Boundary Adversarial Examples Against Adversarial Overfitting

Muhammad Zaid Hameed
IBM Research Europe
Dublin, Ireland
Zaid.Hameed@ibm.com

Beat Buesser
IBM Research Europe
Dublin, Ireland
beat.buesser@ie.ibm.com

Abstract

Standard adversarial training approaches suffer from robust overfitting where the robust accuracy decreases when models are adversarially trained for too long. The origin of this problem is still unclear and conflicting explanations have been reported, i.e., memorization effects induced by large loss data or because of small loss data and growing differences in loss distribution of training samples as the adversarial training progresses. Consequently, several mitigation approaches including early stopping, temporal ensembling and weight perturbations on small loss data have been proposed to mitigate the effect of robust overfitting. However, a side effect of these strategies is a larger reduction in clean accuracy compared to standard adversarial training. In this paper, we investigate if these mitigation approaches are complimentary to each other in improving adversarial training performance. We further propose the use of *helper adversarial examples* that can be obtained with minimal cost in the adversarial example generation, and show how they increase the clean accuracy in the existing approaches without compromising the robust accuracy.

1 Introduction

Adversarial examples have gained great attention in ML research [1, 2, 3] and building models with increased robustness against such examples is still an open challenge [4, 5, 6, 7, 8, 9, 10, 11]. Adversarial training (AT) [4] is the most successful approach against adversarial examples where approximate worst-case adversarial examples are used to train the model.
The success of AT is impeded by high computational costs compared to standard training and gaps between robust and clean accuracy [4, 5, 12, 13]. It has been shown that AT reduces the clean accuracy [5] and mitigation techniques based on label smoothing and stochastic weight averaging [11] or use of additional unlabeled data [14] have been proposed, which solve the problem only partially.
Recently, another phenomenon associated with adversarial training has been discovered and coined *robust overfitting* [15, 11, 16, 17], which occurs when models are adversarially trained for too long. Especially after the first learning rate decay, the robust accuracy tends to decrease with additional training. Even though overfitting to training data results in increased clean accuracy, the robust accuracy is negatively impacted by this additional training. The exact causes of robust overfitting are still unclear and most explanations are conflicting. For example, memorization effects induced by large loss data, which give rise to high confidence predictions, have been shown to be a driving force behind robust overfitting [16]. On the other hand, small loss data and large differences in loss distribution across training samples, tend to amplify as AT progresses, and have been shown to enhance robust overfitting [17]. Reported mitigation approaches like temporal ensembling (TE) [16] and model weight perturbation (WP) [17] alleviate robust overfitting, but tend to trade off more clean accuracy when compared to standard adversarial training (AT) [4]. This shows that increased

regularization induced by these schemes negatively impact the generalization performance on clean data.

Here, we analyze how these two seemingly conflicting approaches, TE and WP, which attribute robust overfitting to training with samples with large and small loss data, respectively, mitigate robust overfitting and if they can be combined to create a compounding mitigation effect. First, we investigate how TE and WP behave during AT and find that they induce different types of regularization on adversarial examples during overfitting. Second, we evaluate a combination of TE and WP and find that there is no compounding improvement in robust accuracy. Third, we minimize loss on clean input data along-with these robust overfitting mitigation approaches which results in increased clean accuracy at the expense of robust accuracy. Thus, we find that the negative impacts on clean accuracy induced by TE and WP can be prevented without affecting the robust accuracy with appropriate regularization.

We propose to employ TE/WP based robust overfitting mitigation with the additional use of helper adversarial examples along-with a TE-based regularization term and demonstrate improvement in clean accuracy over the existing AT approaches. The helper examples can be created during the adversarial example generation in AT with a minimal computational overhead of just one extra forward pass. Our idea is to use perturbed samples close to decision boundaries as helper examples with their property of increasing clean accuracy with only a small degradation in robust accuracy compared to using clean data only.

2 Preliminaries

Adversarial training (AT) has empirically proved to be the most effective defense strategy against adversarial evasion attacks [4, 5, 6, 10, 11]. In the following we will briefly describe the mechanics of AT [4] and two robust mitigation schemes [16, 17].

2.1 Adversarial Training (AT) [4]

We consider a classification problem where an input sample $\mathbf{x} \in \mathbb{R}^n$ belongs to a true class y among a set of classes $\mathcal{Y} = \{1, 2, \ldots, Y\}$. A DNN-based classifier $F_{\boldsymbol{\theta}} : \mathbb{R}^n \times \mathcal{Y} \to \mathbb{R}$, assigns a label $\hat{y} \in \arg\max_{y \in \mathcal{Y}} F_{\boldsymbol{\theta}}(\mathbf{x}, y)$ to $\mathbf{x}$, where $\boldsymbol{\theta} \in \Theta$ denotes the parameters of DNN. With a slight abuse of notation, we also use F to denote the classifier, $F(\mathbf{x})$ to denote the class label assigned to $\mathbf{x}$, and $F(\mathbf{x}, y)$ to denote the score of class y for input $\mathbf{x}$ and $p(\mathbf{x})$ denotes the DNN prediction probability vector. For a correctly classified input $\mathbf{x}$ i.e., $y = F(\mathbf{x})$ is the true label, an adversarial attack aims to find a sample $\tilde{\mathbf{x}}$ in the ϵ-neighborhood of $\mathbf{x}$ in norm p denoted by $\mathcal{B}_\epsilon(\mathbf{x}) = \{\tilde{\mathbf{x}} : \|\tilde{\mathbf{x}} - \mathbf{x}\|_p \leq \epsilon\}$, such that $F(\tilde{\mathbf{x}}) \neq F(\mathbf{x})$. In practice, these adversarial examples are generated by modifying the input $\mathbf{x}$ through optimizing a loss function $\mathcal{L}$ on the classifier [3, 18, 19, 20, 4]. Finally, AT can be considered as a robust optimization problem:

$$\min_{\boldsymbol{\theta}} \max_{\tilde{\mathbf{x}} \in \mathcal{B}_\epsilon(\mathbf{x})} \mathcal{L}(F(\tilde{\mathbf{x}}, y), y), \tag{1}$$

where $\tilde{\mathbf{x}}$ is the adversarial example. Hence, during training, adversarial attacks first generate adversarial examples in the neighborhood $\mathcal{B}_\epsilon(\mathbf{x})$ of input $\mathbf{x}$ to approximately maximize the loss $\mathcal{L}$ (e.g., cross-entropy loss), followed by training on the examples to update the network parameters and achieving robustness against adversarial examples.

2.2 Adversarial Training with Temporal Ensembling (AT-TE) [16]

Robust overfitting is attributed to neural networks training on high loss input data and over-confident predictions (memorization) by the model during AT [11, 16]. To prevent this, a regularization based on temporal ensembling (TE) has been proposed in [16] which penalizes over-confident predictions. Let $z(\mathbf{x})$ denote the ensemble prediction by a model on input $\mathbf{x}$ which is updated as $z(\mathbf{x}) = \eta \cdot z(\mathbf{x}) + (1 - \eta) \cdot p(\mathbf{x})$ in each epoch and the AT objective becomes

$$\min_{\boldsymbol{\theta}} \max_{\tilde{\mathbf{x}} \in \mathcal{B}_\epsilon(\mathbf{x})} \{\mathcal{L}(F(\tilde{\mathbf{x}}, y), y) + w \cdot \|p(\tilde{\mathbf{x}}) - \hat{z}(\mathbf{x})\|_2^2\}, \tag{2}$$

where $\hat{z}(\mathbf{x})$ is obtained by normalizing $z(\mathbf{x})$ and w is regularization weight. In this AT, the regularization term is activated close to the first learning rate decay, and prevents the network from assigning high confidence to samples with large loss.

2.3 Minimum Loss Constrained Adversarial Training (MLCAT-WP) [17]

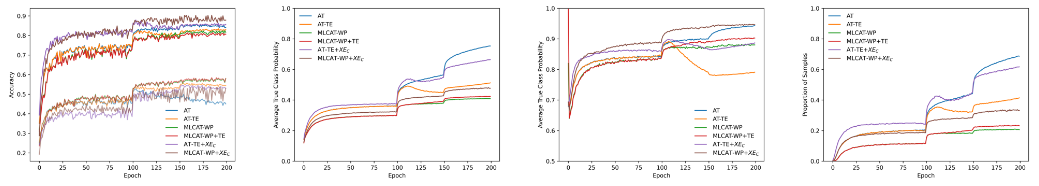
Training samples with small loss and large differences in loss distribution among samples are shown to cause robust overfitting in [17] and adversarial weight perturbation [9] is employed to increase the loss on such samples, which helps to mitigate robust overfitting. The training objective becomes

$$\min_{\boldsymbol{\theta}} \max_{\mathbf{v}\in\mathcal{V}} \max_{\tilde{\mathbf{x}}\in\mathcal{B}_\epsilon(\mathbf{x})} \mathcal{L}(F_{\theta+v}(\tilde{\mathbf{x}}, y), y), \tag{3}$$

where $v \in \mathcal{V}$ is an adversarial weight perturbation and is generated by $\max_{\mathbf{v}\in\mathcal{V}} \sum_i \mathbb{1}_{\{\mathcal{L}_i \leq L_{min}\}} \cdot \mathcal{L}_i$, where $\mathcal{L}_i = \mathcal{L}(F_{\theta+v}(\tilde{\mathbf{x}}_i, y_i), y_i)$, L_{min} is threshold for minimum adversarial loss and $\mathbb{1}_c$ is an indicator function which is 1 only when condition c is true.

3 Comparison of AT, AT-TE and MLCAT-WP

In order to investigate how standard AT and robust overfitting mitigation approaches AT-TE and MLCAT-WP behave as training progresses, we consider image classification on CIFAR-10 [21] with a ResNet-18 model [22]. For all approaches, we use a Projected Gradient Descent (PGD) attack [4] for 10 steps denoted by PGD_{10} in training and PGD_{20} for evaluation; see Appendix A for full details. We observe from Figure 1(b)-(d) that as the training progresses, an AT model starts to classify training data with high true class probability (TCP) (Average TCP ≥ 0.5) after the first learning rate decay at epoch 100 and proportion of training samples with very small loss value (in range [0, 0.5)) and high TCP also grows ($\geq 40\%$ of all training samples). On the other hand, an MLCAT-WP trained model's average TCP on training data increases gradually and stays small (≈ 0.4) at epoch 200 and the proportion of samples in the loss range [0, 0.5) amounts to only 20% of all training data which shows weight perturbation prevents the model from assigning very high TCP to training data and fitting samples with small loss. In case of AT-TE the average TCP for samples with loss in range [0, 0.5) starts to decrease when temporal ensembling is activated (epoch ≥ 90) and even though the proportion of samples increase, it lies between AT and MLCAT-WP. Finally, we can also make this observation from Figure 1(a) that combining weight perturbation and temporal ensembling (MLCAT-WP+TE) does not result in any improvement in robust accuracy and its behavior closely resembles MLCAT-WP. Moreover, modifying AT-TE and MLCAT-WP to AT-TE+$\mathrm{XE_C}$ and MLCAT-WP+$\mathrm{XE_C}$ by including cross entropy loss on clean data results in higher clean accuracy at the cost of reduced robust accuracy and results in increase in TCP and proportions of samples with small loss.



(a) Test Data Accuracy **(b)** TCP **(c)** TCP, loss $\in$ [0, 0.5) **(d)** Sample Proportion, loss $\in$ [0, 0.5)

Figure 1: CIFAR-10 training for ResNet-18. (a) Test accuracy against clean data (dark solid lines) and PGD_{20} attack (dim solid lines) are plotted.

4 Boundary Adversarial Examples for Improving Adversarial Training

To counter the negative effect of regularization in AT-TE and MLCAT-WP on clean accuracy and negative effect of using clean sample on robust accuracy, we propose to extract additional useful information from the adversarial examples generation process in adversarial training. More specifically, we extract intermediate adversarial examples that are close to a decision boundary as soon as the perturbed sample is misclassified. Our underlying idea is that using boundary (intermediate and weak) adversarial examples in place of clean samples will guide the network to attain better clean accuracy without affecting robust accuracy too much. Thus, using this intermediate perturbed sample $\mathbf{x}' \in \mathcal{B}_\epsilon(\mathbf{x})$ and a regularization based on its prediction $p(\mathbf{x}')$ and ensemble prediction $z(\mathbf{x}')$, the AT-TE objective becomes

$$\min_{\boldsymbol{\theta}} \{\mathcal{L}(F(\mathbf{x}', y), y) + w \cdot \|p(\mathbf{x}') - \hat{z}(\mathbf{x}')\|_2^2 + \max_{\tilde{\mathbf{x}}\in\mathcal{B}_\epsilon(\mathbf{x})} \{\mathcal{L}(F(\tilde{\mathbf{x}}, y), y) + w \cdot \|p(\tilde{\mathbf{x}}) - \hat{z}(\mathbf{x})\|_2^2\}\}, \tag{4}$$

where $\hat{z}(\mathbf{x}')$ is normalized $z(\mathbf{x}')$, and $z(\mathbf{x}')$ is updated as $z(\mathbf{x}') = \eta \cdot z(\mathbf{x}') + (1 - \eta) \cdot p(\mathbf{x}')$ in each epoch. We denote this modified AT-TE as AT-$\mathrm{TE_{BS}}$. Similarly, MLCAT-WP objective becomes MLCAT-WP+$\mathrm{TE_{BS}}$ by including TE and boundary sample as

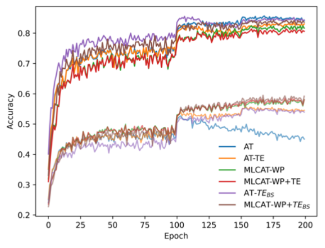
$$\min_{\boldsymbol{\theta}} \{\mathcal{L}(F_{\theta+v}(\mathbf{x}', y), y) + w \cdot \|p(\mathbf{x}') - \hat{z}(\mathbf{x}')\|_2^2 + \max_{\mathbf{v}\in\mathcal{V}} \max_{\tilde{\mathbf{x}}\in\mathcal{B}_\epsilon(\mathbf{x})} \{\mathcal{L}(F_{\theta+v}(\tilde{\mathbf{x}}, y), y) + w \cdot \|p(\tilde{\mathbf{x}}) - \hat{z}(\mathbf{x})\|_2^2\}\} \tag{5}$$

Table 1: Test accuracy (mean and standard deviation of 5 runs). “Last" and “Best" refer to test accuracy at the end of training, and end of epoch that gives the highest accuracy w.r.t. test data respectively.

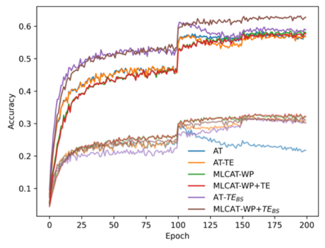
Dataset	Name	Clean(Best)	Clean(Last)	Robust(Best)	Robust(Last)	AA(Best)	AA(Last)
CIFAR-10	AT	82.03±0.42	84.37±0.30	52.64±0.13	45.06±0.70	48.04±0.15	42.64±0.62
	AT-TE	82.11±0.14	82.73±0.22	55.69±0.11	54.15±0.42	49.99±0.16	48.95±0.37
	MLCAT-WP	82.05±0.31	81.78 ±0.26	58.16±0.13	57.46±0.45	50.43±0.07	49.96±0.31
	MLCAT-WP+TE	81.06±0.49	80.74±0.24	58.48±0.08	57.71±0.22	**50.67±0.10**	**50.46±0.20**
	AT-TE$_{\text{BS}}$	**83.99±0.14**	**84.46±0.36**	55.26±0.07	53.58±0.74	50.17±0.09	48.80±0.70
	MLCAT-WP+TE$_{\text{BS}}$	83.712±0.51	83.75±0.41	**59.20±0.21**	**58.53±0.52**	50.38±0.07	50.30±0.26
CIFAR-100	AT	55.49±0.60	56.93±0.15	29.50±0.22	22.11±0.25	25.05±0.44	19.97±0.19
	AT-TE	56.43±0.25	56.94±0.44	32.07±0.06	30.67±0.18	26.19±0.18	25.1±0.09
	MLCAT-WP	57.65±0.41	57.96±0.30	32.73±0.11	31.68±0.61	**27.14±0.14**	26.48±0.41
	MLCAT-WP+TE	56.60±0.56	57.44±0.33	**33.02±0.16**	**32.02±0.56**	27.08±0.16	**26.58±0.54**
	AT-TE$_{\text{BS}}$	58.78±0.23	58.82±0.29	31.59±0.09	30.40±0.16	25.99±0.18	24.98±0.23
	MLCAT-WP+TE$_{\text{BS}}$	**62.50±0.28**	**62.45±0.42**	31.95±0.06	30.72±0.65	26.57±0.14	25.74±0.48
SVHN	AT	89.08±0.47	89.91±0.30	53.21±0.32	44.70±0.60	45.59±0.61	39.97±0.63
	AT-TE	89.17±0.58	89.83±0.34	53.21±0.18	44.24±0.71	45.81±0.43	39.65±0.81
	MLCAT-WP	91.45±0.26	91.91±0.21	60.25±0.28	56.98±0.62	**51.60±0.37**	49.20±0.30
	MLCAT-WP+TE	91.53±0.46	91.67±0.30	**60.33±0.30**	**57.71± 0.84**	51.57±0.31	**49.89±0.54**
	AT-TE$_{\text{BS}}$	91.58±0.32	90.43±0.46	50.94±0.16	44.43±0.66	45.37±0.29	39.43±0.70
	MLCAT-WP+TE$_{\text{BS}}$	**92.43±0.46**	**92.53±0.35**	60.07±0.31	57.61± 0.94	51.27±0.49	49.42±0.54

5 Experimental Evaluation

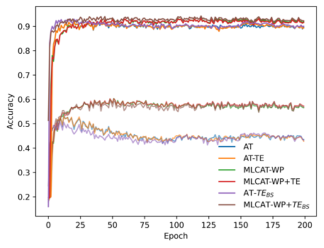
We train a ResNet-18 model using AT, AT-TE, MLCAT-WP, MLCAT-WP+TE, AT-TE$_{\text{BS}}$ and MLCAT-WP+TE$_{\text{BS}}$ for CIFAR-10 [21], CIFAR-100 [21] and SVHN [23]; see Appendix A for details. We use PGD_{10} attack ($\epsilon = 8/255$, L_∞ norm) during training and PGD_{20} at inference. In addition, we also run AutoAttack (AA) [24] which is an ensemble of different attacks for a more reliable evaluation. Table 1 shows that both AT-TE$_{\text{BS}}$ and MLCAT-WP+TE$_{\text{BS}}$ attain significant increase in clean accuracy over their counterparts AT-TE and MLCAT-WP/MLCAT-WP+TE ($\approx$2%-3% in CIFAR-10, 2%-5% in CIFAR-100, and 1%-2% in SVHN). AT-TE and AT-TE$_{\text{BS}}$ do not prevent overfitting in the SVHN dataset because the training data fits very early in training with high confidence, whereas temporal ensembling activates closer to first learning rate decay and thus, regularization based on ensemble prediction is ineffective. On the other hand, weight perturbation based approaches MLCAT-WP and MLCAT-WP+TE$_{\text{BS}}$ result in superior performance across all datasets compared to AT-TE and AT-TE$_{\text{BS}}$ in terms of clean and robust accuracy; and use of adversarial boundary examples significantly boosts the clean accuracy especially for CIFAR-100 and SVHN datasets in MLCAT-WP+TE$_{\text{BS}}$. From Figure 2, we observe that for all three datasets, AT-TE$_{\text{BS}}$ and MLCAT-WP+TE$_{\text{BS}}$approximately retain the robust accuracy of AT-TE and MLCAT-WP, but increase the clean accuracy to match and even surpass AT in case of the CIFAR-100 and SVHN datasets.



(a) CIFAR-10 Accuracy



(b) CIFAR-100



(c) SVHN

Figure 2: Accuracy results for AT using ResNet-18. Clean test accuracy (dark solid lines) and PGD_{20} attack test accuracy (dim solid lines) are plotted.

6 Conclusion

We investigate temporal ensembling and weight perturbation for mitigating robust overfitting and discover that temporal ensembling mainly influences high confidence predictions whereas weight perturbation affects both confidence in predictions and small loss data samples. Overall, adversarial weight perturbation, which directly prevents the model from fitting low loss data samples, achieves better clean and robust accuracy compared to temporal ensembling. Furthermore, we propose to use samples close to decision boundary for improving clean accuracy. These can be directly obtained from the adversarial examples generation process during adversarial training with minimal additional cost. Together with ensemble prediction regularization, this helps in retaining the robust accuracy of both robust overfitting mitigation approaches but significantly increases the clean accuracy.

References

[1] J. Bruna, C. Szegedy, I. Sutskever, I. Goodfellow, W. Zaremba, R. Fergus, and D. Erhan, "Intriguing properties of neural networks," in *International Conference on Learning Representations*, 2014.

[2] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations*, 2015.

[3] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 39–57, IEEE, 2017.

[4] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *International Conference on Learning Representations*, 2018.

[5] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan, "Theoretically principled trade-off between robustness and accuracy," in *International Conference on Machine Learning*, pp. 7472–7482, 2019.

[6] A. Shafahi, M. Najibi, M. A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein, "Adversarial training for free!," in *Advances in Neural Information Processing Systems*, pp. 3353–3364, 2019.

[7] C. Qin, J. Martens, S. Gowal, D. Krishnan, K. Dvijotham, A. Fawzi, S. De, R. Stanforth, and P. Kohli, "Adversarial robustness through local linearization," in *Advances in Neural Information Processing Systems* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, Curran Associates, Inc., 2019.

[8] V. Sehwag, S. Wang, P. Mittal, and S. Jana, "Hydra: Pruning adversarially robust neural networks," in *Advances in Neural Information Processing Systems* (H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, eds.), vol. 33, pp. 19655–19666, Curran Associates, Inc., 2020.

[9] D. Wu, S.-T. Xia, and Y. Wang, "Adversarial weight perturbation helps robust generalization," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[10] S. Gowal, C. Qin, J. Uesato, T. Mann, and P. Kohli, "Uncovering the limits of adversarial training against norm-bounded adversarial examples," *arXiv preprint arXiv:2010.03593*, 2020.

[11] T. Chen, Z. Zhang, S. Liu, S. Chang, and Z. Wang, "Robust overfitting may be mitigated by properly learned smoothening," in *International Conference on Learning Representations*, vol. 1, 2021.

[12] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry, "Robustness may be at odds with accuracy," in *International Conference on Learning Representations*, 2019.

[13] Y.-Y. Yang, C. Rashtchian, H. Zhang, R. Salakhutdinov, and K. Chaudhuri, "A closer look at accuracy vs. robustness," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[14] Y. Carmon, A. Raghunathan, L. Schmidt, J. C. Duchi, and P. S. Liang, "Unlabeled data improves adversarial robustness," in *Advances in Neural Information Processing Systems*, pp. 11190–11201, 2019.

[15] L. Rice, E. Wong, and Z. Kolter, "Overfitting in adversarially robust deep learning," in *International Conference on Machine Learning*, pp. 8093–8104, PMLR, 2020.

[16] Y. Dong, K. Xu, X. Yang, T. Pang, Z. Deng, H. Su, and J. Zhu, "Exploring memorization in adversarial training," in *International Conference on Learning Representations*, 2022.

[17] C. Yu, B. Han, L. Shen, J. Yu, C. Gong, M. Gong, and T. Liu, "Understanding robust overfitting of adversarial training and beyond," in *International Conference on Machine Learning*, pp. 25595–25610, PMLR, 2022.

[18] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2574–2582, 2016.

[19] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh, "Ead: elastic-net attacks to deep neural networks via adversarial examples," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[20] C. Laidlaw and S. Feizi, "Functional adversarial attacks," in *NeurIPS*, 2019.

[21] A. Krizhevsky, “Learning multiple layers of features from tiny images,” tech. rep., University of Toronto, 2009.

[22] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[23] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, “Reading digits in natural images with unsupervised feature learning,” 2011.

[24] F. Croce and M. Hein, “Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks,” in *International conference on machine learning*, pp. 2206–2216, PMLR, 2020.

A Experimental Setup

For all experiments, we use ResNet-18 models [22] which are trained using SGD with momentum value of 0.9 and weight decay of 5×10^{-4}. Initial learning rate is set to 0.1 for CIFAR-10 and CIFAR-100 datasets and 0.01 for SVHN dataset, which is divided by 10 at the 100^{th} and 150^{th} epochs with a total training of 200 epochs. Data augmentation consisting of horizontal flip and random crop is used for CIFAR-10 and CIFAR-100 datasets, while no data augmentation is used for SVHN datatset.

For PGD attack parameters for training, we set the $\epsilon = \frac{8}{255}$ in L_∞ norm (maximum perturbation) and 10 attack steps for all datasets where step size of $\frac{2}{255}$ is used for CIFAR-10 and CIFAR-100 datatsets and step size of $\frac{1}{255}$ for SVHN. For evaluation we consider PGD attack with 20 steps with $\epsilon = \frac{8}{255}$ in L_∞ norm and step size of $\frac{2}{255}$ for all datasets.

For temporal ensembling based approaches, the value of temporal ensembling weight parameter w is adjusted experimentally and is set to 300 in AT-TE, AT-TE$_{\text{BS}}$ and AT-TE$_{\text{CS}}$ for CIFAR-10 and SVHN datasets and 3000 for CIFAR-100 datatset along a Gaussian ramp-up curve [16]. Similarly, for MLCAT-WP+TE, MLCAT-WP+TE$_{\text{BS}}$ and MLCAT-WP+TE$_{\text{CS}}$ $w = 30$ is used in training with CIFAR-10 and SVHN datasets and $w = 300$ is used for CIFAR-100 dataset along a Gaussian ramp-up curve. The momentum term η in ensemble prediction update is set to 0.9 and temporal ensembling activates at 90^{th} epoch [16] for all experiments involving temporal ensembling. For all experiments with MLCAT-WP, MLCAT-WP+TE, MLCAT-WP+TE$_{\text{BS}}$ and MLCAT-WP+TE$_{\text{CS}}$, $L_{min} = 1.5$ for CIFAR-10 and SVHN datasets and $L_{min} = 4.0$ for CIFAR-100 dataset and other parameters are set as per the original work [17].

Furthermore all experiments are run on a single NVIDIA A100 Tensor Core GPU using PyTorch version 1.11.0 on Red Hat Enterprise Linux release 8.5 operating system.

B Additional Results for CIFAR-10, CIFAR-100 and SVHN Datasets

This section contains additional results for CIFAR-10, CIFAR-100 and SVHN datasets. We consider the case when instead of boundary sample $\mathbf{x}'$ we use clean input sample $\mathbf{x}$ and a regularization on network's current prediction on this clean sample $p(\mathbf{x})$ and ensemble prediction $z(\mathbf{x})$. Thus AT-TE$_{\text{BS}}$ modifies to

$$\min_{\boldsymbol{\theta}}\{\mathcal{L}(F(\mathbf{x},y),y) + w \cdot \|p(\mathbf{x}) - \hat{z}(\mathbf{x})\|_2^2 \ + \max_{\tilde{\mathbf{x}} \in \mathcal{B}_\epsilon(\mathbf{x})}\{\mathcal{L}(F(\tilde{\mathbf{x}},y),y) + w \cdot \|p(\tilde{\mathbf{x}}) - \hat{z}(\mathbf{x})\|_2^2\}\} \quad (6)$$

which we denote as AT-TE$_{\text{CS}}$. Similarly, MLCAT-WP+TE$_{\text{BS}}$ objective becomes MLCAT-WP+TE$_{\text{CS}}$ by including TE and clean input sample as

$$\min_{\boldsymbol{\theta}}\{\mathcal{L}(F_{\theta+v}(\mathbf{x},y),y)+w\cdot\|p(\mathbf{x})-\hat{z}(\mathbf{x})\|_2^2 + \max_{\mathbf{v}\in\mathcal{V}}\max_{\tilde{\mathbf{x}} \in \mathcal{B}_\epsilon(\mathbf{x})}\{\mathcal{L}(F_{\theta+v}(\tilde{\mathbf{x}},y),y)+w\cdot\|p(\tilde{\mathbf{x}})-\hat{z}(\mathbf{x})\|_2^2\}\} \quad (7)$$

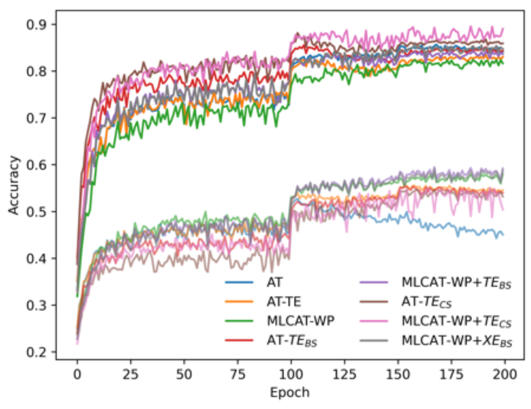
We also consider the case when MLCAT-WP objective is modified to include cross entropy loss on boundary sample $\mathbf{x}'$ but no temporal ensembling is employed. We denote this scheme as MLCAT-WP+XE$_{\text{BS}}$ which is given by

$$\min_{\boldsymbol{\theta}}\{\mathcal{L}(F_{\theta+v}(\mathbf{x}',y),y) \ + \ \max_{\mathbf{v}\in\mathcal{V}}\max_{\tilde{\mathbf{x}} \in \mathcal{B}_\epsilon(\mathbf{x})}\{\mathcal{L}(F_{\theta+v}(\tilde{\mathbf{x}},y),y)\}\} \quad (8)$$

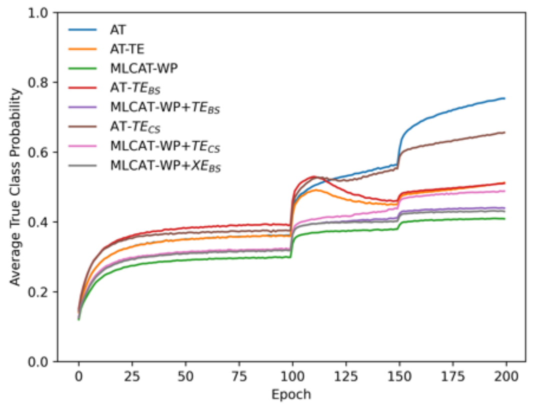
Table 2 shows the inherent trade off between robust accuracy and clean accuracy when clean input sample is used in place of boundary sample. Adversarial weight perturbation based approaches MLCAT-WP+TE$_{\text{BS}}$ and MLCAT-WP+TE$_{\text{CS}}$ seem to be more sensitive to the choice of sample as boundary sample clearly leads to significantly higher robust accuracy whereas clean input sample leads to significantly higher clean accuracy. In addition, robust overfitting occurs more severely for MLCAT-WP+TE$_{\text{CS}}$ compared to MLCAT-WP+TE$_{\text{BS}}$. MLCAT-WP+XE$_{\text{BS}}$ that does not employ temporal ensembling attains clean and robust accuracy that lie in between MLCAT-WP+TE$_{\text{BS}}$ and MLCAT-WP+TE$_{\text{CS}}$. On the other hand, AT-TE$_{\text{CS}}$ with clean input sample has increased clean accuracy and comparable robust accuracy to AT-TE$_{\text{BS}}$ for CIFAR-10 and CIFAR-100 datasets. Both AT-TE$_{\text{BS}}$ and AT-TE$_{\text{CS}}$ suffer from robust overfitting for SVHN dataset training and the reason is that the networks learn data in few epochs and start to assign high confidence predictions to input samples as shown in Figure 5(b). Since regularization based on network current prediction and ensemble prediction activates at a later stage close to first learning rate decay at epoch 100, it becomes ineffective as the ensemble prediction is also very high similar to current prediction. Figure 3-Figure 5 show how accuracy, average TCP and proportion of small loss samples evolve with time for CIFAR-10, CIFAR-100 and SVHN datasets, respectively.

Table 2: Test accuracy (mean and standard deviation of 5 runs). “Last" and “Best" refer to test accuracy at the end of training, and end of epoch that gives the highest accuracy w.r.t. test data respectively.

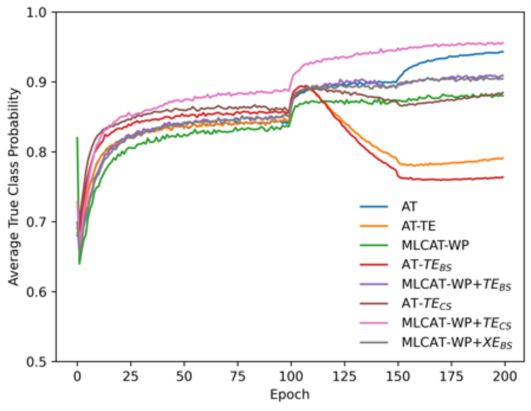
Dataset	Name	Clean(Best)	Clean(Last)	Robust(Best)	Robust(Last)	AA(Best)	AA(Last)
	AT-TE$_{BS}$	83.99±0.14	84.46±0.36	55.26±0.07	53.58±0.74	50.17±0.09	48.80±0.70
	AT-TE$_{CS}$	85.62±0.23	85.76±0.18	54.83±0.23	53.21±0.91	50.09±0.21	48.81±0.61
CIFAR-10	MLCAT-WP+TE$_{BS}$	83.712±0.51	83.75±0.41	**59.20±0.21**	**58.53±0.52**	**50.38±0.07**	**50.30±0.26**
	MLCAT-WP+TE$_{CS}$	**86.17±0.37**	**88.47±0.67**	55.37±0.28	51.68±1.18	48.13±0.40	46.84±0.23
	MLCAT-WP+XE$_{BS}$	84.81±0.3	84.91±0.33	58.76±0.16	57.8±0.61	50.09±0.09	49.75±0.32
	AT-TE$_{BS}$	58.78±0.23	58.82±0.29	31.59±0.09	30.40±0.16	25.99±0.18	24.98±0.23
	AT-TE$_{CS}$	60.71±0.37	60.52±0.41	31.08±0.22	29.72±0.19	25.72±0.23	24.77±0.15
CIFAR-100	MLCAT-WP+TE$_{BS}$	62.50±0.28	62.45±0.42	**31.95±0.06**	**30.72±0.65**	**26.57±0.14**	**25.74±0.48**
	MLCAT-WP+TE$_{CS}$	**66.38±0.82**	**67.25±0.38**	28.87±0.25	26.96±0.53	24.00±0.26	22.59±0.62
	MLCAT-WP+XE$_{BS}$	62.67±0.3	63.1±0.37	31.19±0.15	29.95±0.51	26.29±0.14	25.46±0.49
	AT-TE$_{BS}$	91.58±0.32	90.43±0.46	50.94±0.16	44.43±0.66	45.37±0.29	39.43±0.70
	AT-TE$_{CS}$	91.75±0.69	90.24±0.43	48.72±0.45	43.02±0.79	42.42±0.27	38.22±0.68
SVHN	MLCAT-WP+TE$_{BS}$	92.43±0.46	92.53±0.35	**60.07±0.31**	**57.61± 0.94**	51.27±0.49	**49.42±0.54**
	MLCAT-WP+TE$_{CS}$	**93.18±0.69**	**93.57±0.41**	56.22±1.17	52.05± 1.78	48.71±1.02	45.19±1.01
	MLCAT-WP+XE$_{BS}$	92.47±0.44	93.4±0.82	59.78±0.04	55.21± 1.2	**51.29±0.24**	46.73±1.81



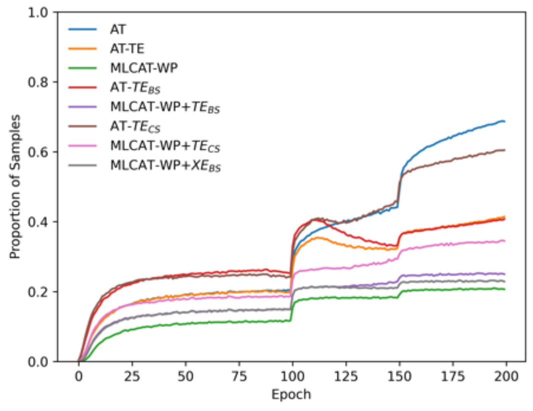
(a) Test Data Accuracy



(b) TCP

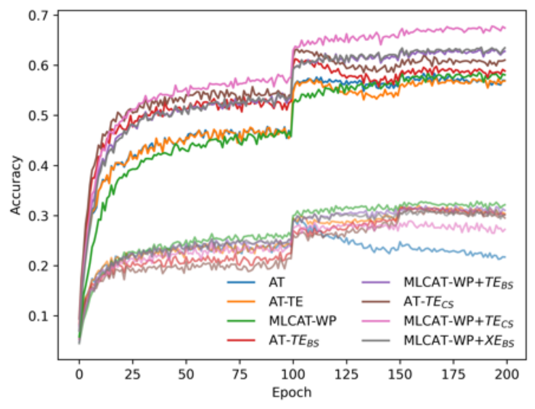


(c) TCP, loss $\in$ [0, 0.5)

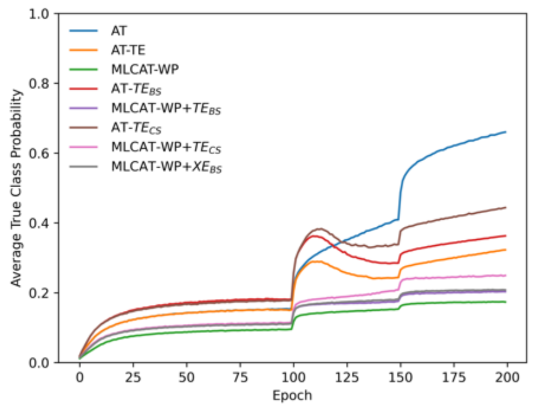


(d) Sample Proportion, loss $\in$ [0, 0.5)

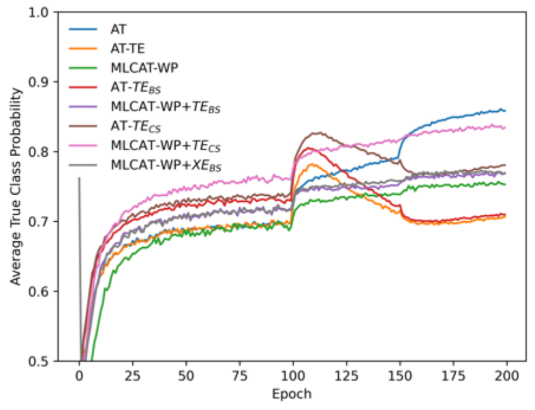
Figure 3: CIFAR-10 training for ResNet-18. (a) Test accuracy against clean data (dark solid lines) and PGD_{20} attack (dim solid lines) are plotted.



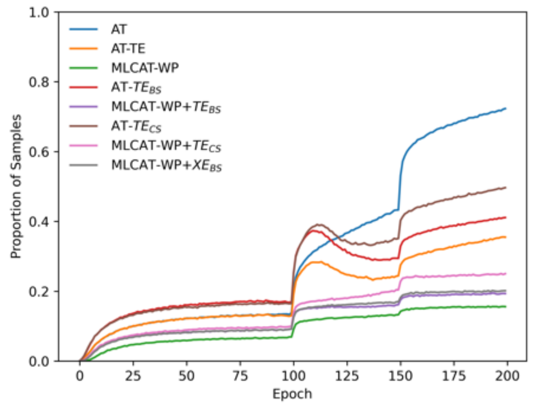
(a) Test Data Accuracy



(b) TCP

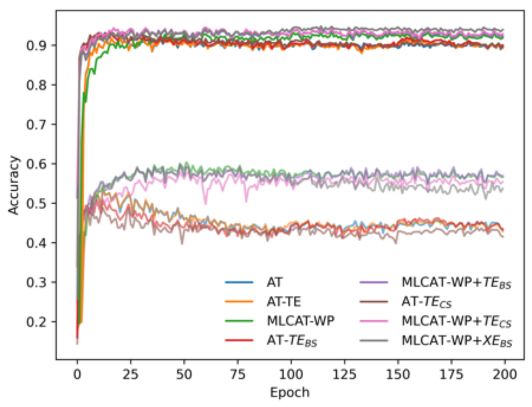


(c) TCP, loss $\in [0, 1.0)$

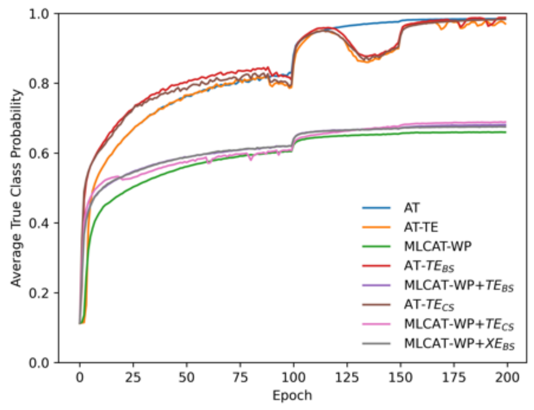


(d) Sample Proportion, loss $\in [0, 1.0)$

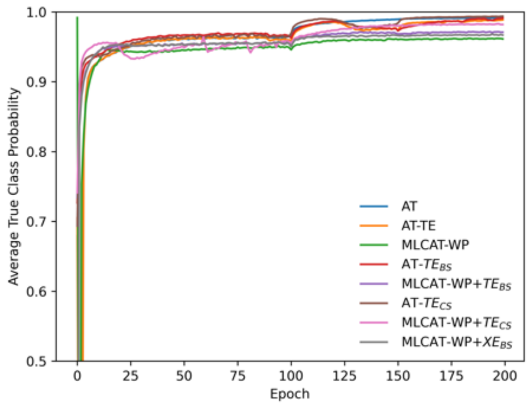
Figure 4: CIFAR-100 training for ResNet-18. (a) Test accuracy against clean data (dark solid lines) and PGD_{20} attack (dim solid lines) are plotted.



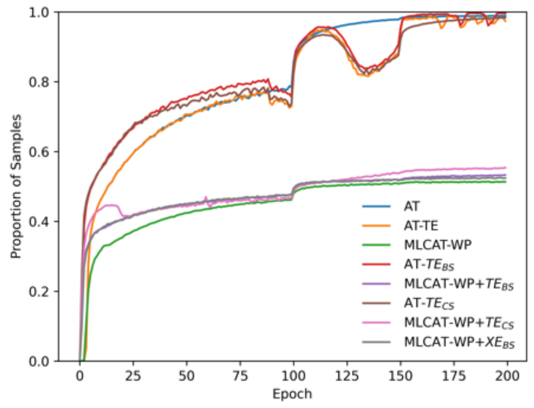
(a) Test Data Accuracy



(b) TCP



(c) TCP, loss $\in [0, 0.5)$



(d) Sample Proportion, loss $\in [0, 0.5)$

Figure 5: SVHN training for ResNet-18. (a) Test accuracy against clean data (dark solid lines) and PGD_{20} attack (dim solid lines) are plotted.