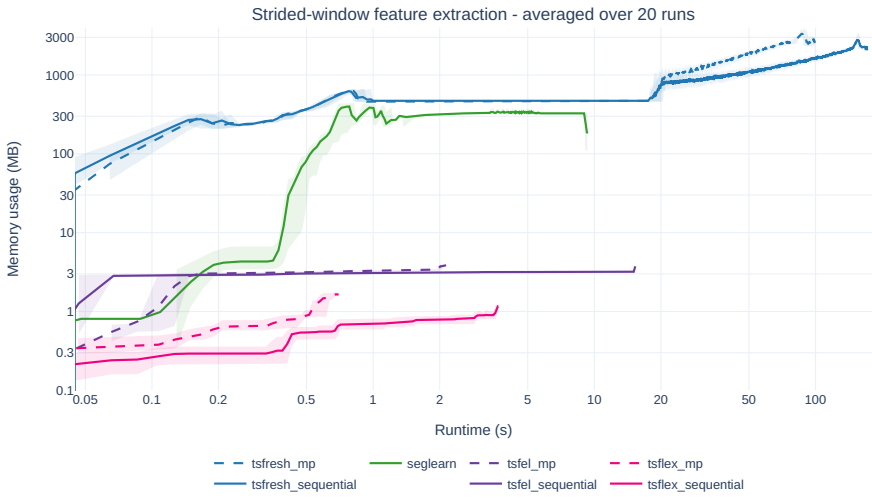


Figure B.2: Average memory usage over time for a feature extraction task on evenly sampled data with a fixed window and stride. The origin of the runtime and memory usage axis starts directly after the synthetic data was constructed; the feature extraction configuration is then initialized and called on the data. As noted in Table B.2, only `seglearn v1.2.3` [14], `tsfresh v0.18.0` [15], and `TSFEL v0.1.4` [16] support defining a (sample-based) window and stride, making this comparison fair as the data for this benchmark is evenly sampled. For reference, the allocated memory for the data was $96.4MB$. An interactive version (where you can switch to linear axes) of this figure is shown here.

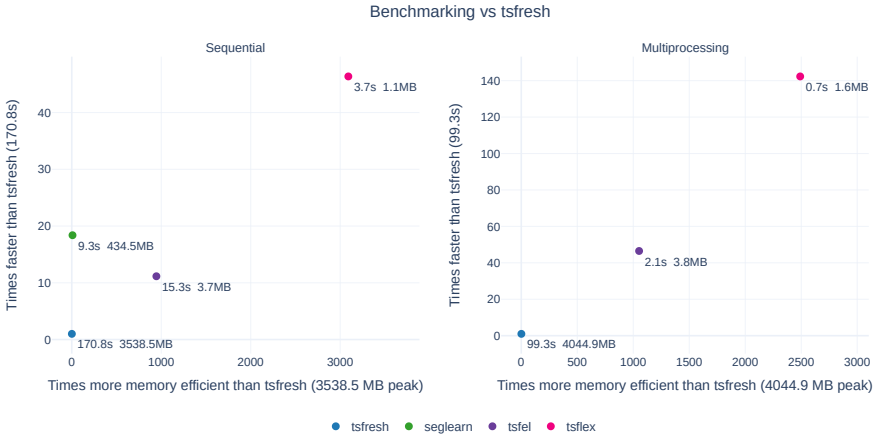


Figure B.3: Normalized runtime (y-axis) and memory peak (x-axis), for the benchmarks presented in Figure B.2, with respect to `tsfresh`. The data points on the chart represent performance improvements in multiples of `tsfresh`'s performance, where higher values are indicative of better performance (on both axes). The left plot illustrates the performance comparison for sequential execution, whereas the right plot showcases the multiprocessing execution.

Table B.3: Tabular summary of `VizTracer` benchmarks, depicted in Figure B.2.

	tsflex	**TSFEL**	**seglearn**	**tsfresh**
mean peak memory usage (MB ± std)				
sequential	**1.3 ± 0.1**	3.5 ± 0.3	435.3 ± 1.5	3540 ± 13.9
multiprocessing	**1.5 ± 0.1**	3.7 ± 0.1	/	4044 ± 14.4
mean runtime (s ± std)				
sequential	**4.3 ± 0.1**	16.4 ± 0.8	9.2 ± 0.1	169.8 ± 1.6
multiprocessing	**0.7 ± 0.0**	2.1 ± 0.0	/	98.5 ± 1.2

monitoring, automatic sleep staging, occupancy detection in buildings, and anomaly detection. `tsflex`'s computational efficiency (in both execution time and memory usage) also paves the way towards applicability in constrained environments, such as streaming or edge computing [28, 24].

B.5 Conclusion

Time series processing and feature extraction are arguably the most important steps in classical machine learning pipelines. However, existing packages are limited in their applicability as they make strong assumptions about the underlying data types and data structure. Furthermore, these toolkits do not prioritize memory and runtime efficiency, creating unnecessary overheads. These existing packages also tend to focus on including numerous feature functions instead of conveniently integrating with other libraries. We argue that there is a need for a more permissive toolkit, which concentrates on the essentials. Therefore, we present `tsflex`, a Python package that focuses on processing and feature extraction for time series. This chapter describes the functionalities and performance of `tsflex` and compares it to other packages. We show that `tsflex` is more permissive than existing Python toolkits, and benchmarking indicates it is over 50% more efficient than comparable work in both runtime and memory usage. The increased flexibility is realized by leveraging sequence-index based arguments and is reflected in the few assumptions that this library makes. We believe that `tsflex`'s integration with other libraries, together with its advanced functionalities, e.g., chunking, comprehensible feature output names, enables real-world, cross-domain applicability.

References

[1] Pedro Domingos. “A few useful things to know about machine learning”. In: *Communications of the ACM* 55.10 (2012), pp. 78–87.

[2] Jeroen Van Der Donckt, Jonas Van Der Donckt, Emiel Deprost, Nicolas Vandenbussche, Michael Rademaker, Gilles Vandewiele, and Sofie Van Hoecke. “Do not sleep on traditional machine learning: Simple and interpretable techniques are competitive to deep learning for sleep scoring”. In: *Biomedical Signal Processing and Control* 81 (2023), p. 104429.

[3] Irfan Al-Hussaini and Cassie S Mitchell. “SeizFt: Interpretable Machine Learning for Seizure Detection Using Wearables”. In: *Bioengineering* 10.8 (2023), p. 918.

[4] Lars Leyendecker, Milena Zuric, Muhammad Atique Nazar, Karl Johannes, and Robert H Schmitt. “Predictive Quality Modeling for Ultra-Short-Pulse Laser Structuring utilizing Machine Learning”. In: *Procedia CIRP* 117 (2023), pp. 275–280.

[5] Jonas Van Der Donckt, Mathias De Brouwer, Pieter Moens, Marija Stojchevska, Bram Steenwinckel, Stef Pletinck, Nicolas Vandenbussche, Annelis Goris, Koen Paemeleire, Femke Ongenae, et al. “From self-reporting to monitoring for improved migraine management”. In: *Engineer meets Physician (EmP)*. 2022.

[6] Francisco J Montáns, Francisco Chinesta, Rafael Gómez-Bombarelli, and J Nathan Kutz. “Data-driven modeling and learning in science and engineering”. In: *Comptes Rendus Mécanique* 347.11 (2019), pp. 845–855.

[7] Ricardo P Masini, Marcelo C Medeiros, and Eduardo F Mendes. “Machine learning advances for time series forecasting”. In: *Journal of economic surveys* 37.1 (2023), pp. 76–111.

[8] Olivier Pieters, Emiel Deprost, Jonas Van Der Donckt, Lore Brosens, Pieter Sanczuk, Pieter Vangansbeke, Tom De Swaef, Pieter De Frenne, et al. “MIRRA: A Modular and Cost-Effective Microclimate Monitoring System for Real-Time Remote Applications”. In: *Sensors* 21.13 (2021), p. 4615.

[9] Eric J Topol. “High-performance medicine: the convergence of human and artificial intelligence”. In: *Nature medicine* 25.1 (2019), pp. 44–56.

[10] Andrew A Cook, Göksel Mısırlı, and Zhong Fan. “Anomaly detection for IoT time-series data: A survey”. In: *IEEE Internet of Things Journal* 7.7 (2019), pp. 6481–6494.

[11] Stephen J Taylor. *Modelling financial time series*. world scientific, 2008.

[12] Pranjul Yadav, Michael Steinbach, Vipin Kumar, and Gyorgy Simon. “Mining electronic health records (EHRs) A survey”. In: *ACM Computing Surveys (CSUR)* 50.6 (2018), pp. 1–40.

[13] Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*. Vol. 793. John Wiley & Sons, 2019.

[14] David M Burns and Cari M Whyne. "Seglearn: a python package for learning sequences and time series". In: *The Journal of Machine Learning Research* 19.1 (2018), pp. 3238–3244.

[15] Maximilian Christ, Nils Braun, Julius Neuffer, and Andreas W Kempa-Liehr. "Time series feature extraction on basis of scalable hypothesis tests (tsfresh–a python package)". In: *Neurocomputing* 307 (2018), pp. 72–77.

[16] Marília Barandas, Duarte Folgado, Letícia Fernandes, Sara Santos, Mariana Abreu, Patrícia Bota, Hui Liu, Tanja Schultz, and Hugo Gamboa. "Tsfel: Time series feature extraction library". In: *SoftwareX* 11 (2020), p. 100456.

[17] Facebook Research. *Kats - One stop schop for time series analysis in Python*. en. 2021. URL: https://facebookresearch.github.io/Kats/ (visited on 06/30/2021).

[18] Charles R. Harris et al. "Array programming with NumPy". In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[19] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[20] Lars Buitinck et al. "API design for machine learning software: experiences from the scikit-learn project". In: *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*. 2013, pp. 108–122.

[21] Mathias De Brouwer et al. "mBrain: towards the continuous follow-up & headache classification of primary headache disorder patients". en. In: *BMC Medical Informatics and Decision Making* (2021).

[22] The pandas development team. *pandas-dev/pandas: Pandas*. Version latest. Feb. 2020. DOI: 10.5281/zenodo.3509134. URL: https://doi.org/10.5281/zenodo.3509134.

[23] Skipper Seabold and Josef Perktold. "Statsmodels: Econometric and statistical modeling with python". In: *Proceedings of the 9th Python in Science Conference*. Vol. 57. Austin, TX. 2010, p. 61.

[24] Weisong Shi and Schahram Dustdar. "The promise of edge computing". In: *Computer* 49.5 (2016), pp. 78–81.

[25] Maximilian Christ. *Awesome time series in Python - standardize time series formats*. 2020. URL: https://github.com/MaxBenChrist/awesome_time_series_in_python/blob/master/standardize_time_series_formats.md (visited on 11/15/2021).

[26] Philip Schmidt, Attila Reiss, Robert Duerichen, Claus Marberger, and Kristof Van Laerhoven. "Introducing wesad, a multimodal dataset for wearable stress and affect detection". In: *Proceedings of the 20th ACM international conference on multimodal interaction*. 2018, pp. 400–408.

[27] Tian Gao. *VizTracer: a low-overhead logging, debugging, and profiling tool to trace and visualize python code execution.* 2020. URL: https://github.com/gaogaotiantian/viztracer/.

[28] V Kavitha and M Punithavalli. "Clustering time series data stream-a literature survey". In: *arXiv preprint arXiv:1005.4270* (2010).