The Academies' Summer Research Fellowship Programme 2020

Exploring Adversarial Robustness in Deep Neural Networks

Nischal A

B.Tech Computer Science and Engineering, Indian Institute of Technology Patna, Bihta, Bihar - 801103

Prof. R. Venkatesh Babu

Video Analytics Laboratory, Indian Institute of Science, CV Raman Road, Bengaluru, Karnataka - 560012

Abstract

Recent work has shown that training models to be robust against adversarial perturbations require larger data as compared to that required by standard classification (Schmidt et al.). Also, this problem has been addressed by making use of unlabelled data to bridge the sample complexity gap that is mentioned earlier. There has been significant improvement in the adversarial robustness with unlabelled data with robust self-training (Carmon et al.) and Unsupervised Adversarial Training (Uesato et al.). This improvement proves that unlabelled data is competitive as compared to labeled data for accounting for a larger dataset for improving adversarial robustness. However, this unlabelled data is taken from a similar data distribution. The authors observe that the inclusion of unrelated data doesn't help in improving the robustness. It is not always practical to assume the availability of additional unlabelled data from the same dataset. We propose to make use of open-source datasets that may not belong to the same distribution as the original dataset to improve adversarial robustness. In this work, I have studied the impact of varying learning rate schedules, batch sizes, and other training hyperparameters on the accuracy of the state-of-the-art adversarially trained models. Also, I predominantly experiment on the usage of unlabelled data from a different distribution as that of the original data. I make use of the CIFAR-100 dataset as the unlabelled data, and generate the pseudo labels for this through the process of self-training as described in Carmon et al. I also experiment on the different weights of the loss function, by assigning different weights to the four losses namely cross-entropy on labeled data, cross-entropy on unlabelled data, KL (kullback-leblier) on labeled data and KL (kullack-leblier) on unlabelled data. Through this, we can get an empirical idea of the effect of the different components of the loss in improving the final robust accuracy of the model.

Keywords or phrases: Robust self-training, Unlabelled data, Open Source Dataset, CIFAR-100, Tiny Images Dataset

Abbreviations

SRF	Summer Research Fellowship
IAS	Indian Academy of Science
RST	Robust Self Training
KL	Kullback-Leibler loss
ce_lab	Weight for the cross-entropy loss on labeled data
ce_unlab	Weight for the cross-entropy loss on unlabelled data
kl_lab	Weight for the KL loss on labeled data
kl_unlab	Weight for the KL loss on unlabelled data

INTRODUCTION

Background/Rationale

In recent years, deep neural networks have become exceedingly capable of learning the exact function representations of the task at hand. In fact, deep neural net-based solutions are performing significantly better than humans on traditional image classification tasks (ImageNet Challenge). However, it is noticed that these highly accurate models are not robust to small perturbations in the input i.e. when a carefully constructed vector is added to the original input thereby perturbing it, the classifiers can be fooled to misclassify the input (even though the change isn't perceivable by humans). Adversarial Training is a method for creating adversarial robust models that aren’t affected by small perturbations in the inputs. In recent years there has been significant development in both the adversarial attacks and the defenses against these attacks. However, it is empirically proven that there exists a sample complexity gap in achieving

the same robust accuracy as that of the standard accuracy (Schmidt et al) i.e. a large amount of data is required for achieving high robust accuracy. Since labeled data is expensive, recent works have shown that unlabelled data works as well as labeled ones for accounting for the sample complexity gap (Carmon et al; Uesato et al). However, these works utilize the unlabelled data from the same distribution as the labeled data. Here, I experiment with making use of the unlabelled data from a different distribution for adversarial robustness.

Statement of the Problems

- To explore the usage of unlabelled data from a different distribution for improving adversarial robustness.
- To explore the different components of the loss function that help in improving adversarial robustness.

Objectives of the Research

Overall objective

The overall objective of this work is to present using unlabelled data from a different distribution as a candidate for improving the adversarial robustness over the vanilla adversarial training (without using the unlabelled data). We cannot always expect the unlabelled data to belong to the same data distribution as that of the labeled one. Hence, we explore whether making use of a generalized data distribution can help in improving the adversarial robustness.

LITERATURE REVIEW

Information

The need for additional data for creating robust classifiers was first shown by Schmidt et al (see references). Schmidt et al. showed in their work that there exists a sample complexity gap in achieving the same robust accuracy as the clean accuracy for a classification task using CIFAR-10 as the dataset.

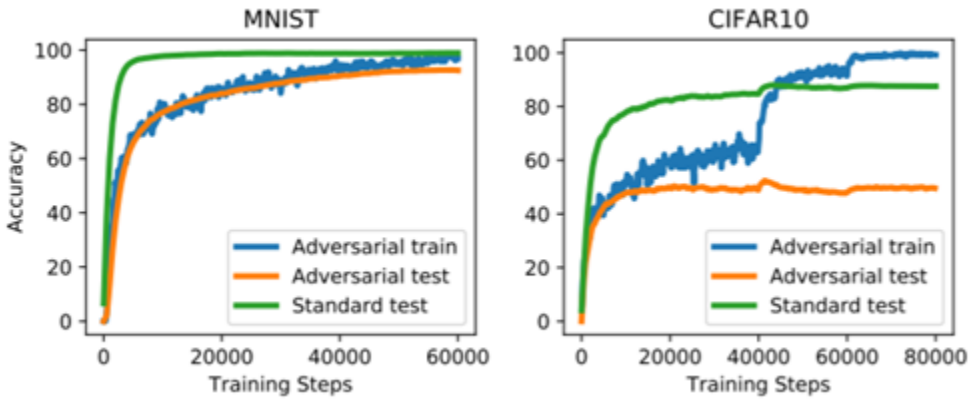

Fig 1 (Left) Plot of accuracy vs training steps for MNIST dataset; (Right) The same plot for CIFAR-10.

We can see from Fig 1 (right) that there exists a gap in the standard and the adversarial accuracy for CIFAR-10 adversarial training. When the number of training steps has increased the gap between the standard and the adversarial train accuracy decreases, but the gap with the adversarial test accuracy still remains. Hence, we can conclude that there is a need for extra data in the case of CIFAR-10 for bridging this gap. However, this gap isn't much prominent with the MNIST dataset Fig 1 (left), the reason being MNIST has sufficient data to achieve high robust accuracy.

Following this work, Carmon et al, proved that using unlabelled data improves adversarial robustness.

The main questions addressed in their work are: -

- How can we account for additional data for improving robustness?
- How to get additional labeled data? Since labeling may be an expensive process.

The solution to the questions is a Semi-Supervised Adversarial Training Algorithm. Here they propose an algorithm Robust Self Training (RST) which is based on: -

- Taking unlabelled data and generating pseudo labels from them (using a network pre-trained with the labeled data)
- Mixing the unlabelled data and the labeled data in a definite proportion.
- Performing adversarial training (TRADES) on this dataset.

The reasoning for using unlabelled data for bridging the sample complexity gap is: -

- Labelling Data is generally an expensive and tedious process.
- Adversarial Robustness requires the predictions to be stable around naturally occurring inputs. Achieving this doesn't really require labels.

The pseudocode of the RST algorithm is shown below: -

Meta-Algorithm 1 Robust self-training

Input: Labeled data $(x_1, y_1, \ldots, x_n, y_n)$ and unlabeled data $(\tilde{x}_1, \ldots, \tilde{x}_{\tilde{n}})$

Parameters: Standard loss L_{standard}, robust loss L_{robust} and unlabeled weight w

1: Learn $\hat{\theta}_{\text{intermediate}}$ by minimizing $\sum_{i=1}^{n} L_{\text{standard}}(\theta, x_i, y_i)$

2: Generate pseudo-labels $\tilde{y}_i = f_{\hat{\theta}_{\text{intermediate}}}(\tilde{x}_i)$ for $i = 1, 2, \ldots \tilde{n}$

3: Learn $\hat{\theta}_{\text{final}}$ by minimizing $\sum_{i=1}^{n} L_{\text{robust}}(\theta, x_i, y_i) + w \sum_{i=1}^{\tilde{n}} L_{\text{robust}}(\theta, \tilde{x}_i, \tilde{y}_i)$

Fig 2 RST algorithm Pseudocode

In general, the training is based on the TRADES paper, hence, $L_{standard}$ is cross-entropy and L_{robust} is KL loss.

The main point to note here is that the unlabelled data used is taken from the Tiny Images 80M dataset which is a superset of the labeled data (CIFAR-10). 500K images are chosen from the Tiny Images dataset that is most similar to the original 10 classes of the CIFAR-10 dataset without including duplicates. Hence, the unlabelled data used here is from the same distribution as the labeled data.

Summary

Hence, from the above papers, we have found out that, using unlabelled data can prove to be beneficial for improving adversarial robustness. In my work, I specifically make use of unlabelled data from a different distribution (CIFAR-100) for finding its effects on the adversarial robustness.

METHODOLOGY

Concepts

To address the problem of misclassification of adversarial examples we predominantly use the following two techniques: -

- Standard Adversarial Training - (Goodfellow et al)
- TRADES (TRadeoff-inspired Adversarial DEfense via Surrogate-loss minimization) - (Zhang et al)

1) Standard Adversarial Training

Here, we augment the original dataset with the perturbed inputs which are perturbed based on a finite norm ball. This augmented dataset is used for normal training of the classifier i.e. to reduce the objective function.

$$\underset{\theta}{\text{minimize}}\, \hat{R}_{\text{adv}}(h_\theta, D_{\text{train}}) \equiv \underset{\theta}{\text{minimize}}\, \frac{1}{|D_{\text{train}}|} \sum_{(x,y)\in D_{\text{train}}} \max_{\delta\in\Delta(x)} \ell(h_\theta(x+\delta)), y).$$

Fig 3 Optimization problem solved in standard adversarial training

All terms used have standard meanings. The function 'h' represents a way to perturb the input. Few examples of such methods are: -

- FGSM (Fast Gradient Sign Method)
- I-FGSM (Iterative Fast Gradient Sign Method)
- PGD (Projected Gradient Descent)

As we can see from Fig 3, the optimization problem requires us to solve a min-max problem. At each step, the maximum perturbed input (perturbation is caused using a definite small norm-ball) is chosen and the model is trained to reduce the standard classification loss function on it (generally the cross-entropy).

2) TRADES

The TRADES method minimizes a regularized surrogate loss L(.,.) (e.g., the cross-entropy loss) for adversarial training:

$$\min_f \mathbb{E}\left\{\mathcal{L}(f(X), Y) + \beta \max_{X'\in\mathbb{B}(X,\epsilon)} \mathcal{L}(f(X), f(X'))\right\}.$$

Fig 4 TRADES optimization

In the above TRADES optimization problem we can see that: -

- The First Term (Standard Loss) - helps to minimize the standard (natural) error on the samples. This is done by enforcing f(X) the prediction from the neural network to be similar to the ground truth value.

- The Second Term (Robust Loss) - enforces smoothness in the decision boundary, by accounting for the error caused by the difference between the prediction of the natural example f(X) and the adversarially perturbed example f(X')
- The Tuning parameter β helps to weigh the standard loss and the robust loss.

Together, they form the objective function that will be optimized during the adversarial training.

f(X) is the prediction distribution from the neural network, and Y is the original distribution.

It is proven that TRADES adversarial training gives higher robust accuracy and is an improvisation on the standard adversarial training.

Methods

I have experimented with the Robust Self Training algorithm in general (Figure 2). I have varied various parameters like the learning rate scheduler, batch size, unsupervised fraction, also I have included a special train-validation split and new evaluation functions.

The main part of the work is making use of unlabelled data from a different distribution, here (CIFAR-100) for the RST code.

The method of preparing the unlabelled data is as follows: -

- Consider the CIFAR-100 dataset and remove all the labels associated with it.
- Pass the CIFAR-100 dataset through a network pre-trained with the CIFAR-10 dataset. This generates a pseudo-label for each of the 50K images of the CIFAR-100 dataset.
- We finally convert this dataset into a pickle file as required by the RST algorithm and use this as the unlabelled data.

Note - In step 2 above, I have pre-trained a Wide-Resnet-28-10 model using CIFAR-10. This model is further used for generating pseudo labels for the CIFAR-100 dataset.

Hence, I explore the use of the new unlabelled data in the context of adversarial robustness.

In general, the method followed is: -

- Forming the dataset by mixing the labeled and unlabelled data in a fixed proportion governed by the unsupervised-fraction (hyperparameter)
- Adversarial training (TRADES) with the given configuration
- Choosing the best epoch using PGD-20 on the validation set (or test set). (Optional)
- Evaluating the performance of the model on the test set, against various attacks like FGSM, IFGSM, PGD.

The models used are Resnet-18, and Wide-Resnet-28-10. Adversarial training involves training with a 10 step PGD adversary.

RESULTS AND DISCUSSION

Note the default parameters for the run are: -

- batch_size = 64
- dataset - CIFAR-10
- pgd_num_steps=10
- pgd_step_size=0.007
- unsup_fraction=0.5 (See appendix 1)
- weight_decay=0.0005
- learning_scheduler = cosine

The above parameters are considered default unless otherwise mentioned.

1) **Train-Eval Setup (without unlabelled data) (Table 1)**

Description - This is used for experimenting with the different Train and Validation splits to see which combination gives the best results. In each cell the format of the result is (epoch, PGD-20 validation accuracy) i.e. epoch value represents that epoch that has the highest accuracy on the corresponding column attack and dataset. Corresponding to this epoch test PGD-20 accuracy is found.

In the table below we see that 49K-1K split gives the best results.

Table 1Different Train - Eval setups

Train Size	Val Size	IFGSM-7 Val	IFGSM-20 Val	PGD-20 VAL	PGD-20 Test
49000	1000	(95, 52.14)		(109, 52.26)	(117, 52.59)
47500	2500	(120, 52.19)		(103, 52.23)	(117, 52.64)
45000	5000	(105, 52.05)	(105, 52.05)	(105, 52.05)	(112, 52.29)
40000	10000	(108, 51.82)	(108, 51.82)	(95, 51.46)	(91, 51.93)

2) **49K-1K Vanilla Runs (Without unlabelled data) (Table 2)**

Description - After choosing 49k-1k as the desired train-val split, I now experiment on different learning rate schedulers. Here, we see that cyclic learning rate (lr_min - 0; lr_max - 0.2; step_size = num_epochs/2;) gives the best results.

trades_120 - step learning rate at 75; 90 and 100;

trades_200 - step learning rate at 125, 150, 170

Table 249K-1K Vanilla Runs

Lr Schedule	IFGSM-7 Val	IFGSM-20 Val	PGD-20 VAL	PGD-20 Test
trades_120	(95, 52.14)		(109, 52.26)	(117, 52.59)
trades_200	(155, 51.94)	(155, 51.94)	(155, 51.94)	(157, 52.58)
cosine	(186, 51.4)	(186, 51.4)	(165, 51.7)	(174, 51.87)
cyclic	(200, 53.05)	(200, 53.05)	(200, 53.05)	(200, 53.05)

3) **Results with Unlabelled Data: -**

The unlabelled data used here is the data used in Carmon et al. Here, I have experimented further with it, by using different learning rate schedules. The result marked in yellow suggests that the PGD-20 attack reported is used directly in Carmon et al. The PGD-20 used in Carmon et al is not as strong as our Pgd-20 attack (ours includes random noise).

This run is for validating the fact that unlabelled data is useful for adversarial robustness. We can see that the accuracy obtained with Wide-ResNet-28-10 is 63.6% and is higher than the previous works (note this run is exactly the same as that of Carmon et al) and similarly Resnet-18 gives higher robust (PGD-20) accuracy with unlabelled data as compared to without.

Results - with TinyImages_500k unlabelled data

Train Size	Val Size	Model	Lr schedule	lfgsm -7	lfgsm - 20	Pgd -20 val	Pgd - 20 test
50000	-	Wrn - 28 -10	cosine				(197, 63.6)
50000	-	Resnet-18	cosine				(187, 57.2)
49000	1000	Resnet-18	cosine	(195, 56.42)	(197, 56.45)	(197, 56.45)	(197, 56.45)
49000	1000	Resnet-18	Trades200 Step - 125, 150, 170	(190, 54.64)	(190, 54.64)	(200, 54.9)	(186, 54.96)

Fig 5 Table - 2a Tiny Images - 500K results

4) **Results Using CIFAR-100 as unlabelled data**

Here, we use data from a different distribution i.e. CIFAR-100 for study. After generating the pseudo labels for CIFAR-100 we have a distribution of the number of images per class shown below. The dataset is more or less balanced.

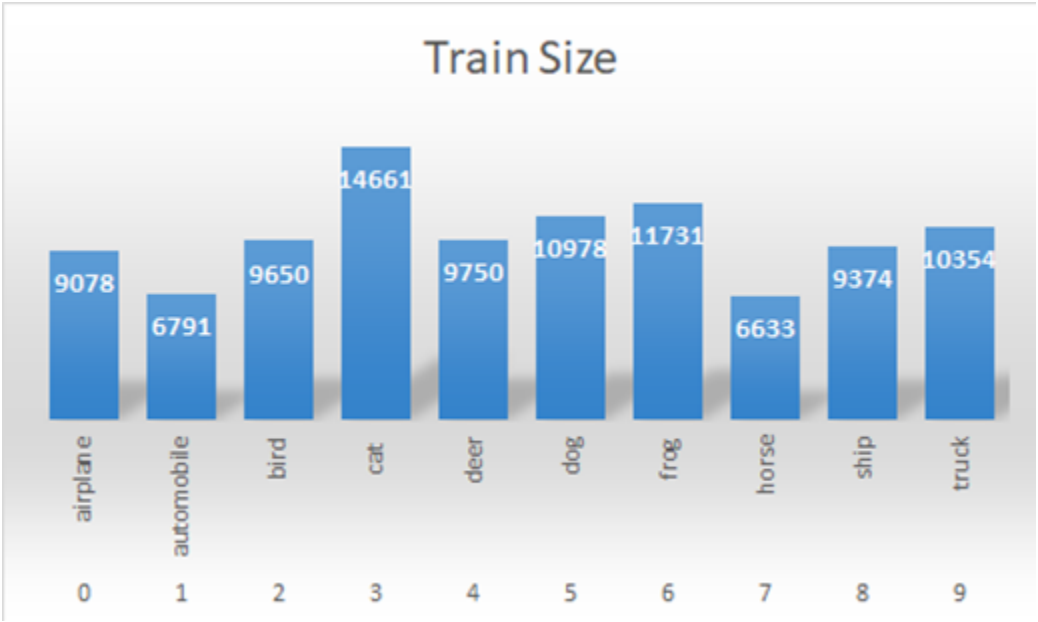

Fig 6 CIFAR-100 PseudoLabeled Distribution

Here, I have experimented with different parameters like the LR scheduler, unsupervised-fraction. We can see in the below table (Fig 7 - Table 2b) that, using CIFAR-100 as the unlabelled data isn't giving as good a results as using Tiny Images (500K), though, the results with 0.25 unsupervised fraction is much better as compared to vanilla adversarial training - Trades (i.e. without using unlabelled data)

Train Size	Val Size	Lr schedule	Unsup frac	lfgsm -7	lfgsm - 20	Pgd -20 val	Pgd - 20 test
50000	-	cosine	0.5				(156, 53.90)
50000	-	cosine	0.25				(182, 55.46); (194, 52.38)
49000	1000	Trades_200 Step (125, 150, 170)	0.25	(188, 51.31)	(189, 51.44)	(161, 50.76)	(190, 51.68)
49000	1000	cosine	0.25	(194, 51.98)	(194, 51.98)	(184, 51.37)	(195, 52.43)
49000	1000	cosine	0.5	(198, 49.42)	(198, 49.42)	(186, 49.08)	(194, 49.90)

Fig 7 Table - 2b Cif -100 used as unlabelled data

5) **Experimenting with the loss function** (See Appendix 3)

a) Table 3 below shows the parameters of the training

Table 3Parameters for TI_500K Loss Experiment Run

Dataset	Unsup_fraction	ce_lab	ce_unlab	kl_lab	kl_unlab
TI_500K dataset	0.5	2	0	6	6
CIFAR-100-50K dataset	0.25	1	0	6	2

Table 4Loss Experiment Results with parameters from Table 3

Clean Accuracy	FGSM	IFGSM-20	PGD-7	PGD-20	PGD-100

81.8	56	53.35	52.81	50.45	50.29
49.72	26.39	25.42	25.33	24.41	24.36

Comments -

- Table 4 (ROW 1) shows that the results on PGD-20 i.e. 50.45 is lesser than the normal configuration (TI_500K) with results mainly because the ce_unlab is eliminated and made zero.
- Table 4 (Row 2) shows that the accuracy using CIFAR-100 with this configuration is much lesser, due to a decrease in ce_unlab and kl_unlab parameters.

CONCLUSION AND RECOMMENDATIONS

Conclusion

From the above results (Fig 7) we have seen that using CIFAR-100 with 0.25 unsupervised-fraction improves the adversarial robustness as compared to the vanilla runs (i.e. without using the unlabelled data).

However, we also observe that with unsupervised-fraction 0.5 the results till lag behind the vanilla case. This is because the data distribution of the unlabelled data is different. Hence, it is in a way hampering the self-training process of the RST algorithm.

Also, CIFAR-100 is not performing as good as the default 500K images of the Tiny Images 80M Dataset because: -

- The CIFAR-100 is from a different distribution whereas, the TI 500K dataset is from the same

- The amount of unlabelled data with CIFAR-100 (50K) is far less than the TI Images (500K).

Hence, in this work, I have explored the importance of unlabelled data and more specifically explored the usage of unlabelled data from a different distribution for adversarial robustness.

Also, by experimenting with different combinations of the loss functions, we can see that the best configuration is (ce_lab = 1, ce_unlab = 1; kl_lab = 6, kl_unlab=6). More research can be carried out on these lines to further validate this result.

Also, with these results, we can further use techniques from DeGan (Sravanti et al) to incorporate the features of the same data distribution in the CIFAR-100 unlabelled distribution so that it works better.

REFERENCES

In this work I have referred to various papers listed below: -

- Using Pre-training can improve model robustness (ICML 2019)
- Unlabeled data improves adversarial robustness (NeurIPS 2019)
- Are labels required for improving adversarial robustness? (NeurIPS 2019)
- TRADES (ICML 2019)
- Towards Deep Learning Models Resistant to Adversarial Attacks
- Adversarially Robust Generalization requires more data (NeurIPS 2018)

ACKNOWLEDGEMENTS

I would like to thank my guide Prof. R. Venkatesh Babu for his support and guidance in carrying out this work. I would also like to thank my mentor Sravanti for her guidance and support.

I would like to thank my family for supporting me. Most importantly, I express deep gratitude to the Indian Academy of Sciences (IAS) for giving me a wonderful opportunity to carry out this research at the Indian Institute of Science (IISc).

APPENDICES

1. **Preparing the Dataset for RST**[1]

Here, at each training epoch, we mix the labeled and unlabelled data to form the complete dataset. The unsupervised fraction governs the mix. For example, if unsupervised-fraction is 0.5, then in every epoch 50% of data is labeled and 50% unlabelled. If the unsupervised fraction is 0.25, then in every epoch 75% of data is labeled and 25% unlabelled.

2. **Loss Functions**

Here, the loss functions we use are: -

$L_{standard}$ - Cross-Entropy Loss.

L_{robust} - KL Loss considering the output distribution of the perturbed input and the normal input itself.

3 . **Experimenting with the Loss function**

Total Loss = ce_lab*$L_{standard}$(lab_data) **+** ce_unlab*$L_{standard}$(unlab_data) **+** kl_lab*L_{robust}(lab_data) **+** kl_unlab*L_{robust}(unlab_data)

[1]