# Probability-Based Classification for Adversarial Image Recognition with a Maximum Entropy-SVM Pipeline

BRIAN H. CHIANG, New York University, USA
YUELONG LI, New York University, USA
OLGA VROUSGOS, New York University, USA

For white-box perturbation attacks of size $||\epsilon||_\infty = 8/255$ that are directed against image classification schemes, we describe a novel algorithm using a maxent-SVM pipeline, provide reasoned conjectures about theoretical bounds on its performance and present preliminary data on the model's accuracy.

## 1 INTRODUCTION

The techniques of machine learning have been quite successfully applied to the problem of classifying images by the objects they contain. However, they sometimes fail to classify correctly on images modified from the original dataset whose changes are imperceptible to humans. In what follows, we formulate this problem in more mathematically precise language.

Suppose we have an input feature vector space $\mathcal{X}$, an output space $\mathcal{Y}$, a target hypothesis $h : \mathcal{X} \longrightarrow \mathcal{Y}$ and a machine $\mathcal{M} : \mathcal{X} \longrightarrow \mathcal{Y}$ that attempts to classify its inputs accurately. Let an image $A$ have feature vector $\boldsymbol{x} \in \mathcal{X}$ and be properly classified, i.e. $h(\boldsymbol{x}) = \mathcal{M}(\boldsymbol{x})$. Let $A'$ be an imperceptibly perturbed, misclassified image with feature vector $\boldsymbol{x} + \boldsymbol{\epsilon}$ for some perturbation vector $\boldsymbol{\epsilon} \in \mathcal{X}, \boldsymbol{\epsilon} > 0$. The misclassification of $A'$ arises from the fact that $h(\boldsymbol{x} + \boldsymbol{\epsilon}) = h(\boldsymbol{x})$, but $\mathcal{M}(\boldsymbol{x} + \boldsymbol{\epsilon}) \neq \mathcal{M}(\boldsymbol{x})$. Our goal, therefore, is to train a machine $\mathcal{M}'$ such that $\mathcal{M}'(\boldsymbol{x} + \boldsymbol{\epsilon}) \neq \mathcal{M}'(\boldsymbol{x})$ on as many feature vectors $\boldsymbol{x}$ and perturbation vectors $\boldsymbol{\epsilon}$ as possible.

For white-box attacks of this form where the size of the perturbation is limited by the constraint $||\boldsymbol{\epsilon}||_\infty \leq 8/255$ and the images are taken from the dataset CIFAR-10, we propose a candidate solution, present heuristic arguments for its correctness and disclose preliminary experimental results for our approach.

## 2 THEORY

### 2.1 Robust Error

Put simply, the goal of the problem is to minimize error under adversarial attacks. For a given model $f : \mathbb{R}^d \longrightarrow \mathbb{R}^k$, an unknown data distribution $\mathcal{D}$ and a perturbation $\epsilon > 0$, the adversarial attack is framed as maximising the loss:

$$\begin{gathered}\max_{\boldsymbol{z} \sim \mathcal{D}} L(f(\boldsymbol{z}), y) \\ \text{subject to} : ||\boldsymbol{z} - \boldsymbol{x}||_\infty < \epsilon\end{gathered}$$

for some unperturbed data point $\boldsymbol{x} \in \mathcal{X}$ with label $y \in \mathcal{Y}$ and surrogate loss function $L$ [Croce and Hein 2020]. If $L$ is the zero-one loss, this maximal value is upper-bounded by the robustness error, given by

$$R_{rob}(f) := \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{D}} \left[\mathbb{1}\left\{\exists Z \in \mathbb{B}(\boldsymbol{X}, \epsilon), f(Z)Y \leq 0\right\}\right] \geq \max_{Z\sim\mathcal{D}} \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{D}} \left[L(f(\boldsymbol{Z}), Y)\right],$$

where $\mathbb{B}(\boldsymbol{x}_0, \epsilon) = \{\boldsymbol{x} \in \mathcal{X} : ||\boldsymbol{x} - \boldsymbol{x}_0||_\infty < \epsilon\}$, i.e. the $\ell_\infty$ ball of radius $\epsilon$ centered at $\boldsymbol{x}_0$ [Zhang et al. 2019].

Authors' addresses: Brian H. Chiang, New York University, New York, New York, USA; Yuelong Li, New York University, New York, New York, USA; Olga Vrousgos, New York University, New York, New York, USA.

Note that the robustness error differs from the generalization error for unperturbed data, known as the natural generalisation error, which is defined by [Zhang et al. 2019] as

$$R_{nat}(f) := \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{D}}\left\{\mathbb{1}\left\{f(\boldsymbol{X})Y \le 0\right\}\right\}.$$

Following [Zhang et al. 2019], we also define $B_d(f)$, the neighborhood of the decision boundary of $f$, as

$$B_d(f) := \{\boldsymbol{x} \in \mathcal{X} : \exists \boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \epsilon), f(\boldsymbol{x})f(\boldsymbol{x}') \le 0\}.$$

Pictorially, this is the region that is within $l_\infty$ norm $\epsilon$ of the decision boundary $f(x) = 0$ (see Figure 1).

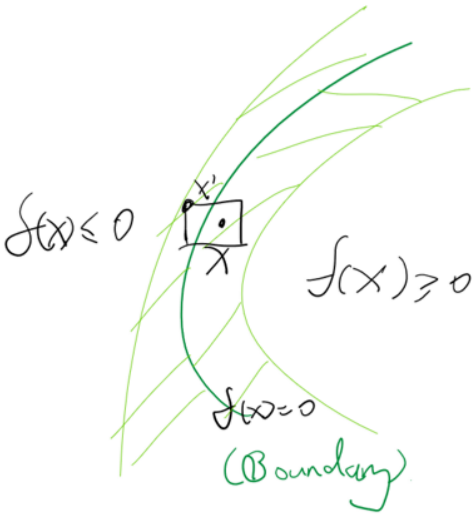

Fig. 1. The green line is the decision boundary put forth by the model, and the shaded light green region corresponds to $B_d(f)$. The diagram is a 2D Euclidean representation of the input space $\mathcal{X}$.

The region in the robust error can be separated into a disjoint union of the decision boundary region and the region defined in the natural generalisation error as follows:

$$\{\exists \boldsymbol{Z} \in \mathbb{B}(\boldsymbol{X}, \epsilon), f(\boldsymbol{Z})Y \le 0\} = \{f(\boldsymbol{X})Y \le 0\} \sqcup \left\{\mathbb{1}\left\{\boldsymbol{X} \in B_d(f), f(\boldsymbol{X})Y > 0\right\}\right\}.$$

From this, one can then dissect robust error into a natural component and a boundary component:

$$R_{rob}(f) = R_{nat}(f) + R_{bd}(f),$$

where $R_{bd}(f)$ is defined as

$$R_{bd}(f) = \mathbb{E}_{(\boldsymbol{X},Y)\sim D}\left\{\mathbb{1}\left\{\boldsymbol{X} \in B_d(f), f(\boldsymbol{X})Y > 0\right\}\right\}.$$

Note that both $R_{nat}$ and $R_{bd}$ are defined by unperturbed data points $\boldsymbol{X}$, and that by extension, $R_{rob}$ is as well.

Now we can upper-bound the robust error by

$$\begin{aligned} R_{rob}(f) &= \mathop{\mathbb{E}}_{X,Y\sim\mathcal{D}}\left[\mathbb{1}\left\{f(X)Y \le 0\right\}\right] + \mathop{\mathbb{E}}_{X,Y\sim\mathcal{D}}\left[\mathbb{1}\left\{X \in B_d(f), f(X)Y > 0\right\}\right] \\ &\le \mathop{\mathbb{E}}_{X,Y\sim\mathcal{D}}\left[\mathbb{1}\left\{f(X)Y \le 0\right\}\right] + \mathop{\mathbb{E}}_{X,Y\sim\mathcal{D}}\left[\mathbb{1}\left\{X \in B_d(f)\right\}\right] \\ &= R_{nat}(f) + \mathop{\mathbb{E}}_{X,Y\sim\mathcal{D}}\left[\mathbb{1}\left\{X \in B_d(f)\right\}\right] \\ &= R_{nat}(f) + \mathbb{P}\left[X \in B_d(f)\right]. \end{aligned}$$

### 2.2 Maxent-SVM Pipeline

The first part of our approach solves a maxent problem with a slightly modified cost function, as detailed in [Zhang et al. 2019]. For unperturbed data, the problem takes the form

$$\begin{aligned} &\min_{\mathrm{p}\in\Delta} \mathrm{D}(\mathrm{p}||\mathrm{p}_0) \\ \text{subject to : } &\left\| \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathrm{p}}\left[\boldsymbol{\Phi}(\boldsymbol{x})\right] - \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\hat{\mathcal{D}}}\left[\boldsymbol{\Phi}(\boldsymbol{x})\right] \right\|_\infty \le \lambda \end{aligned}$$

where $\boldsymbol{\Phi}$ is a general feature mapping that the maxent model uses internally.

We incorporate the perturbed, adversarial nature of the inputs by modifying the maxent optimisation problem as follows:

$$\begin{aligned} &\min_{\mathrm{p}\in\Delta} \mathrm{D}(\mathrm{p}||\mathrm{p}_0) \\ \text{subject to : } &\left\| \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathrm{p}\\ \boldsymbol{\sigma}}}\left[\boldsymbol{\Phi}(\boldsymbol{x}+\epsilon_\infty\boldsymbol{\sigma})\right] - \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\hat{\mathcal{D}}\\ \boldsymbol{\sigma}}}\left[\boldsymbol{\Phi}(\boldsymbol{x}+\epsilon_\infty\boldsymbol{\sigma})\right] \right\|_\infty \le \lambda \end{aligned}$$

for a randomised vector $\boldsymbol{\sigma} \in [-1,1]^n$ that perturbs the input image by $\epsilon_\infty = 8/255$.

As $\mathcal{Y}$ is a finite set of labels (or at least countable), let its elements be represented by consecutive elements of $\mathbb{Z}_+$, up to the integer $|\mathcal{Y}|$.

The output of our maxent model gives us the probabilities for each input point to be in each class. We formalise this output with the mapping $\boldsymbol{\phi} : \mathcal{X} \longrightarrow [0,1]^{|\mathcal{Y}|}$, where $\phi_j := \phi_j(\boldsymbol{x})$ is the $j$th component of $\boldsymbol{\phi} := \boldsymbol{\phi}(\boldsymbol{x})$ and satisfies

$$\phi_j := \phi_j(\boldsymbol{x}) = \mathbb{P}[h(\boldsymbol{x}) = j]$$

for target hypothesis $h$. We call $\boldsymbol{\phi}$ the probability vector for point $\boldsymbol{x}$.

### 2.3 Reduction of robust error via reducing natural and boundary errors

As an outline of the remaining proof sketches, we set a baseline for the robust error by considering the output model from *TRADES*, which we denote by $\boldsymbol{\phi}$:

$$R_{bd}(\boldsymbol{\phi}) \le R_{nat}(\boldsymbol{\phi}) + \mathbb{P}\left[X \in B_d(\boldsymbol{\phi})\right].$$

We hypothesise that the robust error bound on the maxent-SVM model is smaller than the robust error bound for the maxent-only model, i.e.

$$\begin{aligned} R_{rob}(g) &\leq R_{rob}(\boldsymbol{\phi}) \\ R_{nat}(g) + \mathbb{P}\left[\boldsymbol{X} \in B_d(g)\right] &\leq R_{nat}(\boldsymbol{\phi}) + \mathbb{P}\left[\boldsymbol{X} \in B_d(\boldsymbol{\phi})\right] \end{aligned}$$

for $g := \mathrm{SVM} \circ \boldsymbol{\phi}$ denoting the model provided by the maxent-SVM approach. We propose that this hypothesis can be achieved by two assumptions: that $R_{nat}(g) \leq R_{nat}(\boldsymbol{\phi})$, and that $\mathbb{P}\left[\boldsymbol{X} \in B_d(g)\right] \leq \mathbb{P}\left[\boldsymbol{X} \in B_d(\boldsymbol{\phi})\right]$.

### 2.4 Preservation of natural error

Without loss of generality, suppose that the $i$th sample point $\boldsymbol{x}^i \in S$ satisfies $h(\boldsymbol{x}^i) = j$. In the space $[0,1]^{|\mathcal{Y}|}$, for $\boldsymbol{\phi}$ that maps correctly, we have that $\mathrm{argmax}_k \phi^i_k = h(\boldsymbol{x}^i) = j$, which implies that $\phi^i_j \geq \phi^i_k$ for all $k \in \mathcal{Y}, k \neq j$.

From this, we see that any point $\boldsymbol{x}^i$ with a large value for $\phi^i_j$ either belongs to class $j$ or has some other component of the probability vector $\phi^i_k, k \neq j$ such that $\phi^i_k \geq \phi^i_j$. Geometrically, this suggests that for each class $j$, the set $\{\boldsymbol{\phi}(\boldsymbol{x}) : h(\boldsymbol{x}) = j\}$ can be separated from any set $\{\boldsymbol{\phi}(\boldsymbol{x}) : h(\boldsymbol{x}) = k, j \neq j\}$ by a hyperplane.

As an example of the existence of such separating hyperplanes, we formulate $|\mathcal{Y}|$ threshold functions, one for each dimension, to determine the predicted class for each point. We see that a threshold function in dimension $j$ is simply a hyperplane with normal vector parallel to the $j$th dimensional axis that classifies points as either being in class $j$ or not. Then based on our assumptions, SVM can achieve zero empirical loss on the unperturbed data if the inputs are linearly separable, and can closely track the empirical loss of $\boldsymbol{\phi}$ if $\boldsymbol{\phi}(S)$ is not separable. Hence, we hypothesize that

$$R_{nat}(g) = \mathop{\mathbb{E}}_{X,Y\sim\mathcal{D}}\left[\mathbb{1}\left\{g(X)Y \leq 0\right\}\right] \leq \mathop{\mathbb{E}}_{X,Y\sim\mathcal{D}}\left[\mathbb{1}\left\{\boldsymbol{\phi}(X)Y \leq 0\right\}\right] = R_{nat}(\boldsymbol{\phi}),$$

i.e. that error on unperturbed data points is not greater for $g$ than for $\boldsymbol{\phi}$.

### 2.5 Maxent-SVM and robust error

For the second term in the robust error, we hypothesise that $\mathbb{P}\left[\boldsymbol{X} \in B_d(g)\right] \leq \mathbb{P}\left[\boldsymbol{X} \in B_d(\boldsymbol{\phi})\right]$, which we claim with a heuristic argument. By our maxent-SVM model, we train our SVM on the output space of the maxent neural network, $[0,1]^{|\mathcal{Y}|}$, where the $j$th dimension of $[0,1]^{|\mathcal{Y}|}$ corresponds to the probability of being in class $j$.

Without loss of generality, we visualise a naïve decision boundary in $[0,1]^{|\mathcal{Y}|}$ between class 1 and class 2, expressed as the line $\phi_1 = \phi_2$ (see Figure 2). We observe that when we map $B_d(\boldsymbol{\phi})$ into this space, $\boldsymbol{\phi}(B_d(\boldsymbol{\phi}))$ may intersect with the clusters of data corresponding to the labels $y = 1$ or $y = 2$,.

Applying an SVM serves to maximise the decision margins of a separating hyperplane in its input space [Mohri et al. 2018], i.e. $[0,1]^{|\mathcal{Y}|}$. Thus, we see that in doing so, we increase the distance of the clusters from the decision boundary and potentially exclude points in clusters from the boundary region $\boldsymbol{\phi}(B_d(\mathrm{SVM} \circ \boldsymbol{\phi})) = \boldsymbol{\phi}(B_d(g))$ (see Figure 3). This leads us to conjecture that

$$\mathbb{P}[\boldsymbol{X} \in B_d(g)] = \mathbb{P}[\boldsymbol{\phi}(\boldsymbol{X}) \in \boldsymbol{\phi}(B_d(g))] \leq \mathbb{P}[\boldsymbol{\phi}(\boldsymbol{X}) \in \boldsymbol{\phi}(B_d(\boldsymbol{\phi}))] = \mathbb{P}[\boldsymbol{X} \in B_d(\boldsymbol{\phi})].$$

From this and the prior assumption that $R_{nat}(g) \leq R_{nat}(\boldsymbol{\phi})$, we have that

$$R_{rob}(g) = R_{nat}(g) + \mathbb{P}\left[\boldsymbol{X} \in B_d(g)\right] \leq R_{nat}(\boldsymbol{\phi}) + \mathbb{P}\left[\boldsymbol{X} \in B_d(\boldsymbol{\phi})\right] = R_{rob}(\boldsymbol{\phi}).$$

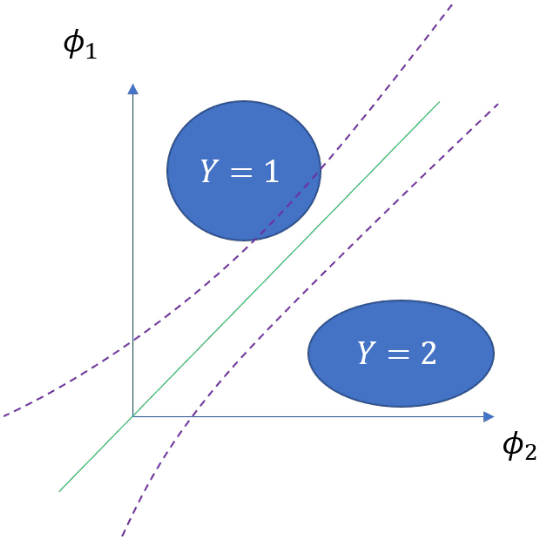

Fig. 2. Ellipses represent clusters of unperturbed training data, with class labels as shown. The green line is the decision boundary, and the region between the dashed purple lines is $\boldsymbol{\phi}(B_d(\boldsymbol{\phi}))$.

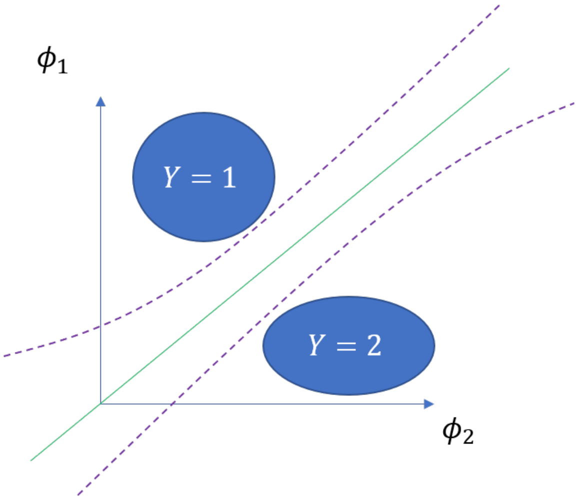

Fig. 3. Ellipses represent clusters of unperturbed training data, with class labels as shown. The green line is the decision boundary, and the region between the dashed purple lines is $\boldsymbol{\phi}(B_d(g))$.

## 3 APPROACH

Given a labelled, unperturbed sample $S$ and a set of labels $\mathcal{Y}$, we do the following to train our machine:

- Generate the labelled sample $S'$ of perturbed data. Define $\mathcal{S} := S \cup S'$.
- Train a maxent model for classification.

- As in the previous section, let the output of the maxent model be represented by a mapping $\boldsymbol{\phi} : \mathcal{X} \to [0, 1]^{|\mathcal{Y}|}$. For each sample point $\boldsymbol{x}^i \in \mathcal{S}$, the output of the maxent model is a probability vector $\boldsymbol{\phi}^i := \boldsymbol{\phi}(\boldsymbol{x}^i)$, where $\phi^i_j$, the $j$th component of $\boldsymbol{\phi}^i$, is the probability that $\boldsymbol{x}^i$ belongs to class $j$.
- For each class $j$, train a support vector machine $\mathrm{SVM}_j$ that determines the boundary between the half-space for points that are in class $j$ and the half-space for points that are not in class $j$. We will say that $\mathrm{SVM}_j$ recognises a point $\boldsymbol{x}^i$ exactly if it classifies $\boldsymbol{x}^i$ as belonging to class $j$. Additionally, we define $\rho^i_j$, the confidence of $\mathrm{SVM}_j$ on point $\boldsymbol{x}^i$, by the $\ell_2$ distance from $\boldsymbol{x}^i$ from to the hyperplane produced by $\mathrm{SVM}_j$.
  - Suppose that a point $\boldsymbol{x}^i$ is recognised by exactly one support vector machine. Without loss of generality, let this support vector machine be $\mathrm{SVM}_j$. Then for all such points $\boldsymbol{x}^i$, we produce the hypothesis that the $\boldsymbol{x}^i$ belongs to class $j$.
  - Suppose that a point $\boldsymbol{x}^i$ is recognised by multiple support vector machines. Without loss of generality, suppose that the support vector machine with maximal confidence on $\boldsymbol{x}^i$ is $\mathrm{SVM}_j$. Then for all such points $\boldsymbol{x}^i$, we produce the hypothesis that $\boldsymbol{x}^i$ belongs to class $j$.
  - Suppose that a point $\boldsymbol{x}^i$ is not recognised by any support vector machines. Without loss of generality, suppose that the support vector machine with minimal confidence on $\boldsymbol{x}^i$ is $\mathrm{SVM}_j$. Then for all such points $\boldsymbol{x}^i$, we produce the hypothesis that $\boldsymbol{x}^i$ belongs to class $j$.

## 4 EXPERIMENTAL RESULTS

We share our code at https://github.com/YuelongLi/FML-Final-Project. We made use of the AutoAttack perturbation generator [Croce and Hein 2020], the TRADES maxent neural network [Zhang et al. 2019] and the libSVM functionality provided by [Chang and Lin 2011].

With the AutoAttack perturbation generator, we generated a labelled sample of adversarial data using projected gradient descent [Croce and Hein 2020]. In the interest of time, we decided to generate 2000 adversarial data points for training and 1000 adversarial data points for testing.

We then used the adversarial data as input to the pre-trained TRADES neural network that is available for download [Zhang et al. 2019]. The TRADES neural network uses relative entropy (Kullback-Leibler divergence) as its loss function [Zhang et al. 2019], and is thus a maxent model for classification.

We took the raw signal (logit) output of the neural network, assuming that the error on the produced data would be small. The raw predictions from the neural network were then transformed into a normalised dataset that libSVM could consume. Since we wanted to train a SVM for each CIFAR-10 class, we created 10 different datasets. For each dataset, a unique class was chosen to be given the positive (+1) label, with all other classes assigned the negative (-1) label, yielding a series of datasets suitable for binary classification.

For each class, we trained an SVM using libSVM after applying some basic hyper parameter tuning, which included the use of polynomial kernels of degree 2, and some experimentation with higher-degree polynomial kernels.

To evaluate our experiments, we compared the classification accuracy of each class between the neural net-only model and our model, which applies the corresponding SVM to the result of the neural network.

The first round of results showed an increase in accuracy that struck us as being unusually high (see Figure 4).

On closer inspection of the generated datasets for each class, we realised that the number of instances that were positively labelled was incredibly low. Moreover, we found that each SVM ended up learning to predict the negative label on every input, which resulted in high accuracy without actually improving the predictive power beyond that of the neural network alone.

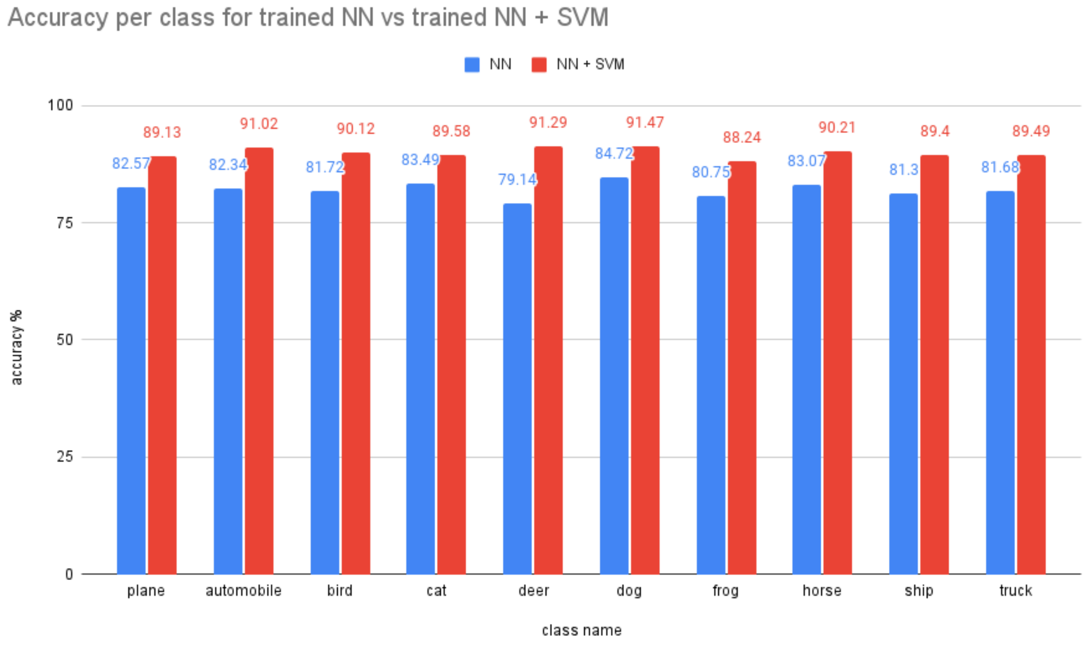

Fig. 4. Accuracy comparison for each class between the trained TRADES neural network and the trained TRADES neural network + corresponding SVM.

To tackle this we decided to make sure that the SVM datasets contained a roughly equal number of positive and negative labels. We did this by randomly rejecting negatively labelled data points when there were more points with negative labels than points with positive labels in the dataset. The resulting accuracy for each class can be seen in Figure 5.

We see that for most classes, applying the SVM increases the overall accuracy of recognising class. However, for some classes, the accuracy decreases. Those classes could potentially benefit from more specific hyper parameter tuning, or by using more data to train the SVMs.

## 5 FUTURE DIRECTIONS

Potential avenues of future research could include a rigorous proof that the SVM layer reduces robust error as theoretical justification for the empirical performance; or the use of non-polynomial kernels to further enhance the separation power of SVMs on the probability space $[0, 1]^{|\mathcal{Y}|}$.

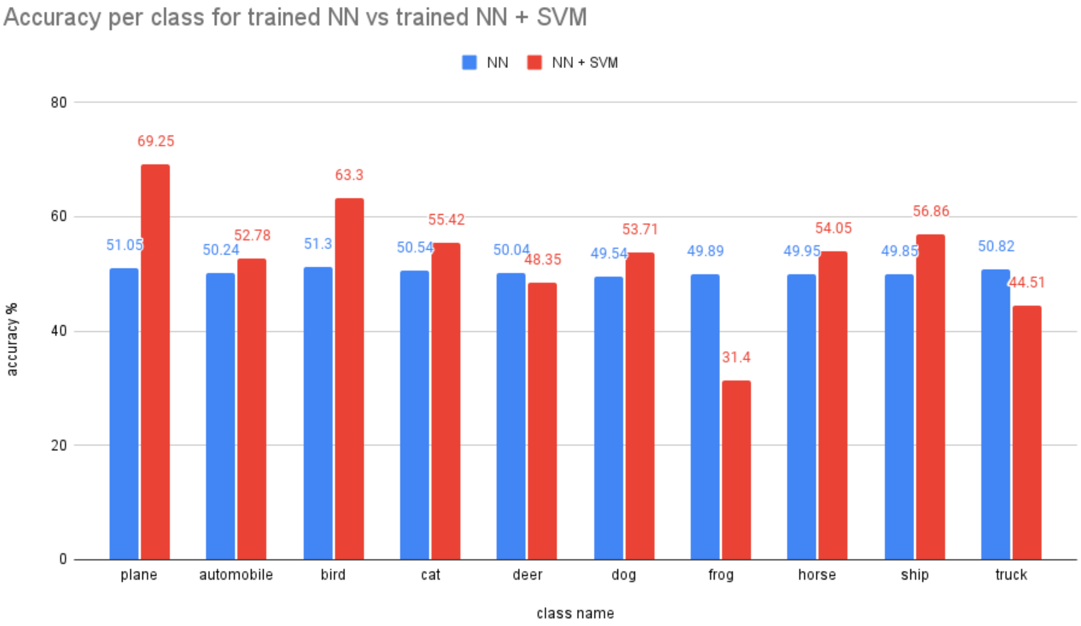

Fig. 5. Accuracy comparison for each class between the trained TRADES neural network and the trained TRADES neural network + corresponding SVM. The datasets chosen here have a roughly equal number of positive and negative labels.

## REFERENCES

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A Library for Support Vector Machines. *ACM Trans. Intell. Syst. Technol.* 2, 3, Article 27 (may 2011), 27 pages. https://doi.org/10.1145/1961189.1961199

Francesco Croce and Matthias Hein. 2020. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. *CoRR* abs/2003.01690 (2020). arXiv:2003.01690 https://arxiv.org/abs/2003.01690

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. 2018. *Foundations of Machine Learning* (2nd ed.). The MIT Press.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. 2019. Theoretically Principled Trade-off between Robustness and Accuracy. https://doi.org/10.48550/ARXIV.1901.08573