EFFICIENT ADVERSARIAL TRAINING WITH DATA PRUNING

Max Kaufmann
University of Cambridge

Jamie Stephenson
Independent

Yiren Zhao
University of Cambridge

Ilia Shumailov
University of Cambridge and Vector Institute

Robert Mullins
University of Cambridge

Nicolas Papernot
University of Toronto and Vector Institute

ABSTRACT

Neural networks are susceptible to adversarial examples — small input perturbations that cause models to fail. Adversarial training is one of the solutions that stops adversarial examples; models are exposed to attacks during training and learn to be resilient to them. Yet, such a procedure is currently expensive – it takes a long time to produce and train models with adversarial samples, and, what is worse, it occasionally fails. In this paper we demonstrate *Data Pruning* — a method for increasing adversarial training efficiency through data sub-sampling. We empirically show that *Data Pruning* leads to improvements in convergence and reliability of adversarial training, albeit with different levels of utility degradation. For example, we observe that using random sub-sampling of CIFAR10 to drop 40% of data, we lose 2% adversarial accuracy against the strongest attackers and reduce runtime by 35%. Interestingly, we discover that in some settings data pruning brings benefits from both worlds — it both improves adversarial accuracy and training time.

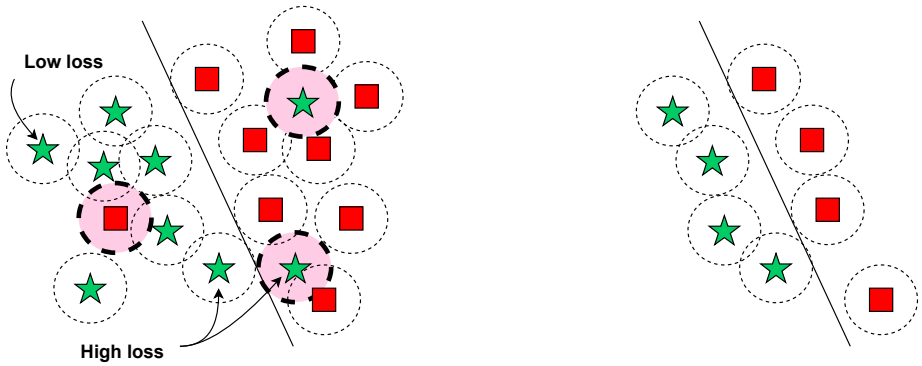
This is a work in progress, for a second version of this paper.

1 INTRODUCTION

The existence of *adversarial examples* (Goodfellow et al., 2015a) poses a real threat to Machine Learning (ML) systems. These are an interesting phenomenon in modern neural networks, wherein small (semantic preserving) perturbations to network inputs lead to incorrect behaviour. Although first found in the image classification context (Szegedy et al., 2013; Biggio et al., 2013), they have since been discovered in a variety of domains—including natural language processing (Samanta and Mehta, 2017; Boucher et al., 2021), audio processing (Carlini and Wagner, 2018; Ahmed et al., 2022) and deep reinforcement learning (Gleave et al., 2021; Zhao et al., 2020).

A common technique to make models less susceptible to such attacks is *adversarial training* (Goodfellow et al., 2015b), wherein correctly labelled adversarial examples are shown at training time, allowing the models to learn correct behaviour even in the face of malicious inputs – such models are said to be *adversarially robust*. In the image classification context, training of this form has been successful in decreasing the susceptibility of models to adversarial examples (Goodfellow et al., 2015a; Madry et al., 2019; locuslab, 2020; Carlini et al., 2019). However, traditional adversarial training regimes have often been prohibitively expensive, with adversarial training being up to thirty times more computationally expensive than training a non-robust model (Shafahi et al., 2019). Historically, this meant that training robust models on large datasets has been exclusive to research groups with access to hundreds of GPUs (Xie et al., 2019; Kannan et al., 2018). Furthermore, the high computational cost adds extra barriers to these systems being deployed in practice—this is relevant when viewing adversarial robustness as an important safety property for modern systems (Hendrycks et al., 2021),

In this paper we seek to explore and expand on these methods—with the aim of continuing to reduce the barrier to training robust models. To this end, we highlight *data pruning* techniques as a particularly attractive direction for reducing the training time of adversarial examples. We find that sub-sampling the



(a) Data close to the decision boundary causes issues during learning.

(b) Pruned data leads to less overlap in the decision boundary.

Figure 1: **Improving training by dropping points close to the decision boundary.** Our low-distance data pruning techniques are aimed at dropping out datapoints which might lead to instabilities during training. We visualise how (a) data before dropout compares (b) data after dropping out our finetuning process. We can see that points close to the decision boundary are likely to cause issues when trying to learn a classifier. A similar dynamic and be seen in the differences between Figures 2a and 2b.

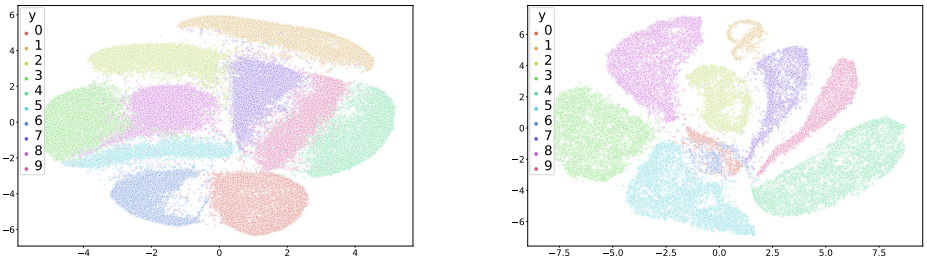
training dataset, we observe significant reduction in adversarial training time, while balancing standard and adversarial performance. We focus on distance-based pruning techniques for reducing the dataset size, showing that it is possible to favour adversarial performance—exploring how this relatively simple idea leads to more complex regimes for adversarial training. We also demonstrate relatively unexpected results—in cases where lack of data is not a constraint to model performance, we observe that dropping more data leads to increased adversarial robustness. Our work suggests that in future a more advanced sub-sampling strategy may help improve adversarial robustness, while providing significant speed ups from selecting information-efficient data subsets.

Overall we make the following contributions:

- We introduce the potential for data pruning as a unexplored frontier for reducing the cost of adversarial training.
- We design a class of *distance-based data pruning techniques* for pruning points in relation to their distance to the decision boundary.
- We evaluate how these techniques trade-off training time with adversarial accuracy, showing that on some datasets they can even *increase* the adversarial accuracy of models, while dropping out 40% of the data.
- We show that on larger datasets, random pruning is a strong baseline, showing promising tradeoffs in accuracy and computation time.

2 Related work

Adversarial robustness Adversarial machine learning in the context of modern deep networks was introduced in a seminal papers by *Szegedy et al.* (Szegedy et al., 2013) and *Biggio et al.* (Biggio et al., 2013), where it was noted that modern machine learning approaches are vulnerable when faced with small perturbations to their inputs. A follow up paper by *Goodfellow et al.* (Goodfellow et al., 2015a) then gave the first example of *adversarial training*, introducing a relatively simple *Fast Gradient Sign Method* (FGSM) for the generation of adversarial examples. This was the paper originally noting that exposing models to adversarial examples at training time increases their performance in the face of an attacker.



(a) Stadard Dataset

(b) Pruned Dataset

Figure 2: **Low-distance pruning leads to more cleanly delineated class boundaries.** We use T-distributed Stochastic Neighbour Embedding (Van der Maaten and Hinton, 2008) to visualise the effect of low-distance pruning (using loss as a distance proxy) on the MNIST dataset. As can be seen, the plots mirror fig. 1, we see a cleaner deliniation between the competing classes.

3 METHODOLOGY

In this section we introduce our class of *Data Pruning* techniques and explain the motivation behind them. Data pruning as a method for reducing the computational cost of adversarial training The computational cost of adversarial training is very dependent on the exact algorithms used, where performance can be factored as follows. In a single one of the E epochs $\mathcal{O}(N)$ gradient operations occur on each of the M data points. Since training performance is dominated by the number of passes required through the network (Wong et al., 2020), this factors the run-time to be well described by $\mathcal{O}(NME)$. This points towards three directions for improvements in computational-efficiency:

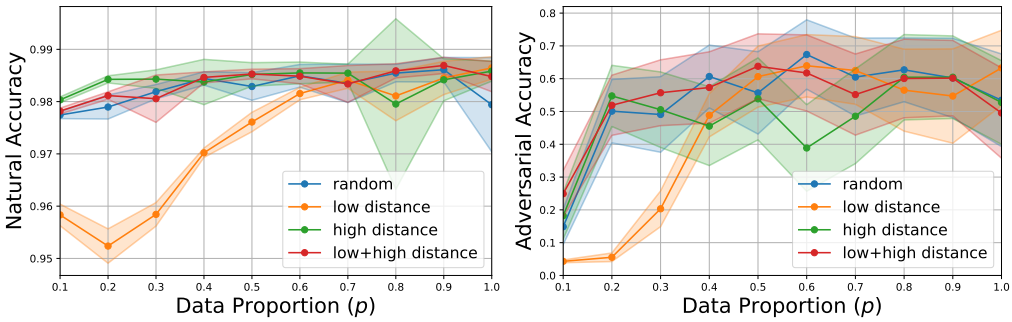
1. **Reduce N (Number of network passes per data point)**: As is done in the RS+FGSM algorithm, reducing the number of passes required to compute an adversarial example (while preserving its effectiveness) decreases computation time. Since our method is developed on top of RS+FGSM, we exploit this reduction.
2. **Reduce M (Number of data points)**: One can aim to reduce the amount of data required for convergence of the algorithm, by carefully selecting datapoints.
3. **Reduce E (Number of epochs)**: Faster convergence of the algorithm leads to reduced computation time.

When aiming to further reduce the computational costs of adversarial training, these factors can be separately considered. Due to the existing low-cost nature of the RS+FGSM (only one gradient pass), and the fact that faster convergence of neural networks is already heavily optimised in the modern machine learning pipeline, **we focus instead on increasing performance by training on less data**. This is the main motivation behind *Data Pruning* as a promising directoin for reducing the overhead of adversarial training.

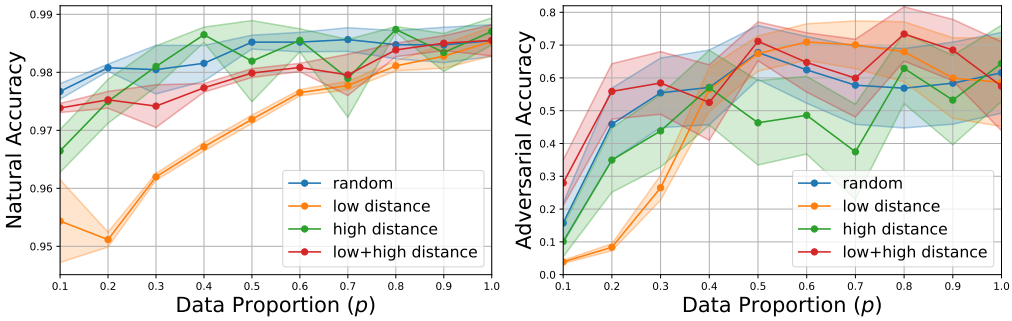
3.1 DECISION-BOUNDARY DISTANCE AS A METRIC FOR PRUNING DATA

In order to reduce the number of data-points while limiting any negative impact to the performance of the model, we must consider which data-points are least important for or even detrimental to a model's successful training. We explore and implement two competing intuitions from the literature:

High distance pruning Hua et al. (2021) claims that, during training, "boundary examples" that occur close to the decision boundary of a model are the most helpful in making the model more adversarially robust. This is because these hard examples hold the highest influence over the decision boundary, while other points are part of the core are examples which the model can already classify correctly even when adversarially perturbed (and could therefore be considered redundant (Birodkar et al., 2019)). This suggests that pruning data that is far from the model's decision boundary will limit any negative impact on the model's performance.



(a) *DeepFool*



(b) Loss

Figure 3: **Distance-based pruning on MNIST.** Mean natural data accuracy and mean AutoAttack accuracy of models trained on MNIST using distance-based and random pruning in which a proportion p of the data is kept after the pruning epoch $e = 1$. The *DeepFool* algorithm and loss are used for distance proxies. Error bars denote the 95% confidence interval using 30 models per datapoint. We use a standard LeNet-like network which achieves 99% standard accuracy when trained without adversarial methods.

Low distance pruning A competing intuition is that points close to the decision boundary, is that outlier examples are particularly harmful for adversarial robustness (Dong et al., 2021) introduces a measure of a data-point's quality and claims that training on low-quality data, which models are more uncertain around, causes a model to become less robust to adversarial perturbations. These outlier points (pictured in Section 1) can be pruned away, leading to a less marked effect on the existing decision boundaries.

In this work, we explore how to use these approximations of distance to prune adversarial training datasets.

3.2 Data pruning with distance

We design a simple algorithm for pruning data which is far away from the decision boundary, which is controlled by two hyperparameters (data proportion p, and pruning epoch e):

1. Select an approximation for distance from the original data-point.
2. Begin standard adversarial training on your dataset as normal, continue until you reach epoch e.
3. At epoch e, use your approximation to order all datapoints by their distance from the decision boundary.
4. Drop out either the bottom or top p percent of data, continue to train on the rest of the datapoints.

For a full description, see appendix D.

Within this work, we consider two main approximations to distance from the decision boundary:

DeepFool. We use the *DeepFool* algorithm, from (Moosavi-Dezfooli et al., 2016) to directly approximate the ℓ_2 decision boundary distance. We briefly describe the algorithm below:

Pruning Method	p	Natural data accuracy	AutoAttack Accuracy	Training Time (s)
Low distance (*DeepFool*)	0.6	97.83 ± 0.33	56.62 ± 5.13	50.39 ± 0.25
Low distance (Loss)	0.6	97.45 ± 0.07	64.17 ± 4.67	29.91 ± 0.11
Random	0.6	98.32 ± 0.16	56.66 ± 5.90	29.73 ± 0.10
None	1.0	98.46 ± 0.14	51.39 ± 6.61	39.09 ± 0.12

Table 1: Performance of RS+FGSM training on MNIST with different pruning strategies. The values in each column correspond to the 95% confidence interval using 120 models for each strategy. We note that (for p=0.6), loss-based pruning seems to beat both random pruning and training on the full dataset.

Let $\hat{k}(\mathbf{x}_0)$ denote class that the current model f predicts for a given datapoint $\mathbf{x}_0$. The current model defines the decision boundary, which splits input space into different regions depending on what class f predicts for each datapoint. We first find the region closest to $\mathbf{x}_0$ for which the associated class is not $\hat{k}(\mathbf{x}_0)$. We denote this class as $\hat{l}(\mathbf{x}_0)$ and we compute it using a linear approximation of the decision boundary about $\mathbf{x}_0$ as follows:

$$\hat{l}(\mathbf{x}_0) = \underset{k \neq \hat{k}(\mathbf{x}_0)}{\arg\min} \frac{\left| f_k(\mathbf{x}_0) - f_{\hat{k}(\mathbf{x}_0)}(\mathbf{x}_0) \right|}{\left\| \nabla f_k(\mathbf{x}_0) - \nabla f_{\hat{k}(\mathbf{x}_0)}(\mathbf{x}_0) \right\|_2}$$

Next we find the minimum perturbation $\boldsymbol{r}^*(\mathbf{x}_0)$ which is the vector that projects $\mathbf{x}_0$ on to the hyperplane given by the linear approximation of the decision boundary, i.e.,

$$\boldsymbol{r}^*(\mathbf{x}_0) = \frac{\left| f_{\hat{l}(\mathbf{x}_0)}(\mathbf{x}_0) - f_{\hat{k}(\mathbf{x}_0)}(\mathbf{x}_0) \right|}{\left\| \nabla f_{\hat{l}(\mathbf{x}_0)}(\mathbf{x}_0) - \nabla f_{\hat{k}(\mathbf{x}_0)}(\mathbf{x}_0) \right\|_2^2} \left(\nabla f_{\hat{l}(\mathbf{x}_0)}(\mathbf{x}_0) - \nabla f_{\hat{k}(\mathbf{x}_0)}(\mathbf{x}_0) \right). \tag{1}$$

In other words, we find the closest projection of $\mathbf{x}_0$ on faces of the polyhedron obtained from a linear approximation of the decision boundary of f. Finally we define $\mathbf{x}_{i+1} = \mathbf{x}_i + \boldsymbol{r}^*(\mathbf{x}_i)$ and iterate using Equation (1) (without recalculating $\hat{l}(\mathbf{x}_0)$) until we find the smallest n such that $\hat{k}(\mathbf{x}_0) \neq \hat{k}(\mathbf{x}_n)$. From this we have that the approximate decision boundary distance d is given by:

$$d = \left\| \sum_{i=0}^{n} \boldsymbol{r}^*(\mathbf{x}_i) \right\|_2$$

In practice this normally takes very few iterations as the linear approximation is fairly accurate and a small overshoot is added to $\boldsymbol{r}^*$.

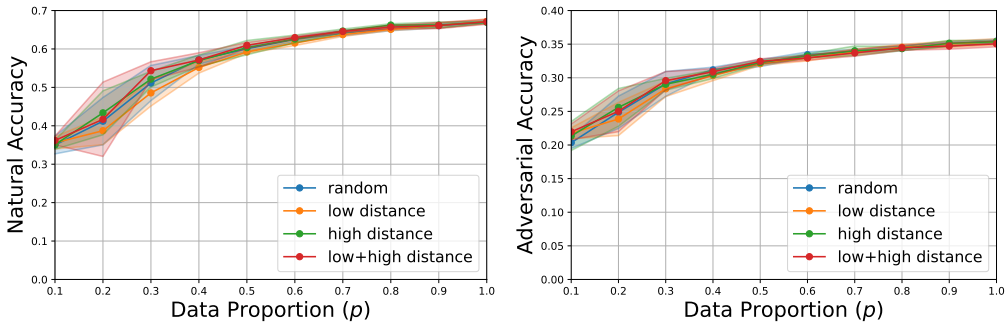
Loss as a cheap alternative. The *DeepFool* algorithm uses several network passes to estimate the distance to the decision boundary - making it unsuitable for a more computationally efficient training method. We use the loss on a specific datapoint as a cheap alternative for estimating points which work on the *DeepFool* algorithm in the next few passes. This is the main technique which we explore as a computationally efficient alternative for full data training.

4 Evaluation

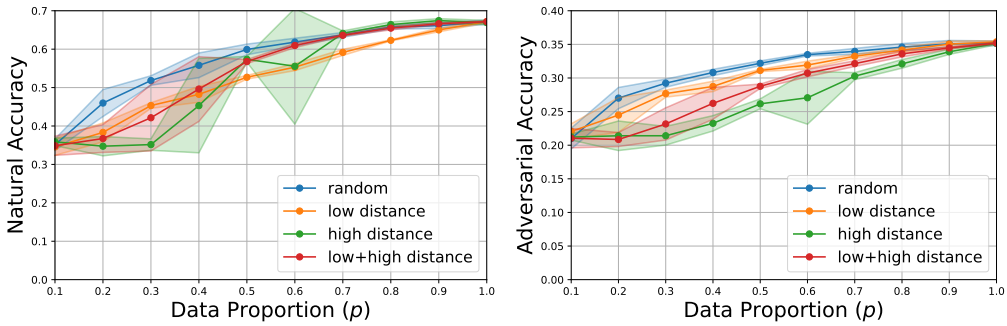
4.1 The effect of *Data Pruning* on model accuracy

To evaluate the performance of our *Data Pruning* techniques, we compare the effects of data pruning techniques on models on MNIST (LeCun et al., 2010) (Figure 3) and CIFAR10 (Krizhevsky et al., 2009) (Figure 4). We vary the amount of data removed during the pruning epoch and evaluated both the AutoAttack (Croce and Hein, 2020) and natural data accuracy after training was completed. From these results we draw the following conclusions:

High levels of *Data pruning* often lead to minimal drops in accuracy. For MNIST, in most cases both AutoAttack and natural data accuracy are maintained (or increased) as data proportion is decreased. For CIFAR10, random pruning and distance-based pruning using a *DeepFool* distance proxy results in only



(a) *DeepFool*



(b) Loss

Figure 4: **Accuracies for models trained on CIFAR10.** We use pruning epoch $e = 1$. Error bars denote the 95% confidence interval using 5 models per datapoint. We use a PreAct-ResNet18 (Ilyas et al., 2019) network which achieves 91% standard accuracy when trained without adversarial methods.

a small drop in accuracy even when up to 50% of the data is dropped. For example, on CIFAR-10 using random pruning with $p = 0.6$ only gives a 2% drop in AutoAttack accuracy and a 5% drop in natural data accuracy. These small sacrifices in accuracy lead to large time savings, as can be seen in **??**. This is exactly the effect we were seeking when designing our *Data Pruning* techniques..

On simpler datasets, *Data Pruning* can beat training on the full dataset. We carry out hypothesis tests table 1 on data proportions of $p = 0.6$ and $p = 1$ on low-distance pruning. We observe that in both cases *Data Pruning* increases the mean AutoAttack accuracy of a model (51.4% $\rightarrow$ 56.6% at $\alpha < 0.15$ (*DeepFool*) and 51.4% $\rightarrow$ 64.2% at $\alpha < 0.001$ (Loss)). More details can be found in Appendix B.

On simpler datasets, loss-based pruning beats random pruning. We found during MNIST experimentation that, while its variance is large, random pruning occasionally can give a model whose AutoAttack accuracy is above 90%. This suggests the presence of the "core set" mentioned is Section 3.1; a subset on which adversarial training is very effective. While random pruning would be producing such a set by chance, it seems that low distance pruning with a loss distance proxy produces a better performing subset with more consistency, shown by the lower standard deviation and higher mean AutoAttack accuracy in Figure 3.

Indeed, Table 1 also contains data from a large sample of models trained with random pruning from which we can see that distance pruning with a loss distance proxy does perform better than random pruning (56.6% $\rightarrow$ 64.2% at $\alpha < 0.025$).

5 Conclusion

In this paper we presented *Data Pruning* – a method to increase the speed of adversarial training. We show several data sub-sampling strategies for reducing the number of data points required for adversarial training. We observe an intriguing phenomenon that a loss-based sub-sampling strategy can not only reduce the

training time but also improve the adversarial robustness on MNIST, hopefully opening the door for future work in this area.

ACKNOWLEDGMENTS

We would like to acknowledge our sponsors, who support our research with financial and in-kind contributions: CIFAR through the Canada CIFAR AI Chair, DARPA through the GARD project, Intel, Meta, NFRF through an Exploration grant, and NSERC through the COHESA Strategic Alliance. Resources used in preparing this research were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute.

REFERENCES

Shimaa Ahmed, Yash Wani, Ali Shahin Shamsabadi, Mohammad Yaghini, Ilia Shumailov, Nicolas Papernot, and Kassem Fawaz. Pipe overflow: Smashing voice authentication for fun and profit, 2022.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013.

Vighnesh Birodkar, Hossein Mobahi, and Samy Bengio. Semantic redundancies in image-classification datasets: The 10% you don't need, 2019. URL https://arxiv.org/abs/1901.11409.

Nicholas Boucher, Ilia Shumailov, Ross Anderson, and Nicolas Papernot. Bad characters: Imperceptible nlp attacks, 2021.

Nicholas Carlini and David Wagner. Audio adversarial examples: Targeted attacks on speech-to-text, 2018.

Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian J. Goodfellow, Aleksander Madry, and Alexey Kurakin. On evaluating adversarial robustness. *CoRR*, abs/1902.06705, 2019. URL http://arxiv.org/abs/1902.06705.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks, 2020. URL https://arxiv.org/abs/2003.01690.

Chengyu Dong, Liyuan Liu, and Jingbo Shang. Data quality matters for adversarial training: An empirical study, 2021.

Rhett N. D'souza, Po-Yao Huang, and Fang-Cheng Yeh. Structural analysis and optimization of convolutional neural networks with a small sample size. *Nature*, 10(1), Jan 2020. doi: 10.1038/s41598-020-57866-2. URL https://www.nature.com/articles/s41598-020-57866-2#citeas.

Adam Gleave, Michael Dennis, Cody Wild, Neel Kant, Sergey Levine, and Stuart Russell. Adversarial policies: Attacking deep reinforcement learning, 2021.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *International Conference on Learning Representations (ICLR)*, 2015a.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015b.

Dan Hendrycks, Nicholas Carlini, John Schulman, and Jacob Steinhardt. Unsolved problems in ml safety, 2021.

Weizhe Hua, Yichi Zhang, Chuan Guo, Zhiru Zhang, and G. Edward Suh. Bullettrain: Accelerating robust neural network training via boundary example mining, 2021.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features, 2019.

Harini Kannan, Alexey Kurakin, and Ian Goodfellow. Adversarial logit pairing, 2018.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

locuslab. Github - locuslab/fast_adversarial: [iclr 2020] a repository for extremely fast adversarial training using fgsm, Jul 2020. URL https://github.com/locuslab/fast_adversarial.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2019.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks, 2016.

Suranjana Samanta and Sameep Mehta. Towards crafting text adversarial samples, 2017.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free!, 2019.

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

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy, 2018. URL https://arxiv.org/abs/1805.12152.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training, 2020.

Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan Yuille, and Kaiming He. Feature denoising for improving adversarial robustness, 2019.

Yiren Zhao, Ilia Shumailov, Han Cui, Xitong Gao, Robert Mullins, and Ross Anderson. Blackbox attacks on reinforcement learning agents using approximated temporal information. In *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, pages 16–24. IEEE, 2020.

A Hyperparameters

The models and hyperparameters for all adversarial training are chosen to be compatible with (Wong et al., 2020) and are included in Figure 5.

Hyperparameter	**MNIST**	**CIFAR10**
Architecture	CNN (16,32 filters, 100 FC units)	PreAct-ResNet18
Optimiser - momentum - weight decay	Adam n/a 0	SGD 0.9 0.0005
Number of training epochs	10	15
Cyclic learning rate maximum	0.005	0.2
Attack ℓ_∞ bound Attack step size	0.3 0.375	$8/255$ $10/255$

Figure 5: Hyperparameters

B Hypothesis Testing

In Section 4.1 we ran hypothesis tests on the results in Table 1 to test the following hypotheses:

1. Low distance pruning can give a more robust model than training with full data.

2. Low distance pruning with a loss distance proxy can give a more robust model than training with random pruning.

To write these hypotheses more formally we set $p = 0.6$ and use the following notation:

Let $A(m, p)$ be the AutoAttack accuracy of a model trained using pruning method m and data proportion p. Let $\mu(m, p) = \mathbb{E}(A(m, p))$. Our two tests can then be written as:

Test 1:

$$\mathbf{H}_0 : \mu(\text{low distance}, 0.6) \leqslant \mu(\text{none}, 1)$$
$$\mathbf{H}_1 : \mu(\text{low distance}, 0.6) > \mu(\text{none}, 1)$$

Test 2:

$$\mathbf{H}_0 : \mu(\text{low distance}, 0.6) \leqslant \mu(\text{random}, 0.6)$$
$$\mathbf{H}_1 : \mu(\text{low distance}, 0.6) > \mu(\text{random}, 0.6)$$

We repeated **Test 1** using both the *DeepFool* and loss distance proxies. Due to the potential difference in population variances we carry out one sided Welch's t-tests all with sample sizes of 120. The results are displayed in Figure 6.

Test	Distance Proxy	p value
1	*DeepFool*	0.108322
	Loss	0.000999
2	Loss	0.024602

Figure 6: Hypothesis test results.

C TOY MODEL FOR ROBUST FEATURES

In Section 4.1 we observed that low distance pruning could be used to improve the robustness of our models compared with full data training. We also saw that in most cases high distance pruning did not perform as well as low distance. Interestingly, we also found that low distance pruning disproportionately affected the standard accuracy of our models. In this section, we expand on the explanation involving robust and fragile features in Section 3.1 to give more in depth explanations for these two phenomena.

It was previously shown that the robustness of a model is highly linked to its ability to learn *robust features*, and that this is dependant on which features are most usefully predictive within the dataset (Ilyas et al., 2019). We present a natural toy model (inspired by (Tsipras et al., 2018)), in which high loss data-points correspond to those which are well predicted by fragile features, explaining why high-loss dropout is effective during adversarial training but disproportionately affects the standard accuracy of our models.

Binary classification The data model consists of input-label pairs $(\mathbf{x}, y)$, sampled from a a distribution $\mathcal{D}$, containing both points which are well predicted by robust features and points which are well described by fragile features:

$$(\mathbf{x}, y) \leftarrow \begin{cases} (\mathbf{x}, y) \sim \mathcal{D}_{\text{robust}} & \text{w.p. } p_{\text{R}} \\ (\mathbf{x}, y) \sim \mathcal{D}_{\text{fragile}} & \text{w.p. } 1 - p_{\text{R}} \end{cases} \tag{2}$$

Where our inputs are feature vectors, $\mathbf{x} = [x_1, \ldots, x_{d+1}]$ and we have that, for $L \in \{\text{robust}, \text{fragile}\}$, if $(\mathbf{x}, y) \sim \mathcal{D}_{\text{L}}$:

$$y \overset{u.a.r}{\sim} \{-1, +1\}, \qquad x_1 = \begin{cases} +y, & \text{w.p. } p_{\text{L}} \\ -y, & \text{w.p. } 1 - p_{\text{L}} \end{cases}, x_2, \ldots, x_{d+1} \overset{i.i.d}{\sim} \mathcal{N}(\eta y, 1). \tag{3}$$

Where x_1 is a robust feature, and $x_2 \ldots, x_{d+1}$ are 'fragile' features. We choose η large enough that a simple linear classifier can attain high standard accuracy (>99%), for which it suffices to set $\eta = 3/\sqrt{d}$. Furthermore, robust features are better correlated to the label for datapoints which are well described by robust features, i.e. $0.5 < p_{\text{fragile}} < p_{\text{robust}}$.

We can then define a simple linear classifier f on this data, described by:

$$f_{\text{pred}}(\mathbf{x}) := \text{sign}(\mathbf{w}^\top \mathbf{x}), \qquad \text{where } \mathbf{w} := [w_1, w_2, \ldots, w_{d+1}]. \tag{4}$$

Robust classifiers One finds that (see (Tsipras et al., 2018)), in the absence of an adversary a classifier relying purely on fragile features (i.e. where $\mathbf{w} = [0, \frac{1}{d}, \ldots, \frac{1}{d}]$) reaches high accuracy, however in the face of an adversary we find that the accuracy of this classifier drops rapidly. Indeed, for an adversary with a budget of $\epsilon \geq 2\eta$,

$$\begin{aligned}\min_{\|\delta\|_\infty \leq \epsilon} \mathbb{P}[f_{\text{pred}}(\mathbf{x}+\delta) = y] &= \min_{\|\delta\|_\infty \leq \epsilon} \mathbb{P}\left[\frac{y}{d}\sum_{i=1}^{d} \mathcal{N}(\eta y, 1) - \delta > 0\right] \\ &\leq \mathbb{P}\left[\mathcal{N}\left(\eta, \frac{1}{d}\right) - \epsilon > 0\right] \\ &\leq \mathbb{P}\left[\mathcal{N}\left(-\eta, \frac{1}{d}\right) > 0\right].\end{aligned} \tag{5}$$

This corresponds to a classification accuracy of less than 1%. On the other hand, a robust classifier $\mathbf{w} = [1, 0, \ldots, 0]$, reaches an accuracy of $p_{\text{R}} p_{\text{fragile}} + (1 - p_{\text{R}}) p_{\text{robust}}$, bounded from below by 50%.

Loss-based dropout The above discussion suffices to motivate the idea that we are likely to receive a classifier which focuses on robust features, after adversarial training. However, in the high-loss pruning case we then run such a classifier on standard data, and remove data points which have a high loss—what does our model have to say about this?

Assuming we have received a robust classifier ($\mathbf{w} = [1, 0, \ldots, 0]$) from adversarial training on a dataset $D = (\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^M, y^M)$, and then the high loss data points are dropped out, using the typical mean squared error loss function: $\mathcal{L}(f, \mathbf{x}, y) = |f(\mathbf{x}) - y|^2 = |\mathbf{w}^T \mathbf{x} - y|^2 = |x_1 - y|^2$.

We can then see that the data points $(\mathbf{x}, y)$ which are dropped out are the ones in which the value of x_1 is different to the value of y. This occurs with probability $1 - p_{\text{robust}}$ if $\mathbf{x} \in \mathcal{D}_{\text{robust}}$, and probability $1 - p_{\text{fragile}}$ if $\mathbf{x} \in \mathcal{D}_{\text{fragile}}$. As we have that $1 - p_{\text{robust}} < 1 - p_{\text{fragile}}$, we can see that given two data points from $\mathcal{D}_{\text{fragile}}$ and $\mathcal{D}_{\text{robust}}$, we preferentially drop those from $\mathcal{D}_{\text{fragile}}$, from which it is harder to learn robust features. As a result, we arrive at a dataset with a higher proportion of robust features compared to the original dataset.

This gives motivation for the empirical finding that the removal of high loss data points disproportionately affects the standard accuracy of a classifier, as we dropout points which are well described by fragile features. Standard accuracy is affected, as fragile features are still highly predictive of the label (Tsipras et al., 2018), but robust accuracy suffers less as datapoints enforcing robust accuracy are affected less by this dropout. In paradigms such as MNIST, where small subsets of the dataset suffice to achieve high accuracy (D'souza et al., 2020), we believe that this 'distillation' of the dataset into points which are well predicted by robust features is the mechanism by which we are able to actually improve the adversarial robustness of our classifier.

Conversely, removing low loss datapoints corresponds to removing datapoints from $\mathcal{D}_{robust}$, leading to a dataset wherein robust features are harder for models to learn. This provides a further explanation as to why low-loss dropout was found to be degrade model performance.

D FULL ALGORITHM

Algorithm 1 RS+FGSM based adversarial training of a classifier f_θ for E epochs, on a dataset S of size M, using learning rate η, subsampling method F, drop proportion d. The adversary here is constrained to the l_∞ ϵ-ball, and uses a step size of α.

procedure RSFFGSMTRAINING(f_θ,S)
  for $e = 1 \dots E$ **do**
    if e == pe **then**
      M' = {}
      for $i = 1 \dots M$ **do**
        M'.ADD((i, $\ell(f_\theta(x_i + \delta), y_i)$))
        SORT(M') ▷ using loss magnitude to sort data
        if F == low **then**
          Drop $d * |M|$ data from M' with low loss
        end if
        if F == high **then**
          Drop $d * |M|$ data from M' with high loss
        end if
        if F == random **then**
          Drop $d * |M|$ data from M' with probability $\frac{1}{|M|}$
        end if
        if F == high+low **then**
          SORT(M') ▷ using abs(loss magnitude - mean loss magnitude) to sort data
          Drop $d * |M|$ data with lowest cost
        end if
      end for
      $S \leftarrow S$
    end if
    for $i = 1 \dots M$ **do**
      $x_i, y_i \leftarrow$ GETDATAPOINT(i, S)
      $\delta \leftarrow$ UNIFORMRV$(-\epsilon, \epsilon)$ ▷ element-wise uniform initialisation in the l_∞ ball
      $\delta \leftarrow \delta + \alpha \operatorname{sgn}(\nabla_\delta \mathcal{L}(f_\theta(x_i + \delta), y_i))$ ▷ gradient descent to find adversarial example
      $\delta \leftarrow \max(\min(\delta, \epsilon), -\epsilon)$ ▷ element-wise clipping to keep δ in l_∞ ball
      $\theta \leftarrow \theta - \eta \nabla_\theta \ell(f_\theta(x_i), y_i)$
    end for
  end for
end procedure