PeerAiD: Improving Adversarial Distillation from a Specialized Peer Tutor

Jaewon Jung, Hongsun Jang, Jaeyong Song, and Jinho Lee*
Department of Electrical and Computer Engineering, Seoul National University, Seoul, Korea
{jungjaewon, hongsun.jang, jaeyong.song, leejinho}@snu.ac.kr

Abstract

Adversarial robustness of the neural network is a significant concern when it is applied to security-critical domains. In this situation, adversarial distillation is a promising option which aims to distill the robustness of the teacher network to improve the robustness of a small student network. Previous works pretrain the teacher network to make it robust against the adversarial examples aimed at itself. However, the adversarial examples are dependent on the parameters of the target network. The fixed teacher network inevitably degrades its robustness against the unseen transferred adversarial examples which target the parameters of the student network in the adversarial distillation process. We propose PeerAiD to make a peer network learn the adversarial examples of the student network instead of adversarial examples aimed at itself. PeerAiD is an adversarial distillation that trains the peer network and the student network simultaneously in order to specialize the peer network for defending the student network. We observe that such peer networks surpass the robustness of the pretrained robust teacher model against adversarial examples aimed at the student network. With this peer network and adversarial distillation, PeerAiD achieves significantly higher robustness of the student network with AutoAttack (AA) accuracy by up to 1.66%p and improves the natural accuracy of the student network by up to 4.72%p with ResNet-18 on TinyImageNet dataset. Code is available at https://github.com/jaewonalive/PeerAiD.

1. Introduction

Deep learning is undoubtedly an irreplaceable tool in many domains such as images [11, 17, 27], natural language processing [5, 9, 35], voice recognition [4, 15], and many real-life scenarios. However, it has been found that DNNs are vulnerable to an imperceptible noise crafted by attackers [14, 28], and this severely raises concerns about deploying DNNs in critical domains, e.g., autonomous driving [29] and healthcare [44]. Though [28] found that a large model has better robustness than a smaller neural network, a large model is not always applicable to all circumstances, especially in an edge device which has a small size of memory and limited computational capability. Currently, the only defense technique known to be effective is a variant of adversarial training [14, 28], which essentially trains a model from attacked samples to gain robustness. Even though the training samples are adversarially perturbed, guiding the training with the correct label allows the model to learn features that are not fooled by similar perturbations.

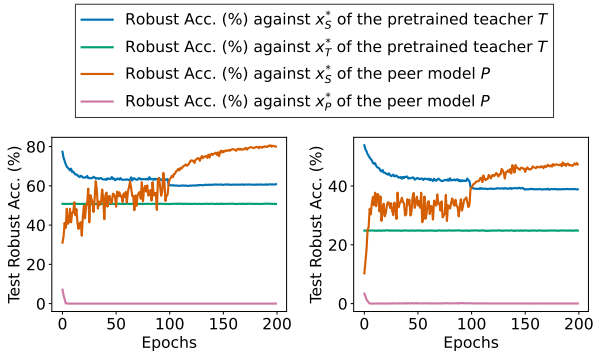


Figure 1. Test robust accuracy of a pretrained robust teacher and a peer network trained from scratch. x_S^*, x_T^* and x_P^* denotes the adversarial examples generated from the student, teacher, and peer model, respectively. Results on CIFAR-10 (left) and CIFAR-100 (right) show that the peer sustains increasing robustness for the student not provided by the pretrained teacher.

Among the adversarial training family, one promising and popular way to enhance robustness of the small student network is adversarial distillation (AD), which uses a robustly pre-trained teacher to guide the student. In an extenstion of knowledge distillation [18] which uses the pretrained teacher to approximate the label distribution of the data, many AD methods [19, 45, 46] use the pretrained robust teacher network which approximates the label distribution of adversarial examples aimed at the student network.

For such AD methods, an underlying assumption is that the robustness contained within the pretrained teacher is maintained along the adversarial distillation process. In other words, we expect the teacher to provide a defense

*Corresponding author

for the adversarial examples produced by the student model during adversarial training. However, such an assumption does not hold in adversarial distillation, especially as the student is trained for several epochs toward convergence.

In Fig. 1, we test if such an assumption holds in actual training. Over 200 epochs of AD training, we attack the student network to create perturbed samples using Projected Gradient Descent (PGD) [28]. With the blue curves, we plot the robust accuracy the teacher achieves against those samples (attacked with students). Initially, the teacher provides a good defense against the student-attacked samples, whose prediction could be a reliable guide to the student [45]. However, the robustness quickly drops as the student is trained, and further reduces at the learning rate step. In fact, similar phenomena have been reported in other literature [45]. IAD [45] partially trusts the teacher network depending on its reliability. AKD^2 [7] uses the naturally trained teacher together with the pre-trained robust teacher. MTARD [43] also employs both teachers, balancing their influence based on how much students converge towards each teacher. However, while these approaches are effective to some degree, they are limited in that they do not improve the guidance of the teacher, but only reduce the effect of some bad guidance.

In such a regard, the orange curves in Fig. 1 show an intriguing observation. We use the same teacher model but is randomly initialized and trained from scratch as a *peer* against the student-attacked samples. While it is within the expectation that the robust accuracy goes up, the robust accuracy against transferred adversarial examples x_S^* from the student network reaches much higher than that of the robustly pretrained teacher. However, the peer has almost no defense (close to 0%) against adversarial samples x_P^* that attack itself (i.e., peer-attacked samples). This states that the peer is specialized at defending against attacks on a student, instead of being a general robust model.

From these, we propose PeerAiD, a new AD method that achieves much higher adversarial robustness from training a peer tutor of the target student model. The method contains the structure to train a peer model for AD, in addition to a novel loss function to train the student to take better guidance from the peer. Our contributions are summarized as follows:

- We observe that training a peer model from the student-attacked sample can build a peer tutor with better guidance for adversarial distillation.
- We propose PeerAiD that trains a peer using adversarial examples aimed at the student and uses it for AD.
- We propose a loss function that is suitable for peer-tutored adversarial distillation.
- An extensive set of experiments show that PeerAiD gains significantly higher robust accuracy over the prior art in several models and datasets.

2. Related Work

2.1. Adversarial Training

Adversarial Training (AT) [14, 28] is an effective method that defends against many white-box and black-box attacks [8, 14, 28]. Adversarial training is a robust optimization problem and consists of inner maximization and outer minimization. Mathematically, adversarial training can be formulated as follows.

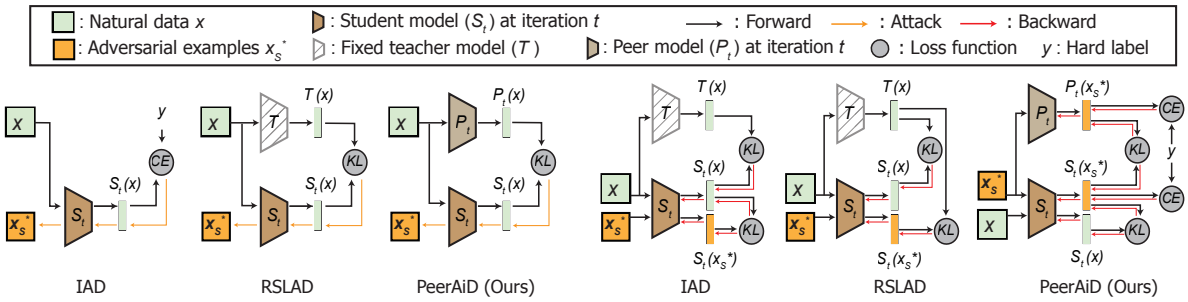
$$\min_{\theta} L_{min}(f(\theta, x_i^*), y_i) \tag{1}$$

$$\text{where } x_i^* = \arg\max_{\tilde{x}_i \in B(x_i,\epsilon)} L_{max}(f(\theta, \tilde{x}_i), y_i)$$

$\{(x_i, y_i)\}_{i=1}^N$ is a training dataset with N samples of input x_i and label y_i. f is a neural network with the parameter θ. L_{max} is a loss which is used to find the adversarial examples which increases the loss of the neural network. The parameter of the neural network is optimized by the L_{min} toward the direction which reduces this L_{min}. $B(x_i, \epsilon)$ is a ball which restricts the distance between the adversarial example $\tilde{x}_i$ and the original data x_i. Popularly used constraint involves l^∞ norm and $B(x_i, \epsilon) = \{\tilde{x}_i \mid \|x_i - \tilde{x}_i\|_\infty \leq \epsilon\}$ is widely used in adversarial training. FGSM [14] proposes a single-step attack that uses only one iteration of getting gradients of inputs and taking the sign of the gradients to solve the inner maximization. PGD [28] was developed to iteratively solve the inner maximization problem. After that, many works [41, 46] adopted Kullback-Leibler (KL) divergence loss in inner maximization to find a better solution for the inner maximization problem to improve the quality of adversarial examples. The outer minimization problem is to find model parameters that reduce classification loss (e.g., Cross Entropy) given adversarial examples. The model parameters solved by outer minimization in adversarial training give higher robust accuracy than the model parameters found by standard training.

2.2. Adversarial Distillation

Many researchers studied adversarial distillation to transfer the robustness of a large teacher network to the student network. ARD [13] adopts the idea of standard knowledge distillation. It used the prediction of the robust teacher network on natural data to guide the student network. AKD^2[7] show that adversarial distillation and weight averaging [20] could prevent robust overfitting problem [1, 10]. IAD [45] focuses on the reliability of teacher networks during adversarial distillation. It claims that adversarial samples generated from a student network become challenging in later epochs, so this makes the student network more trustable itself, while the teacher becomes more unreliable on adversarial training data generated from the student network. RSLAD [46] finds that the inner maximization process of



(a) Inner maximization (b) Outer minimization

Figure 2. The adversarial distillation procedure overview of baselines [45, 46] and PeerAiD. (a) describes inner maximization to generate adversarial examples from the student model. Baselines use hard labels or pretrained teachers for this. On the other hand, PeerAiD uses a peer model. (b) illustrates the outer minimization procedure to optimize model parameters. Baselines only train the student models with the prediction of the pre-trained teachers, but PeerAiD trains both peer and student models simultaneously.

generating adversarial samples could be improved by replacing the hard label with the soft label produced by a robust teacher network on the natural data. AdaAD [19] presents that maximizing prediction discrepancy between a student network and a robust teacher network could improve the inner maximization process of adversarial training. CAT [25] proposes to simultaneously train multiple robust student networks and exchange their adversarial examples among themselves. CAT aligns with online knowledge distillation [16, 30, 39, 42] which trains multiple student networks simultenously in standard training.

3. Proposed Method

3.1. Preliminary

Adversarial distillation usually pretrains the robust teacher with the adversarial training following Eq. (1). The pretrained teacher $T(\cdot)$ is used to produce soft label with natural data x_i or adversarial examples $x^*_{i,S}$ of a student model $S(\cdot)$. In adversarial distillation, $T(x_i)$ and $T(x^*_{i,S})$ replace the hard label y_i in the outer minimization of Eq. (1). The KL divergence loss is popularly chosen as L_{min}. RSLAD [46] found that replacing the hard label y_i with the soft label produced by the pretrained robust teacher in the inner maximization improves the robust accuracy in adversarial training. It regards a prediction of the pretrained robust teacher on natural data as a fixed target and chose $KL(T(x_i)\|S(\tilde{x}_i))$ as L_{max}.

3.2. Peer Tutoring

We suggest *peer tutoring* to use online knowledge distillation in an adversarial distillation setting. Peer tutoring trains the peer model P with the student model S from scratch while making the peer model robust to the adversarial examples aimed at the student model. The peer model provides reliable guidance with peer tutoring because it becomes much more robust to the adversarial examples generated from the student model.

Inner maximization. As illustrated in Fig. 2(a), previous approaches for inner maximization can be categorized into two ways. The former [7, 45] uses the hard label y only. The latter [19, 46] uses the prediction of the adversarily pre-trained teacher to generate adversarial examples of the student network. However, each approach has an obvious limitation. First, using only hard labels loses the chance to learn the probability distribution of non-target classes from other networks. Second, the fixed pre-trained robust teacher network has a limitation on the natural accuracy, which indicates how well the teacher network approximates the true label distribution of the natural samples x. This trade-off between robustness and the natural accuracy is theoretically studied [36, 41] and empirically observed phenomenon [31] in many works.

Instead, PeerAiD uses the prediction of a peer network which interactively learns with the student network to generate adversarial examples x^*_S of the student network S as depicted in Fig. 2(a). The adversarial examples x^*_S are generated with PGD [28] by finding the gradients of input which increases the KL divergence between the prediction of the peer model and the student model on the training data. With peer tutoring, the peer model provides the approximated label distribution to the student model. The label distribution provides information on non-target classes which is not contained in the hard label. The peer model also does not suffer from the degradation in the natural accuracy, which will be discussed in Sec. 4.4.

Outer minimization. Previous approaches [19, 45, 46] used the prediction of pre-trained teachers to distill better robustness. However, the teacher models are adversarially pretrained with adversarial examples aimed at themselves. The teacher model has never seen adversarial examples of the student network during the pretraining process.

On the other hand, PeerAiD trains the peer network and the student network simultaneously with the same adversarial examples x^*_S generated from the student network as illus-

trated in Fig. 2(b). The main reason PeerAiD uses the adversarial examples x_S^* generated from a student network is to make peer network robust to the adversarial examples of the student network. Though many previous works focused on the transferability of the adversarial examples among neural networks, we found that it is not necessarily true that adversarial examples have large similarities and small distances. Therefore, we can build the peer network which is specialized in being robust to the adversarial examples of the student network while not being robust to the adversarial examples of itself. We further describe this in Sec. 4.4. Note that the ultimate goal of our novel outer minimization process is to build a robust student network and not a robust peer network. Surprisingly, we find that the peer network is not robust at all to the adversarial examples aimed at itself while being robust to the adversarial examples of the student network.

Loss function. To provide the soft label in inner maximization, we use the prediction of the peer model on the natural samples. KL divergence loss is used to find the adversarial examples maximizing the discrepancy between the prediction of the peer model on natural images and the prediction of the student model on the adversarial examples :

$$L_{max} = KL(P_t(x) \| S_t(\tilde{x})) \tag{2}$$

where $P(\cdot)$ and $S(\cdot)$ denote the prediction output of a peer model and a student model, respectively. We use the subscript t to denote the training iteration and highlight the peer network is not fixed in the process of adversarial distillation compared to other baselines. $\tilde{x}$ is the adversarial examples which should satisfy the constraint on the magnitude of the perturbation as described in Eq. (1).

In outer minimization, the loss of the peer model consists of Cross Entropy (CE) loss and KL divergence loss.

$$L_{peer} = \gamma_1 * H(y, P_t(x_S^*)) + \gamma_2 * \tau^2 * KL(S_t^\tau(x_S^*) \| P_t^\tau(x_S^*)) \tag{3}$$

where τ is a temperature parameter that smooths the output of a softmax layer. The key aspect is that the peer is trained using samples adversarial to the student (x_S^*). The cross-entropy (CE) loss H is intended to make the peer model learn the label distribution of adversarial examples x_S^* with the hard label, which provides the consistent guidance. The KL divergence loss of a peer model is to distill the knowledge of a student model, which provides the learned probability distribution of non-target classes by the student model.

The loss of the student model also consists of the cross entropy loss and KL divergence loss. However, it also has an additional regularization term which prevents the large discrepancy between the prediction on natural images and adversarial examples [41]. The soft label provided by the peer model is treated as the constant soft target in the loss of the student model.

$$\begin{aligned} L_{student} = {} & \lambda_1 * H(y, S_t(x_s^*)) \\ & + \lambda_2 * \tau^2 * KL(P_t^\tau(x_s^*) \| S_t^\tau(x_s^*)) \\ & + \lambda_3 * \tau^2 * KL(S_t^\tau(x) \| S_t^\tau(x_s^*)) \end{aligned} \tag{4}$$

Then, L_{min} is the sum of L_{peer} and $L_{student}$. The parameters of the two models are optimized simultaneously.

$$L_{min} = L_{peer} + L_{student} \tag{5}$$

4. Experimental Results

4.1. Experiment Settings

CIFAR-10 and CIFAR-100 [22]. For all the baselines [7, 19, 25, 28, 41, 45, 46] results, we trained the baselines following their original settings. For PeerAiD results, we followed the training setting of [46]. In detail, we trained PeerAiD for 300 epochs and the training batch size is 128. The learning rate is $1e-1$ and it decays at epochs of 215, 260, and 285 by a factor of 10. The weight decay is $2e-4$. We applied weight averaging to PeerAiD and AKD^2 following [7] for a fair comparison. The detailed hyperparameters can be found in the supplementary materials.

TinyImageNet [23]. We follow the hyperparameters of [7, 45] for the baselines. For PeerAiD, we trained ResNet-18 [17] with 200 epochs and WideResNet34-10 [40] with 100 epochs. The total batch size is 128. We use the SGD optimizer with 0.9 momentum and the weight decay $2e-4$. The initial learning rate is set to $1e-1$ and decays at epoch 100 and 150 for ResNet-18, and at epoch 50 and 80 for WideResNet34-10 by a factor of 10. We applied weight averaging to PeerAiD and AKD^2 following [7] for a fair comparison. For more details, please refer to the supplementary materials.

Evaluation metrics. We tested the white-box robustness of the baselines and PeerAiD. We report the best robust accuracy validated by PGD-10 [28] with a step size of 2/255 and a perturbation budget of $\epsilon = 8/255$. Only the non-robust model, which is denoted by *Natural* was chosen by the best natural accuracy because it is not robust at all in the course of training. PGD-20 attack was conducted with a step size of 2/255 and a perturbation budget of $\epsilon = 8/255$. FGSM [14] attack also used the same perturbation budget as PGD. However, these attacks are not perfect for checking the robustness of a model because it is vulnerable to gradient obfuscation [3]. AutoAttack (AA) [8] is prevalently regarded as the strongest attack. It includes targeted, untargeted PGD attacks and black-box score-based attacks [2].

Teacher models of baselines. In adversarial distillation, the performance of the student model also depends on the teacher model. The larger models usually show better robustness than the smaller ones [28], so we used the

Dataset	Method	ResNet-18				WideResNet34-10			
		Clean	FGSM	PGD-20	AA	Clean	FGSM	PGD-20	AA
CIFAR-10	Natural	**95.42**	35.74	0.00	0.00	**96.08**	45.92	0.00	0.00
	PGD-AT	84.21	56.93	49.71	46.79	86.27	57.69	49.94	48.07
	TRADES	81.47	57.75	52.92	49.35	84.48	60.07	54.33	51.88
	AKD^2	83.99	59.52	53.72	50.12	87.83	**64.14**	56.68	54.25
	RSLAD	81.00	58.65	**54.40**	51.03	83.80	61.48	55.25	52.37
	IAD	80.63	58.13	53.43	49.88	83.51	60.91	54.33	51.89
	CAT	82.40	58.56	53.39	50.06	86.40	63.66	56.77	54.17
	AdaAD	81.41	57.45	53.51	50.08	84.49	60.65	55.98	53.38
	PeerAiD	85.01	**61.28**	54.36	**52.57**	85.64	63.40	**56.81**	**55.21**
CIFAR-100	Natural	**75.48**	8.70	0.00	0.00	**79.68**	12.58	0.04	0.00
	PGD-AT	57.30	28.47	24.15	21.84	59.57	30.17	25.73	23.99
	TRADES	54.90	30.93	28.29	23.69	55.70	32.53	30.02	26.07
	AKD^2	58.84	33.07	30.33	25.83	61.83	36.40	**33.20**	28.88
	RSLAD	55.45	33.11	30.78	25.96	57.42	33.95	30.75	27.20
	IAD	54.98	32.87	30.28	25.44	57.92	34.30	31.47	27.55
	CAT	57.81	33.94	**31.44**	25.93	61.68	36.82	32.95	28.39
	AdaAD	56.08	31.79	29.76	25.03	57.99	34.03	31.89	27.88
	PeerAiD	59.35	**34.41**	29.69	**27.33**	61.33	**37.08**	32.39	**30.06**
TinyImageNet	Natural	**64.74**	1.65	0.02	0.00	**68.81**	2.32	0.02	0.00
	PGD-AT	46.25	24.47	22.53	17.80	51.10	27.50	24.83	20.57
	TRADES	48.87	24.64	22.31	16.90	52.49	27.61	25.36	19.67
	AKD^2	50.47	27.25	25.12	20.01	54.82	31.83	29.33	24.09
	RSLAD	43.19	24.61	22.92	17.17	51.06	30.28	28.35	22.80
	IAD	47.67	26.11	23.86	18.88	45.96	27.15	25.72	20.80
	CAT	40.66	23.19	22.06	15.19	40.85	24.78	23.20	16.76
	AdaAD	49.97	25.79	23.98	18.16	52.22	28.32	26.53	21.13
	PeerAiD	55.19	**29.42**	**26.10**	**21.67**	58.07	**33.04**	**29.51**	**24.82**

Table 1. The white-box robustness under various attack methods.

same or larger teacher network than a student in the evaluation. Without mention, the same architecture of the teacher model is used for AD as a default. We adversarially trained the robust teacher model using TRADES [41] as the teacher model because it shows better robustness than PGD Adversarial Training (PGD-AT) [28]. We mainly evaluated with ResNet-18 [17] and WideResNet34-10 [40], which are representative models in the adversarial robustness community.

4.2. Adversarial Robustness Result

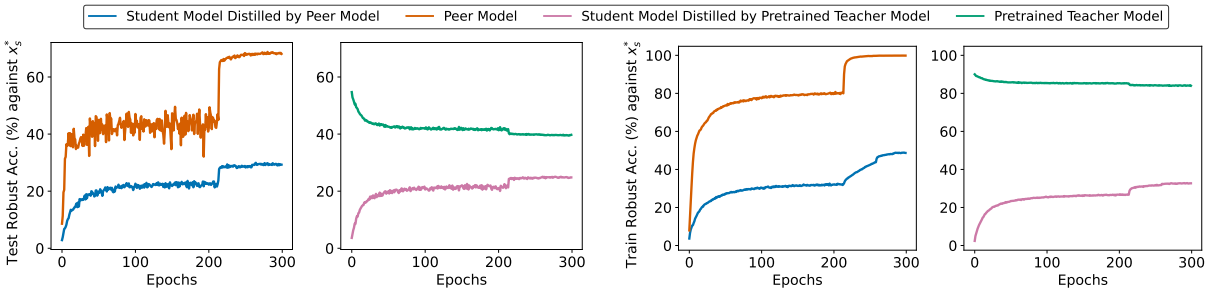
Tab. 1 reports the white-box robustness of various baselines and PeerAiD. AutoAttack [8] is the most reliable metric because many gradient-based attacks (FGSM and PGD attacks) are vulnerable to gradient obfuscation and give a false sense of security [3]. PeerAiD shows higher AutoAttack accuracy than all the baselines from $0.73\%p$ to $1.66\%p$, and surpasses the clean accuracy of the other adversarial distillation baselines by up to $4.72\%p$. The improvement in PeerAiD is more significant with ResNet-18 compared to WideResNet34-10. This is a favorable result in adversarial distillation settings because the distillation is often conducted to improve the robustness of a small model. Overall, PeerAiD provides a much better trade-off between the clean accuracy and the robust accuracy than the other adversarial distillation baselines. The effect of PeerAiD is not limited to small-scale datasets and small models because PeerAiD also improves the result of large-scale dataset TinyImagenet with WideResNet34-10. In Sec. 4.5, we conduct an ablation on weight averaging which is applied to AKD^2 and PeerAiD by applying it to the other baselines. We include the results of the transfer-based attack and gradient obfuscation tests in Sec. 4.7 to exclude the possibility of gradient obfuscation.

4.3. Effectiveness of Peer Tutoring

In Fig. 3, we plotted the training and test robust accuracy curves of each model against the adversarial examples x_s^* generated from a student model to show the effectiveness of peer tutoring. The pretrained teacher model is adversarially trained beforehand with PGD-10. The peer model is simultaneously trained with the student model with the loss of Eq. (3). For an ablation study, we applied the same training loss to the student model with Eq. (4) regardless of whether it is distilled by the pretrained robust teacher or the peer model.

There are distinct patterns with peer teaching compared to the training with the pretrained teacher model. In Fig. 3(a), the test robust accuracy of the peer network against x_S^* (orange) jumps up at the early epoch of adversarial distillation. Then it experiences an additional jump right after the learning rate decay at epoch 215. However, the test robust accuracy of the pretrained model against x_S^* (green) keeps decreasing after the initial epoch of the training. This degradation in the robust accuracy of the pretrained teacher model is due to the increasing complexity of the adversarial examples generated from a student model as the student model becomes robust along the adversarial training [45].

The improvement that comes from peer teaching can be explained by the empirical robustness that the peer model attained during the adversarial distillation. In Fig. 3(b), the train robust accuracy of the peer model with the adversar-



(a) Test robust accuracy with student-generated adversarial examples. (b) Train robust accuracy with student-generated adversarial examples.

Figure 3. Robust accuracy against student-generated adversarial examples x_s^*. Test (a) and train (b) robust accuracy are presented. ResNet-18 is used to measure the robust accuracy with CIFAR-100.

f	$Clean_f(x)$	$Rob_f(x_f^*)$	$Rob_f(x_S^*)$	$Rob_S(x_S^*)$	$cos(f(x), f(x_S^*))$	$cos(f(x_S^*), f(x_f^*))$	$cos(f(x), f(x_f^*))$
Peer tutor (ours)	**75.63**	0.00	**69.19**	**29.69**	0.95	0.37	0.35
Pretrained robust teacher	57.30	24.15	39.46	24.48	0.96	0.97	0.96
Naturally pretrained teacher	**75.48**	0.00	43.05	14.39	0.80	0.47	0.41

Table 2. The relationship between the cosine similarities of the penultimate layer representation measured with ResNet-18 on CIFAR-100. $Rob_f(\cdot)$ denotes the robust accuracy of f. S denotes the student network. $Clean_f(x)$ means the clean accuracy of f on natural samples.

ial examples x_S^* (orange) increases because the peer model learns x_S^* directly and the weak part of the student model. After the first learning decay at epoch 215, the train robust accuracy of the student keeps increasing. However, the pretrained teacher suffers from the distribution shift between the adversarial examples it learned during adversarial training and the transferred adversarial examples x_S^* in the process of the adversarial distillation. Therefore, the training robust accuracy of the pretrained teacher model on x_S^* (green) keeps decreasing as illustrated in Fig. 3(b).

4.4. Robustness of Peer Network

In PeerAiD, the peer model plays the role of guiding student models in both the inner maximization and outer minimization process. As shown in Tab. 2, we found that this peer model is specialized in defending the attack samples x_S^* generated from the student model. This peer model has higher robust accuracy against x_S^* ($Rob_f(x_S^*)$) than the pretrained robust model. However, notably, this peer network is not robust at all, and its robust accuracy against itself with PGD-20 ($Rob_f(x_f^*)$) is 0% as illustrated in Tab. 2.

In Tab. 2, we also measured the cosine similarity between adversarial examples and natural examples in the feature space. We intend to find how the peer model can provide reliable guidance to the student model while it is not robust at all against the adversarial examples aimed at itself. The peer model shows a comparable cosine similarity between natural samples x and x_S^* in the features space compared to the pre-trained robust teacher. It embeds x_S^* around x and this is the desirable property of the robust model because the robust model is expected to make the prediction on the adversarial examples equal to the natural samples.

Method	Clean	FGSM	PGD-20	AA
Natural+SWA	**67.42**	2.22	0.04	0.00
PGD-AT+SWA	49.10	26.32	23.95	19.46
TRADES+SWA	50.22	26.10	23.79	18.47
RSLAD+SWA	42.13	24.33	22.99	17.32
IAD+SWA	48.70	26.94	24.88	19.62
CAT+SWA	38.13	22.13	20.99	14.53
AdaAD+SWA	50.41	25.81	24.20	18.43
PeerAiD w/o SWA	54.01	28.21	25.13	20.00
PeerAiD	55.19	**29.42**	**26.10**	**21.67**

Table 3. Ablation of SWA with ResNet-18 on TinyImageNet

However, the peer model has much higher natural accuracy which even slightly surpasses the naturally pretrained teacher. The peer model achieves a natural accuracy of 75.63%, significantly higher than the 57.30% achieved by the pretrained teacher model. This superior performance the peer model shows with natural examples and x_S^* implies that it is a better approximator for the label distribution of natural samples and x_S^* than the pretrained robust teacher model. The pretrained robust teacher inevitably suffers from the degradation in the natural accuracy compared to the standard training due to the trade-off between the clean accuracy and the robust accuracy [31, 36, 41], whereas the peer model does not experience this trade-off.

4.5. Ablation on Weight Averaging

We include the ablation of Stochastic Weight Averaging (SWA) to check the effectiveness of PeerAiD without SWA. In Tab. 3, PeerAiD without SWA shows the higher clean accuracy than adversarial training baselines which incorporate SWA by up to 3.6%p, while also exhibiting superior robust accuracy with ResNet-18 on TinyImageNet. We applied weight averaging to the baselines following [7], which

Method	Clean	FGSM	PGD_{20}	AA
AKD^2	57.33	32.89	30.26	25.69
RSLAD	55.39	32.21	29.29	24.49
IAD	55.51	31.44	28.54	24.41
CAT	56.61	34.06	31.16	25.50
AdaAD	56.72	31.95	29.01	24.86
PeerAiD	57.63	34.33	30.17	**26.99**

Table 4. CIFAR-100 robust accuracy of ResNet-18 with WRN34-10 teacher (peer) model.

Method	Clean	FGSM	PGD_{20}	AA
AKD^2	82.75	57.03	52.68	48.45
RSLAD	79.91	57.25	53.54	49.85
IAD	80.15	57.97	53.23	49.10
CAT	77.22	53.13	49.37	45.17
AdaAD	79.81	54.81	51.21	47.59
PeerAiD	82.41	57.43	52.00	**50.02**

Table 5. CIFAR-10 Robust accuracy of MobileNetV2.

applies SWA from the epoch of the first learning rate decay to the last epoch. SWA also increases the robust accuracy of the adversarial training baselines except for CAT. The results in Tab. 3 indicate that SWA is not an essential part of PeerAiD but complements it.

4.6. Teacher Sensitivity

Some works [7, 25, 45] assume a situation where a teacher model and a student model have the same capacity. However, other approaches [19, 46] chose a larger teacher model than a student model to measure its effectiveness. For a fair comparison, we also tested the effectiveness of PeerAiD with the large teacher (peer) model on CIFAR-100 dataset. In Tab. 4, the teacher (peer) model is WideResNet34-10, and the student model is ResNet-18.

We observed that the student model trained with PeerAiD still maintains improved performance compared to the baselines with a large teacher model. These results support the proposed method of PeerAiD is not limited to the case when the peer network has to be the same architecture. In Tab. 5, we also tested the effectiveness of PeerAiD with MobileNetV2 [33] on CIFAR-10, a widely used neural network in distillation settings. PeerAiD also shows the highest AutoAttack accuracy and better tradeoff between the robustness and the clean accuracy with MobileNetV2.

4.7. Gradient Obfuscation Test

It has been highlighted that any robust model should be secure against transfer-based attacks [3, 6, 32]. Therefore, it must be checked whether PeerAiD shows robustness against transfer-based attacks. Here, we train two surrogate models, which are ResNet-34 [17] and MobileNetV2 [33] with PGD-10 adversarial training [28]. The training setting is the same as the PGD-AT baseline in Sec. 4.1. We transferred the adversarial examples generated from these two

Surrogate	ResNet-34		MobileNetV2	
Method	FGSM	PGD_{20}	FGSM	PGD_{20}
PGD-AT	38.54	37.11	38.62	37.08
TRADES	38.84	38.02	38.16	37.34
AKD^2	40.95	39.77	38.89	37.88
RSLAD	40.12	39.29	38.85	37.82
IAD	39.63	38.81	39.16	37.97
CAT	42.38	41.66	39.46	38.42
AdaAD	39.24	38.30	38.60	37.27
PeerAiD	**44.23**	**43.61**	**42.15**	**40.73**

Table 6. Checking gradient obfuscation by measuring the robust accuracy of ResNet-18 on CIFAR-100 dataset under transfer-based attacks.

Dataset	Model	PGD-10	PGD-1K	$\epsilon = \infty$
CIFAR-10	ResNet-18	55.54	53.94	0.00
	WRN34-10	57.89	56.43	0.00
CIFAR-100	ResNet-18	30.66	29.36	0.00
	WRN34-10	33.24	31.99	0.00
TinyImageNet	ResNet-18	26.34	25.86	0.00
	WRN34-10	29.99	29.34	0.00

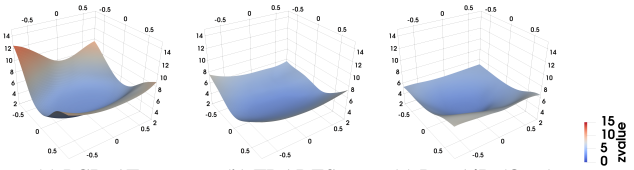
Table 7. Obfuscated gradient test results proposed in [3].

surrogate models to ResNet-18 which is trained by each method. FGSM and PGD-20 were used to create adversarial examples and the robust accuracy of models trained by each method are described in Tab. 6 It shows that PeerAiD is more robust than baselines against transfer-based attacks.

It is also known that previous works on adversarial training actually rely on the obfuscated gradient, giving a false sense of security [3]. We conducted the gradient obfuscation test mentioned in [3]. First, the robust accuracy of PeerAiD against PGD-10 is similar to the robust accuracy against PGD-1K in Tab. 7. Second, the unbounded attack on PeerAiD successfully reaches 0% robust accuracy in Tab. 7. Third, Tab. 1 shows that the success rate of a one-step attack is lower than PGD-20 with PeerAiD. It implies that the inner maximization process is not stuck in a local solution. Lastly, AutoAttack [8] includes SQUARE attack [2], a black-box score-based attack. Therefore, the above results exclude the possibility of gradient obfuscation in PeerAiD.

4.8. Loss Landscape Visualization

Prior works [38] found that a flat loss landscape is favorable to the generalization of neural networks. [12] showed that a sharp loss landscape causes a big difference between training and test distribution. Especially in the context of adversarial robustness, previous works [7] showed that a flatter loss landscape helps to mitigate robust overfitting. In this regard, many previous works about adversarial robustness [7, 19, 21, 26, 38] showed that their method makes the loss landscape flatter as expected. The loss landscape of PeerAiD also coincides with these arguments. In Fig. 4, we visualize the loss landscape in weight space. Compared to PeerAiD, the loss landscape of PGD-AT and TRADES is



(a) PGD-AT (b) TRADES (c) PeerAiD (Ours)

Figure 4. Comparison of weight loss landscape visualization [24] between baselines [28, 41] and PeerAiD. The WRN34-10 [40] model trained with CIFAR-100 by each method is perturbed along a random direction within the range of $[-0.75, 0.75]$. The vertical axis z denotes the loss value.

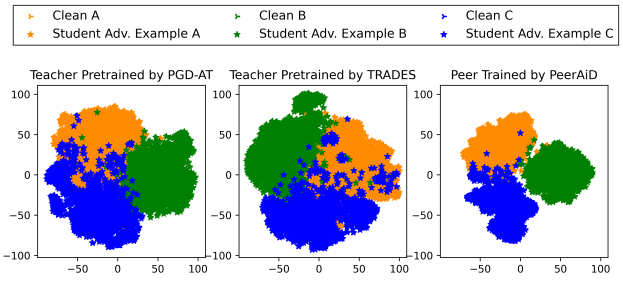


Figure 5. t-SNE results of the penultimate layer representation with the pretrained robust teacher model and PeerAiD.

sharper than the one of PeerAiD. The flat loss landscape of PeerAiD explains the generalization ability and the adversarial robustness of the student network against the adversarial examples generated from the unseen test dataset.

4.9. Visualization of Feature Representation

In Fig. 5, we visualize the feature representation of the teacher (peer) model with the adversarial examples x_S^* generated from the student model. Two ResNet-18 are adversarially pretrained with PGD-AT and TRADES, respectively. Three classes from CIFAR-10 are randomly chosen for better visualization. The peer model trained by PeerAiD has a better ability to embed the natural examples and transferred adversarial examples x_S^* because it clearly separates one class from other classes in Fig. 5, which is consistent with Sec. 4.4. On the other hand, the feature representation of the pre-trained robust teacher model shows more overlaps among classes.

4.10. Visualization of Semantic Gradients

It is generally perceived that adversarially robust models show semantic or interpretable gradient with respect to inputs $\nabla_x L$ [31, 34, 36]. We observe the same phenomenon with the student model trained by PeerAiD in Fig. 6. We visualized the semantic gradient of three models (student, peer, and non-robust model) to find distinct patterns among models with TinyImageNet dataset and ResNet-18.

In the second column of Fig. 6, pixels along the edge of the objects in the pictures have a large magnitude of gradients with respect to input with the student model. It indicates that the edge or unique pattern of the object needs to be attacked to fool the robust student model [37]. In contrast, the last column shows that the high-magnitude gradients are spread across broad regions, showing that non-robust models can be easily fooled by attacking any pixels. Note that the values in Fig. 6 are normalized individually within each column of the same model, and their brightness cannot be directly compared between columns. In the third column of Fig. 6, we also observe an interesting pattern of the gradient of inputs with the peer model trained with PeerAiD. Although the peer model is not robust at all against the adversarial examples aimed at itself as illustrated in Tab. 2, the gradient of input coming from the peer model also exhibits a similar pattern to the student model. This can be interpreted as the peer model has some knowledge of defense similar to the student model.

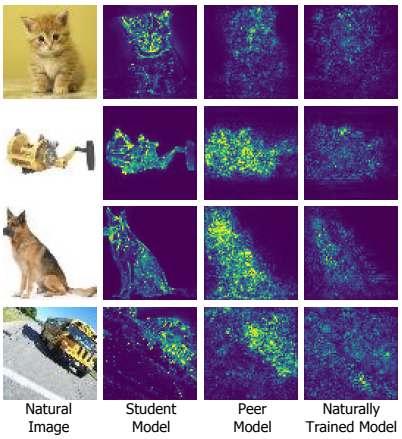


Figure 6. Visualization of semantic gradients. We attack each model and get gradients $\nabla_x L$. Within the same attacked model, we clip pixels to ± 3 standard deviations and sum the absolute pixel values across channels. We scale the values to [0,1].

5. Conclusion

We propose a novel online adversarial distillation method PeerAiD which significantly boosts the robust accuracy. We found that it is possible to build a *peer model* not being robust at all against the white-box attack while being much more robust to the attack examples of the student network. The peer network is specialized in defending the attack samples of the student network and this leads to the more reliable guidance of the peer network than the pretrained robust model used in conventional methods. With peer tutoring, we improved both the robust accuracy and natural accuracy of the student network compared to various baselines.

Acknowledgement. This work was supported by the New Faculty Startup Fund from Seoul National University (50%) and Culture, Sports and Tourism R&D Program through the Korea Creative Content Agency grant funded by the Ministry of Culture, Sports and Tourism in 2024 (CR202104003, 50%)

References

[1] Maksym Andriushchenko and Nicolas Flammarion. Understanding and Improving Fast Adversarial Training. In *NeurIPS*, 2020. 2

[2] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square Attack: A Query-efficient Black-box Adversarial Attack via Random Search. In *ECCV*, 2020. 4, 7

[3] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. In *ICML*, 2018. 4, 5, 7

[4] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. Wav2vec 2.0: A Framework for Self-supervised Learning of Speech Representations. In *NeurIPS*, 2020. 1

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language Models Are Few-shot Learners. In *NeurIPS*, 2020. 1

[6] Nicholas Carlini and David Wagner. Towards Evaluating the Robustness of Neural Networks. In *S&P*, 2017. 7

[7] Tianlong Chen, Zhenyu Zhang, Sijia Liu, Shiyu Chang, and Zhangyang Wang. Robust Overfitting May Be Mitigated by Properly Learned Smoothening. In *ICLR*, 2021. 2, 3, 4, 6, 7

[8] Francesco Croce and Matthias Hein. Reliable Evaluation of Adversarial Robustness with an Ensemble of Diverse Parameter-free Attacks. In *ICML*, 2020. 2, 4, 5, 7

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*, 2018. 1

[10] Chengyu Dong, Liyuan Liu, and Jingbo Shang. Label Noise in Adversarial Training: A Novel Perspective to Study Robust Overfitting. In *NeurIPS*, 2022. 2

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR*, 2021. 1

[12] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *ICLR*, 2021. 7

[13] Micah Goldblum, Liam Fowl, Soheil Feizi, and Tom Goldstein. Adversarially Robust Distillation. In *AAAI*, 2020. 2

[14] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In *ICLR*, 2015. 1, 2, 4

[15] Alex Graves. Sequence Transduction with Recurrent Neural Networks. *arXiv preprint arXiv:1211.3711*, 2012. 1

[16] Qiushan Guo, Xinjiang Wang, Yichao Wu, Zhipeng Yu, Ding Liang, Xiaolin Hu, and Ping Luo. Online Knowledge Distillation via Collaborative Learning. In *CVPR*, 2020. 3

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016. 1, 4, 5, 7

[18] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*, 2015. 1

[19] Bo Huang, Mingyang Chen, Yi Wang, Junda Lu, Minhao Cheng, and Wei Wang. Boosting Accuracy and Robustness of Student Models via Adaptive Adversarial Distillation. In *CVPR*, 2023. 1, 3, 4, 7

[20] Joong-won Hwang, Youngwan Lee, Sungchan Oh, and Yuseok Bae. Adversarial Training with Stochastic Weight Average. In *ICIP*, 2021. 2

[21] Xiaojun Jia, Yuefeng Chen, Xiaofeng Mao, Ranjie Duan, Jindong Gu, Rong Zhang, Hui Xue, and Xiaochun Cao. Revisiting and Exploring Efficient Fast Adversarial Training via LAW: Lipschitz Regularization and Auto Weight Averaging. *arXiv preprint arXiv:2308.11443*, 2023. 7

[22] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 4

[23] Ya Le and Xuan Yang. Tiny Imagenet Visual Recognition Challenge. *CS 231N course*, 2015. 4

[24] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the Loss Landscape of Neural Nets. In *NeurIPS*, 2018. 8

[25] Qizhang Li, Yiwen Guo, Wangmeng Zuo, and Hao Chen. Collaborative Adversarial Training. *arXiv preprint arXiv:2303.14922*, 2023. 3, 4, 7

[26] Chen Liu, Mathieu Salzmann, Tao Lin, Ryota Tomioka, and Sabine Süsstrunk. On the Loss Landscape of Adversarial Training: Identifying Challenges and How to Overcome Them. In *NeurIPS*, 2020. 7

[27] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *ICCV*, 2021. 1

[28] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In *ICLR*, 2018. 1, 2, 3, 4, 5, 7, 8

[29] Ana I Maqueda, Antonio Loquercio, Guillermo Gallego, Narciso García, and Davide Scaramuzza. Event-based Vision Meets Deep Learning on Steering Prediction for Self-driving Cars. In *CVPR*, 2018. 1

[30] Usma Niyaz and Deepti R Bathula. Augmenting Knowledge Distillation with Peer-to-Peer Mutual Learning for Model Compression. In *ISBI*, 2022. 3

[31] Tianyu Pang, Min Lin, Xiao Yang, Jun Zhu, and Shuicheng Yan. Robustness and accuracy could be reconcilable by (proper) definition. In *ICML*, 2022. 3, 6, 8

[32] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in Machine Learning: From Phenomena to Black-box Attacks Using Adversarial Samples. *arXiv preprint arXiv:1605.07277*, 2016. 7

[33] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted Residuals and Linear Bottlenecks. In *CVPR*, 2018. 7

[34] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial Training for Free! In *NeurIPS*, 2019. 8

[35] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open Foundation and Fine-tuned Chat Models. *arXiv preprint arXiv:2307.09288*, 2023. 1

[36] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness May Be at Odds with Accuracy. In *ICML*, 2019. 3, 6, 8

[37] Xin Wang, Jie Ren, Shuyun Lin, Xiangming Zhu, Yisen Wang, and Quanshi Zhang. A Unified Approach to Interpreting and Boosting Adversarial Transferability. In *ICLR*, 2021. 8

[38] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial Weight Perturbation Helps Robust Generalization. In *NeurIPS*, 2020. 7

[39] Guile Wu and Shaogang Gong. Peer Collaborative Learning for Online Knowledge Distillation. In *AAAI*, 2021. 3

[40] Sergey Zagoruyko and Nikos Komodakis. Wide Residual Networks. *arXiv preprint arXiv:1605.07146*, 2016. 4, 5, 8

[41] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically Principled Trade-off between Robustness and Accuracy. In *ICML*, 2019. 2, 3, 4, 5, 6, 8

[42] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep Mutual Learning. In *CVPR*, 2018. 3

[43] Shiji Zhao, Jie Yu, Zhenlong Sun, Bo Zhang, and Xingxing Wei. Enhanced Accuracy and Robustness via Multi-teacher Adversarial Distillation. In *ECCV*, 2022. 2

[44] Xiaokang Zhou, Wei Liang, I Kevin, Kai Wang, Hao Wang, Laurence T Yang, and Qun Jin. Deep-learning-enhanced Human Activity Recognition for Internet of Healthcare Things. *IoT-J*, 2020. 1

[45] Jianing Zhu, Jiangchao Yao, Bo Han, Jingfeng Zhang, Tongliang Liu, Gang Niu, Jingren Zhou, Jianliang Xu, and Hongxia Yang. Reliable Adversarial Distillation with Unreliable Teachers. In *ICLR*, 2022. 1, 2, 3, 4, 5, 7

[46] Bojia Zi, Shihao Zhao, Xingjun Ma, and Yu-Gang Jiang. Revisiting Adversarial Robustness Distillation: Robust Soft Labels Make Student Better. In *CVPR*, 2021. 1, 2, 3, 4, 7