Calibrated Multi-Probabilistic Prediction as a Defense against Adversarial Attacks*

Jonathan Peck[1,2], Bart Goossens[3], and Yvan Saeys[1,2]

[1] Department of Applied Mathematics, Computer Science and Statistics, Ghent University, Ghent, 9000, Belgium
[2] Data Mining and Modeling for Biomedicine, VIB Inflammation Research Center, Ghent, 9052, Belgium
[3] Department of Telecommunications and Information Processing, IMEC/Ghent University, Ghent, 9000, Belgium

Abstract. We propose the *MultIVAP*, a scalable technique for hedging the predictions of any machine learning classifier. The algorithm incurs a reasonably small computational overhead and is able to significantly increase the robustness of the underlying model to adversarial perturbations without sacrificing accuracy. This increase in robustness is experimentally confirmed against defense-oblivious attacks as well as a white-box attack specifically designed for the MultIVAP. Code is available at `https://github.com/saeyslab/multivap`.

Keywords: machine learning · adversarial robustness · conformal prediction

1 Introduction

Machine learning techniques have made great progress in recent years, obtaining state of the art performance in areas such as natural language processing [25] as well as image and speech recognition [23]. However, the theoretical properties of the deep neural networks responsible for this success remain poorly understood. At present, there is no theory which can satisfactorily explain the success of deep learning and many open questions remain [32]. A peculiar example of this lack of theoretical understanding is the existence of so-called *adversarial perturbations* [3]. These are small modifications to the inputs of a model which can drastically change its output, even though the alterations are completely insignificant. This is perhaps nowhere as apparent as in image recognition, which is where the phenomenon was first studied for deep neural networks [27]. Figure 1 shows an example of an adversarial perturbation in this setting. Recently,

* We thank the NVIDIA Corporation for the donation of a Titan Xp GPU with which we were able to carry out our experiments. Jonathan Peck is sponsored by a fellowship of the Research Foundation Flanders (FWO). Yvan Saeys is an ISAC Marylou Ingram scholar.

significant progress has been made towards generating adversarial examples for domains other than images. This includes speech recognition, where inaudible distortions of spoken fragments can lead to erroneous transcriptions [20], and text processing where small typographical errors can mislead the models [8].

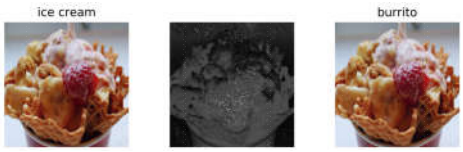

Fig. 1: Example of an adversarial perturbation for image classification models. The image on the left is correctly classified as ice cream by the ResNet50 deep neural network from He et al. [11]. When the perturbation shown in the middle is added to this image, we obtain the sample on the right. Although visually indistinguishable, the altered image is classified as a burrito instead.

As pointed out by Biggio and Roli [3], adversarial examples already have a long history. However, to date, it appears there is no consensus regarding the causes of this vulnerability in modern deep neural networks nor do there appear to exist satisfactory solutions. Despite intense research effort, at present there does not exist any real defense against this phenomenon; almost all serious defenses which have been proposed thus far have either been broken completely or have difficulties scaling to realistic problems [2].

In this work, we propose a novel defense against adversarial manipulation which aims to scale to realistic problems and provide non-trivial robustness. It is based on methods from *conformal prediction* and therefore enjoys frequentist guarantees of validity [29]. Empirical evaluations as well as theoretical results also support the idea that our defense can be scaled to realistic models.

1.1 The calibration hypothesis

The basic idea behind our method of defense is what we call the *calibration hypothesis*. This hypothesis explains the existence of adversarial examples as a consequence of overconfident extrapolation by current machine learning models. This explanation can also be found, for example, in De Vries et al. [7]. More formally, the calibration hypothesis states that the predictions of our models are not *calibrated*. To understand precisely what this means, consider a typical machine learning model for multiclass classification. Such a model often works by estimating probabilities $p_i(x) \approx \Pr[Y = i \mid X = x]$, that is, the probability that a given sample x belongs to class i. The final prediction is then given by $\hat{y}(x) = \operatorname{argmax}_i \ p_i(x)$. These are *point predictions* in the sense that only a single output $\hat{y}(x) \in \mathcal{Y}$ is given. If any measure of uncertainty accompanies these predictions at all, it is usually the estimated class probability $p_i(x)$. However,

as discussed in De Vries et al. [7], these probabilities can be badly mistaken: often there is no theoretical guarantee that the estimated probability $p_i(x)$ will approximate the true probability $\Pr[Y = i \mid X = x]$ to any degree of accuracy. Although it does not constitute a proof of any kind, given this information it is not surprising that adversarial examples exist: one can easily imagine an adversarial attack which exploits miscalibration in order to make the model predict wrong classes with high confidence.

Conformal prediction is a field of machine learning that is dedicated to precisely this problem of finding predictors which have provable guarantees on the reliability of their predictions [29]. If one accepts the calibration hypothesis, i.e., the idea that adversarial examples exploit approximation errors in the estimated class probabilities, then conformal predictors are a natural choice to defend against them.

1.2 Related work

To the best of our knowledge, aside from our own work in Peck et al. [19], little prior work has considered using conformal prediction as a defense against adversarial attacks. Papernot and McDaniel [18] propose a deep k-nearest neighbor classification scheme inspired by the conformal prediction algorithm from Vovk et al. [29]. Not many evaluations of the robustness of this method appear to have been performed thus far, however: we are only aware of the work of Sitawarin and Wagner [24] in this regard, who develop a moderately strong attack against the scheme. Moreover, as this method requires a k-nearest neighbor search in high-dimensional spaces, its computational efficiency is questionable. Card et al. [5] augment existing classifiers with a novel non-conformity measure in order to improve robustness to out-of-sample data, but they do not specifically evaluate this method against adversarial attacks and so its robustness against worst-case manipulations is unclear. Other work regarding defense against adversarial perturbations focuses on robust optimization [14, 21], but these approaches are difficult to scale to large networks.

1.3 Organization

The rest of this paper is organized as follows. In section 2, we give a brief introduction to conformal prediction, in order to establish the necessary background. Section 3 introduces our proposed defense, which we have called the *MultIVAP* algorithm. Experiments with this algorithm are carried out in section 4. Finally, section 5 concludes the work.

2 Conformal prediction

As its name suggests, the field of conformal prediction is concerned with making predictions based on how well a previously unseen sample "conforms" to the data that has already been seen [29]. Formally, a *conformal predictor* is a

function Γ which takes a confidence level $\varepsilon \in [0,1]$, a bag[4] of instances $B = \{\!(x_1, y_1), \ldots, (x_n, y_n)\}\!$ and a new object $x \in \mathcal{X}$ and outputs a set $\Gamma^\varepsilon(B, x) \subseteq \mathcal{Y}$ of possible labels. Intuitively, the set $\Gamma^\varepsilon(B, x)$ consists of those labels which the predictor believes might be the true label with a probability of at least $1 - \varepsilon$. This property is called *exact validity* [29]. Formally, exact validity can be defined as follows. Let $\omega = (x_1, y_1), (x_2, y_2), \ldots$ be an infinite sequence of samples from an exchangeable[5] distribution. Define the error after n samples at significance level ε as

$$\mathrm{err}_n^\varepsilon(\Gamma, \omega) = \begin{cases} 1 & \text{if } y_n \notin \Gamma^\varepsilon(\{\!(x_1, y_1), \ldots, (x_{n-1}, y_{n-1})\}\!, x_n), \\ 0 & \text{otherwise.} \end{cases}$$

Exact validity means that the random variables $\mathrm{err}_1^\varepsilon(\Gamma, \omega), \mathrm{err}_2^\varepsilon(\Gamma, \omega), \ldots$ are all independent and Bernoulli distributed with parameter ε. For *deterministic* predictors, however, this property can never be satisfied. Therefore, in the deterministic case, we only demand that the errors are dominated in distribution by a sequence of independent Bernoulli distributed random variables. This property is called *conservative validity* and implies the following asymptotic result by the law of large numbers:

$$\lim_{n \to \infty} \frac{1}{n} \sum_{i=1}^{n} \mathrm{err}_i^\varepsilon(\Gamma, \omega) \leq \varepsilon \text{ almost surely.}$$

Hence, for deterministic conformal predictors, one can still say that the fraction of mistakes they make tends to be bounded by ε as n grows large.

2.1 Inductive Venn-ABERS predictors

In this work, we will mainly be interested in a particular conservatively valid predictor known as the *inductive Venn-ABERS predictor* (IVAP). First proposed by Vovk et al. [30], an IVAP works by taking advantage of another inductive learning rule (such as a neural network) and calibrating its output in order to hedge the predictions of that rule.

The IVAP is designed only for *binary* classification problems. When instantiated with a scoring rule and an additional calibration data set, it outputs two scalars $p_0, p_1 \in [0,1]$ with $p_0 \leq p_1$ for each new prediction. The theoretical guarantee enjoyed by the IVAP is that these quantities form bounds on the conditional probability that the true label is 1 given the input:

$$p_0(x) \leq \Pr[Y = 1 \mid X = x] \leq p_1(x).$$

In prior work, we showed that it is possible to use IVAPs to detect adversarial manipulation of inputs for binary classification tasks [19]. Our goal here is to

[4] A *bag* or *multiset* is a collection of objects where the order is irrelevant (like a set) but duplicates are allowed (like a list).

[5] A probability distribution is said to be *exchangeable* if every permutation of a sequence is equally likely.

extend this work to the multiclass case and to construct an algorithm which provides stronger robustness and higher accuracy than the method we proposed previously.

3 MultIVAP

The IVAP is only designed for *binary* classification tasks and cannot directly work with multiclass problems. There exist two common methods for extending binary classifiers to the multiclass case, known as *one-vs-one (OvO)* and *one-vs-all (OvA)*. To keep the computational overhead to a minimum, we opt for the OvA strategy here with a particular choice of aggregation method which will give us certain theoretical calibration guarantees. The full pseudocode is given in algorithm 1, which we have called the *MultIVAP*. Consistent with the OvA strategy, the MultIVAP essentially works by fitting K IVAPs, one for each class, where the ith IVAP must decide whether a sample belongs to class i or not. The MultIVAP then takes the lower and upper bounds output by each of the IVAPs, $\left(p_0^{(1)}, p_1^{(1)}\right), \ldots, \left(p_0^{(K)}, p_1^{(K)}\right)$, and solves the optimization problem (MultIVAP) described below. The point of this optimization is to assign a (possibly empty) set of labels $V \subseteq \mathcal{Y}$ to the sample x such that certain probabilistic guarantees can be given on the result. Specifically, we consider a stochastic set L consisting of all labels which one can assign to X. We interpret the probabilities given by the IVAPs as lower and upper bounds on the membership probability of a label in L:

$$p_0^{(i)} \leq \Pr[i \in L \mid X] \leq p_1^{(i)}.$$

Our objective now is to find the largest set of labels $V \subseteq \mathcal{Y}$ such that $\Pr[V \subseteq L \mid X]$ is maximized. To this end, we note that

$$\Pr[V \subseteq L \mid X] \geq \sum_{i \in V} p_0^{(i)} - (|V| - 1).$$

Furthermore,

$$\Pr[V \subseteq L \mid X] \leq \min_{i \in V} \; p_1^{(i)}.$$

Fix an $\varepsilon \in [0, 1]$ and let $\alpha_i = \mathbb{1}[i \in V]$. In order for $\Pr[V \subseteq L \mid X] \geq \varepsilon$ to hold, by the above it is sufficient to have

$$\sum_{i=1}^{K} \alpha_i \left(p_0^{(i)} - 1\right) \geq \varepsilon - 1. \tag{1}$$

In order to maximize the upper bound on $\Pr[V \subseteq L \mid X]$, it is necessary to maximize the smallest $p_1^{(i)}$ over all $i \in V$. That is,

$$\Pr[V \subseteq L \mid X] \leq \min_{i=1,\ldots,K} \; 1 + \alpha_i \left(p_1^{(i)} - 1\right). \tag{2}$$

Our objective is now formally to maximize the cardinality of V and the upper bound (2) while maintaining the constraint (1). This leads us to the following mixed integer linear program (MILP):

$$\begin{aligned} \underset{\alpha_1,\ldots,\alpha_K,\,q}{\text{maximize}} \quad & \alpha_1 + \cdots + \alpha_K + q \\ \text{subject to} \quad & \alpha_1, \ldots, \alpha_K \in \{0,1\}, \\ & q \in [0,1], \\ & q + \alpha_i \left(1 - p_1^{(i)}\right) \leq 1, \qquad i = 1, \ldots, K, \\ & \sum_{i=1}^{K} \alpha_i \left(p_0^{(i)} - 1\right) \geq \varepsilon - 1. \end{aligned} \tag{MultIVAP}$$

Here, the binary variables α_i determine which labels get included in the solution V and q is a dummy scalar which represents the smallest $p_1^{(i)}$ over all $i \in V$. Every solution V to (MultIVAP) satisfies the bounds

$$\varepsilon \leq \Pr[V \subseteq L \mid X] \leq \begin{cases} \min_{i \in V} \; p_1^{(i)} & \text{if } V \text{ is non-empty,} \\ 1 & \text{otherwise.} \end{cases}$$

Note that at least one feasible solution to (MultIVAP) always exists, namely $\alpha_1 = \cdots = \alpha_K = 0$ and $q = 1$. This corresponds to taking the empty set $V = \emptyset$ as the prediction for L, which of course satisfies the trivial guarantee $\Pr[\emptyset \subseteq L \mid X] = 1$. If no better solution than the empty set exists, then the prediction is rejected at the ε significance level.

3.1 Computational complexity

There is still the question of the computational efficiency of algorithm 1. Note that we can split the algorithm into two distinct stages:

1. *Calibration.* When the MultIVAP is first initialized, it is instantiated with a learning algorithm and a training data set. During calibration, it runs the learning algorithm and then runs the IVAP algorithm K times, once for each class.
2. *Inference.* Once the MultIVAP is fully initialized, new samples can be processed using the precomputed isotonic regressions to obtain the probabilistic bounds. Then, the final prediction is computed by solving a mixed integer linear program.

The complexity of the inference step is hard to quantify due to the solution of an optimization problem; we perform timing experiments in section 4 in order to empirically estimate the overhead in this phase. It is easy to see, however, that the overhead incurred in the calibration step is $\mathcal{O}(Kc \log c)$ where c is the size

Algorithm 1: The MultIVAP algorithm.

Input: bag of examples $\wr z_1, \ldots, z_n \wr \subseteq \mathcal{Z}$, object $x \in \mathcal{X}$, learning algorithm A, significance level ε

Output: subset of labels from $\{1, \ldots, K\}$

Divide the bag of examples into a proper training set $\{z_1, \ldots, z_m\}$ and a calibration set $\{z_{m+1}, \ldots, z_n\}$.

Run the learning algorithm A on the proper training set to obtain a scoring classifier F.

for *i from 1 to K* **do**

  Let F_i be the score assigned by F to the ith class.

  Create a new calibration set $S_i = \{(x_{m+1}, y'_{m+1}), \ldots, (x_n, y'_n)\}$ where $y'_j = \mathbb{1}[y_j = i]$.

  Use an IVAP to obtain probabilities $p_0^{(i)}, p_1^{(i)}$ for x using S_i and F_i for calibration.

end

Solve (MultIVAP) to obtain an optimal solution $V \subseteq \mathcal{Y}$.

return V.

of the calibration set. This follows because we fit K IVAPs and the complexity of the IVAP is dominated by the sorting step in the isotonic regressions, as discussed in Vovk et al. [30].

3.2 White-box attack

To enable a fair and thorough evaluation of the MultIVAP, we also design a custom white-box attack specifically for fooling this defense. Given that the MultIVAP is a multi-probabilistic predictor returning a set of labels rather than a single prediction, we must slightly modify the typical goal of causing a misclassification since that is ill-defined here. On the one hand, we do not want the correct label to be present in the resulting label set; on the other hand, we do not want the MultIVAP to output too many or too few labels either as this can also raise suspicion. We settle on the following optimization problem:[6]

$$\min_{\tilde{x} \in \mathcal{X}} \ \|x - \tilde{x}\|_\infty + \lambda \|F(\tilde{x}) - s\|_\infty. \tag{3}$$

Here, $F : \mathcal{X} \to \mathbb{R}^K$ is the scoring classifier, $\lambda \in \mathbb{R}$ is a parameter and $s \in \mathbb{R}^K$ is a target vector of scores. The scalar λ is optimized via binary search so that the magnitude of the perturbation $\|x - \tilde{x}\|_\infty$ is as small as possible. The score vector s is determined by searching the calibration set of the MultIVAP for all calibration samples (x', y') such that y' does not belong to the prediction region of x. Among these candidates, we randomly sample one element x' and take its calibration score vector as the target $s = F(x')$.

An adversarial example produced by (3) is accepted only if the perturbation stays within the budget (that is, $\|x - \tilde{x}\|_\infty \leq \eta$), the MultIVAP does not return

[6] We use the ℓ_∞ norm everywhere as this is recommended by Madry et al. [14].

an empty prediction region and the true label is not present in the prediction region returned by the MultIVAP. Solutions to (3) satisfying these properties are counted as "successes", and the *success rate* of the attack is the fraction of samples that were counted as such.

The intuition behind our white-box attack is the following. The K IVAPs used internally by the MultIVAP will output the same upper and lower bounds for all samples that are assigned the same scores by the classifier F. Therefore, if we can corrupt a sample x into a sample $\tilde{x}$ which shares the score vector of another sample x', then $\tilde{x}$ will be associated with the same probabilities as x' and the optimization problem (MultIVAP) will have the same solution sets for both. This will cause the MultIVAP to output the same prediction regions for $\tilde{x}$ and x'. If, furthermore, the distance between x and $\tilde{x}$ is small and if the true label of x is not present in the prediction region of x', then $\tilde{x}$ can be considered an adversarial example. This is what we aim to achieve in (3). In finding an appropriate sample x' to target, we try to minimize $\|F(x) - F(x')\|_\infty$ in order to "warm-start" the attack.

Note additionally that the objective (3) can be optimized using gradient descent without issue, as the scoring classifier F is a standard (fully differentiable) neural network. In particular, our white-box attack will not suffer from any gradient masking.

4 Experiments

We perform experiments on Fashion-MNIST, CIFAR-10, Asirra and SVHN data sets [31, 13, 9, 17]. For each data set, we normalize the pixel values to the interval $[0, 1]$. For the Asirra data set, we also resized all images to 64×64 pixels to facilitate processing by our pipeline as the images come in various irregular sizes. We then train a standard convolutional neural network and apply algorithm 1 to obtain a calibrated multi-probabilistic predictor. We used the Keras library with the TensorFlow backend for training the neural networks [6, 1]. Each network was optimized using the Adam optimizer [12] and trained for 10 epochs, except for Asirra which was trained for 50 epochs. For the IVAPs, we made use of an implementation by Toccaceli [28]. We report several metrics to assess the performance of the MultIVAP:

- The accuracy (ACC) of the original model for comparison. We give both the accuracy of the original model on the full test set as well as the accuracy of the model on the subset of predictions accepted by the MultIVAP. This is referred to as the "corrected accuracy" (CORR).
- The accuracy (ACC) of the MultIVAP on the samples with non-empty prediction regions. We consider a prediction to be correct if the true label is included in the prediction region.
- The *predictive efficiency* (EFF), which is the average number of labels the MultIVAP outputs across all samples. Ideally, this number should be very close to one.

– The significance level ε at which the results for the MultIVAP were obtained. This level was determined by tuning the threshold on a held-out validation set such that the predictive efficiency was as close to one as possible.
– The true rejection rate (TRR) along with the false rejection rate (FRR) and the overall rejection rate (REJ). Formally, these are computed as follows:

$$\mathrm{TRR} = \frac{\mathrm{TR}}{\mathrm{TR} + \mathrm{FA}}, \quad \mathrm{FRR} = \frac{\mathrm{FR}}{\mathrm{FR} + \mathrm{TA}}, \quad \mathrm{REJ} = \frac{\mathrm{TR} + \mathrm{FR}}{\mathrm{TR} + \mathrm{FR} + \mathrm{TA} + \mathrm{FA}}.$$

The quantities appearing in these equations are the number of true rejections (TR), false acceptances (FA), false rejections (FR) and true acceptances (TA). Samples are rejected if the MultIVAP returns an empty prediction region. A rejection is true if the underlying scoring classifier F would have returned an erroneous prediction and false otherwise.

It is of particular importance for us to study the accuracy, efficiency, TRR and FRR of the MultIVAP together and not simply the accuracy. This is because the MultIVAP is a *multi-probabilistic* predictor, meaning it yields a set of possible labels for each sample instead of a single point prediction like most models do. When considering the performance of such a method compared to a standard classifier that outputs an argmax over a set of estimated probabilities, we must ask ourselves the following questions:

1. How often does the MultIVAP output a set of labels containing the correct label? This is measured by the accuracy score.
2. How many labels does the MultIVAP output on average? This is the predictive efficiency.
3. When the MultIVAP rejects a prediction, what is the probability that this rejection was "justified" (in the sense that the underlying model used by the MultIVAP was wrong)? This is quantified by the true rejection rate and the false rejection rate.

Naturally, we want the accuracy at 100%, the predictive efficiency close to one, the TRR close to 100% and the FRR close to 0%. We can exercise some amount of control over these metrics by varying the significance level ε. In our experiments, we used a held-out validation set to tune ε in order to have a predictive efficiency as close to one as possible. Depending on the particular application, however, one can use many other methods to tune ε. For instance, in cybersecurity settings, we might be more concerned with minimizing the FRR than we are with maximizing predictive efficiency and so we might tune ε so that the FRR does not exceed some specified threshold.

To assess the robustness of the MultIVAP against adversarial perturbations, we consider two threat models:

– *Defense-oblivious.* Here, we evaluate the defense against non-adaptive transfer attacks. That is, the attacks have no knowledge of the defense; they are crafted against the baseline model and then simply transferred to the MultIVAP. This is the bare minimum of robustness that any defense should hope

to achieve. The attacks we tested against here are DeepFool [15], projected gradient descent (PGD; Madry et al. [14]), fast gradient sign method (FGSM; Goodfellow et al. [10]), single pixel [26] and local search [16]. All implementations were provided by the Foolbox library [22]. We chose this particular set of attacks because of their diversity: there are gradient-based attacks (DeepFool, PGD and FGSM), non-gradient-based attacks (Single pixel and local search), iterative (DeepFool, PGD, single pixel, local search) and single step attacks (FGSM).

– *Defense-aware.* In this setting, we use our own adaptive attack detailed in section 3 to specifically bypass the MultIVAP. This evaluation should be a worst-case scenario for our defense and provides an estimation of a lower bound on its actual robustness.

Finally, we address the question of computational efficiency of our method. It is difficult to precisely quantify the complexity of the MultIVAP as it requires the solution of a MILP, which is NP-complete in general [4]. As such, we perform timing experiments where we measure the average time per prediction for both the baseline model and the MultIVAP on the test sets of each task.

The code for all our experiments is available at `https://github.com/saeyslab/multivap`.

4.1 Results

Results on clean data are reported in table 1. We can see that the MultIVAP always noticeably improves the accuracy of the original model, at the cost of rejecting a small fraction of predictions (REJ). The predictive efficiency is very close to one on all tasks, with high TRRs and relatively low FRRs.

Results for adversarial transfer attacks are presented in table 2. For Fashion-MNIST, we chose $\eta = 0.3$; for all other data sets, we let $\eta = 0.03$ (all norms are ℓ_∞). We can see that the MultIVAP is significantly more accurate than the original models and that it generally has a high TRR on adversarial samples.

Metrics for the white-box attack are reported in table 3 for 100 iterations[7] and different values of η in ℓ_∞ norm. We used the same values of η for the white-box attack as we did for the transfer attacks in order to make a fair comparison. Although the white-box adversarial samples appear very damaging for the MultIVAP, our attack does not succeed in producing many of them; the white-box attack therefore has only very limited success.

Figure 2 shows the results of our timing experiments where we compute the average time per prediction for the baselines as well as the MultIVAPs. Our findings indicate that the overhead incurred by our algorithm is proportional to the number of classes rather than any specific property of the data set or the underlying model.

[7] We found that increasing the number of iterations beyond 100 had no significant effect on the results, even up to 1,000 iterations with 10 random restarts per sample.

	Baseline	MultIVAP					
Task	ACC (CORR)	ACC	ε	EFF	TRR	FRR	REJ
Fashion-MNIST	91.81% (93.84%)	94.22%	24.20%	1.04	29.62%	4.45%	6.51%
CIFAR-10	76.40% (81.52%)	83.36%	20.77%	1.05	32.73%	8.36%	14.11%
Asirra	88.82% (89.04%)	88.56%	41.86%	1.00	2.69%	0.48%	0.73%
SVHN	92.22% (96.40%)	96.81%	25.23%	1.01	59.22%	7.70%	11.71%

Table 1: Results of the baseline models and the MultIVAPs on the different data sets.

		Baseline	MultIVAP				
Task	Attack	ACC (CORR)	ACC	EFF	TRR	FRR	REJ
Fashion-MNIST ($\eta = 0.3$)	DeepFool	27.98% (28.14%)	71.36%	0.66	43.60%	39.90%	42.56%
	PGD	0.00% (0.00%)	74.23%	0.69	41.59%	N/A	41.59%
	FGSM	1.10% (1.11%)	76.26%	0.53	56.76%	2.38%	56.19%
	Single pixel	85.05% (88.25%)	94.92%	1.02	35.28%	3.45%	8.21%
	LocalSearch	51.10% (52.46%)	83.10%	0.76	60.46%	2.84%	31.01%
CIFAR-10 ($\eta = 0.03$)	DeepFool	0.01% (0.01%)	63.52%	0.95	24.16%	16.10%	24.10%
	PGD	0.00% (0.00%)	63.37%	0.91	28.13%	N/A	28.13%
	FGSM	0.00% (0.00%)	59.26%	0.83	32.36%	0.00%	32.25%
	Single pixel	60.68% (68.12%)	84.82%	1.01	37.53%	4.39%	17.43%
	LocalSearch	18.99% (21.39%)	59.91%	0.73	48.62%	4.10%	40.16%
Asirra ($\eta = 0.03$)	DeepFool	0.00% (0.00%)	47.01%	0.99	1.18%	12.50%	12.04%
	PGD	0.00% (0.00%)	46.76%	0.98	1.60%	N/A	1.60%
	FGSM	0.00% (0.00%)	46.90%	0.98	1.64%	0.00%	1.63%
	Single pixel	75.40% (75.50%)	83.29%	1.00	1.63%	0.00%	0.40%
	LocalSearch	57.80% (57.87%)	74.01%	1.00	1.49%	0.00%	0.63%
SVHN ($\eta = 0.03$)	DeepFool	0.01% (0.01%)	72.58%	0.82	36.00%	16.34%	35.86%
	PGD	0.00% (0.00%)	73.92%	0.80	41.33%	0.00%	41.33%
	FGSM	0.00% (0.00%)	68.80%	0.54	58.72%	0.00%	58.55%
	Single pixel	69.59% (77.38%)	92.32%	0.94	58.50%	1.99%	19.17%
	LocalSearch	22.91% (25.47%)	71.10%	0.51	71.61%	2.01%	55.67%

Table 2: Accuracy of the baseline models and the MultIVAPs on adversarial transfer attacks.

			Baseline	MultIVAP				
Task	η	Success	ACC	ACC	EFF	TRR	FRR	REJ
Fashion-MNIST	0.3	18.96%	27.42%	0.00%	0.93	7.81%	5.53%	7.19%
CIFAR-10	0.03	27.55%	10.10%	0.00%	1.00	2.93%	2.70%	2.90%
Asirra	0.03	47.57%	24.14%	2.79%	1.00	0.00%	0.00%	0.00%
SVHN	0.03	9.85%	14.29%	0.00%	0.98	3.87%	6.83%	4.29%

Table 3: Results of the baseline models and the MultIVAPs on the adversarial white-box attack with 100 iterations of optimization.

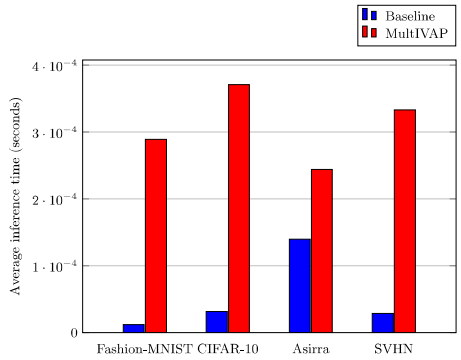

Fig. 2: Comparison of the computational efficiency of the baseline models and the MultIVAPs.

5 Conclusion

We have proposed a computationally efficient multi-class generalization of the inductive Venn-ABERS prediction algorithm which we have called the *MultIVAP*. In our experiments, we have found that this method significantly increases both the accuracy as well as the adversarial robustness of any model with which it is instantiated. The MultIVAP takes a single hyperparameter, the significance level ε, which can be tuned according to various objectives. The method enjoys the theoretical guarantee that its label sets will contain the true label with a probability of at least ε. The code for our implementation is available at `https://github.com/saeyslab/multivap`.

Several avenues for future work are possible. Obtaining tighter probabilistic bounds with stronger guarantees on the label set is perhaps the most interesting. For this one might explore different constraints or even a different cost function (such as a log-likelihood). However, the resulting optimization problem might no longer be linear or even convex. Improving the computational efficiency of this method when the number of classes becomes very high is definitely a worthwhile avenue for future work as well. Currently, the MultIVAP appears to increase inference time by a factor that is approximately equal to the number of classes. In applications that are very time-sensitive and which have a large number of classes, this might be unacceptable. It may also be possible to develop stronger white-box attacks against the MultIVAP. We therefore invite the community to scrutinize our defense and develop stronger attacks against it. Nevertheless, even if our particular defense is ever broken, we believe that methods inspired by the field of conformal prediction have much unexplored potential in this area.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. "TensorFlow: A system for large-scale machine learning". In: *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. 2016, pp. 265–283.

[2] Anish Athalye, Nicholas Carlini, and David Wagner. "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples". In: *arXiv preprint arXiv:1802.00420* (2018).

[3] Battista Biggio and Fabio Roli. "Wild patterns: Ten years after the rise of adversarial machine learning". In: *Pattern Recognition* 84 (2018), pp. 317–331.

[4] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[5] Dallas Card, Michael Zhang, and Noah A. Smith. "Deep Weighted Averaging Classifiers". In: *Proceedings of the Conference on Fairness, Accountability, and Transparency*. FAT* '19. Atlanta, GA, USA: ACM, 2019, pp. 369–378. ISBN: 978-1-4503-6125-5. DOI: 10.1145/3287560.3287595.

[6] François Chollet et al. *Keras*. 2015. URL: `https://keras.io` (visited on 07/17/2019).

[7] Harm De Vries, Roland Memisevic, and Aaron C Courville. "Deep Learning Vector Quantization." In: *ESANN*. 2016.

[8] Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. "Hotflip: White-box adversarial examples for text classification". In: *arXiv preprint arXiv:1712.06751* (2017).

[9] Jeremy Elson, John (JD) Douceur, Jon Howell, and Jared Saul. "Asirra: A CAPTCHA that Exploits Interest-Aligned Manual Image Categorization". In: *Proceedings of 14th ACM Conference on Computer and Communications Security (CCS)*. Association for Computing Machinery, Inc., Oct. 2007.

[10] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples". In: *arXiv preprint arXiv:1412.6572* (2014).

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[12] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[13] Alex Krizhevsky and Geoffrey Hinton. *Learning multiple layers of features from tiny images*. Tech. rep. Citeseer, 2009.

[14] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. "Towards deep learning models resistant to adversarial attacks". In: *arXiv preprint arXiv:1706.06083* (2017).

[15] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. "Deepfool: a simple and accurate method to fool deep neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2574–2582.

[16] Nina Narodytska and Shiva Prasad Kasiviswanathan. "Simple black-box adversarial perturbations for deep networks". In: *arXiv preprint arXiv:1612.06299* (2016).

[17] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. "Reading digits in natural images with unsupervised feature learning". In: *NIPS Workshop on Deep Learning and Unsupervised Feature Learning* (2011).

[18] Nicolas Papernot and Patrick McDaniel. "Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning". In: *arXiv preprint arXiv:1803.04765* (2018).

[19] Jonathan Peck, Bart Goossens, and Yvan Saeys. "Detecting adversarial examples with inductive Venn-ABERS predictors". In: *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*. 2019, pp. 143–148.

[20] Yao Qin, Nicholas Carlini, Garrison Cottrell, Ian Goodfellow, and Colin Raffel. "Imperceptible, Robust, and Targeted Adversarial Examples for

Automatic Speech Recognition". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, June 2019, pp. 5231–5240.

[21] Aditi Raghunathan, Jacob Steinhardt, and Percy S Liang. "Semidefinite relaxations for certifying robustness to adversarial examples". In: *Advances in Neural Information Processing Systems*. 2018, pp. 10877–10887.

[22] Jonas Rauber, Wieland Brendel, and Matthias Bethge. "Foolbox v0. 8.0: A Python toolbox to benchmark the robustness of machine learning models". In: *arXiv preprint arXiv:1707.04131* 5 (2017).

[23] Jonathan Shen, Patrick Nguyen, Yonghui Wu, Zhifeng Chen, Mia X Chen, Ye Jia, Anjuli Kannan, Tara Sainath, Yuan Cao, Chung-Cheng Chiu, et al. "Lingvo: a modular and scalable framework for sequence-to-sequence modeling". In: *arXiv preprint arXiv:1902.08295* (2019).

[24] Chawin Sitawarin and David Wagner. "On the Robustness of Deep K-Nearest Neighbors". In: *arXiv preprint arXiv:1903.08333* (2019).

[25] David So, Quoc Le, and Chen Liang. "The Evolved Transformer". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, June 2019, pp. 5877–5886.

[26] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. "One pixel attack for fooling deep neural networks". In: *IEEE Transactions on Evolutionary Computation* (2019).

[27] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. "Intriguing properties of neural networks". In: *arXiv preprint arXiv:1312.6199* (2013).

[28] Paolo Toccaceli. *Venn-ABERS predictor*. 2017. URL: `https://github.com/ptocca/VennABERS` (visited on 07/17/2019).

[29] Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*. Springer Science & Business Media, 2005.

[30] Vladimir Vovk, Ivan Petej, and Valentina Fedorova. "Large-scale probabilistic predictors with and without guarantees of validity". In: *Advances in Neural Information Processing Systems*. 2015, pp. 892–900.

[31] Han Xiao, Kashif Rasul, and Roland Vollgraf. *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. 2017. arXiv: `cs.LG/1708.07747 [cs.LG]`. (Visited on 07/17/2019).

[32] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. "Understanding deep learning requires rethinking generalization". In: *arXiv preprint arXiv:1611.03530* (2016).