Metric Learning for Adversarial Robustness

Chengzhi Mao
Columbia University
cm3797@columbia.edu

Ziyuan Zhong
Columbia University
ziyuan.zhong@columbia.edu

Junfeng Yang
Columbia University
junfeng@cs.columbia.edu

Carl Vondrick
Columbia University
vondrick@cs.columbia.edu

Baishakhi Ray
Columbia University
rayb@cs.columbia.edu

Abstract

Deep networks are well-known to be fragile to adversarial attacks. We conduct an empirical analysis of deep representations under the state-of-the-art attack method called PGD, and find that the attack causes the internal representation to shift closer to the "false" class. Motivated by this observation, we propose to regularize the representation space under attack with metric learning to produce more robust classifiers. By carefully sampling examples for metric learning, our learned representation not only increases robustness, but also detects previously unseen adversarial samples. Quantitative experiments show improvement of robustness accuracy by up to 4% and detection efficiency by up to 6% according to Area Under Curve score over prior work. The code of our work is available at https://github.com/columbia/Metric_Learning_Adversarial_Robustness.

1 Introduction

Deep networks achieve impressive accuracy and wide adoption in computer vision [17], speech recognition [14], and natural language processing [21]. Nevertheless, their performance degrades under adversarial attacks, where natural examples are perturbed with human-imperceptible, carefully crafted noises [35, 23, 12, 18]. This degradation raises serious concern — especially when we deploy deep networks to safety and reliability critical applications [29, 43, 41, 20, 36]. Extensive efforts [37, 31, 47, 7, 25, 12, 35, 48] have been made to study and enhance the robustness of deep networks against adversarial attacks, where a defense method called adversarial training achieves the state-of-the-art adversarial robustness [19, 16, 46, 49].

To better understand adversarial attacks, we first conduct an empirical analysis of the latent representations under attack for both defended [19, 16] and undefended image classification models. Following the visualization technique in [28, 30, 33], we investigate what happens to the latent representations as they undergo attack. Our results show that the attack shifts the latent representations of adversarial samples away from their *true* class and closer to the *false* class. The adversarial representations often spread across the false class distribution in such a way that the natural images of the false class become indistinguishable from the adversarial images.

Motivated by this empirical observation, we propose to add an additional constraint to the model using metric learning [15, 32, 44] to produce more robust classifiers. Specifically, we add a triplet loss term on the latent representations of adversarial samples to the original loss function. However, the naïve implementation of triplet loss is not effective because the pairwise distances of a natural sample x_a, its adversarial sample x'_a, and a randomly selected natural sample of the false class x_n are hugely uneven. Specifically, given considerable data variance in the false class, x_n is often far from the decision boundary where x'_a resides, therefore x_n is too easy as a negative sample. To address this

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

problem, we sample the negative example for each triplet with the closest example in a mini-batch of training data. In addition, we randomly select another sample x_p in the correct class as the positive example in the triplet data.

Our main contribution is a simple and effective metric learning method, Triplet Loss Adversarial (TLA) training, that leverages triplet loss to produce more robust classifiers. TLA brings near both the natural and adversarial samples of the same class while enlarging the margins between different classes (Sec. 3). It requires no change to the model architecture and thus can improve the robustness on most off-the-shelf deep networks without additional overhead during inference. Evaluation on popular datasets, model architectures, and untargeted, state-of-the-art attacks, including projected gradient descent (PGD), shows that our method classifies adversarial samples more accurately by up to 4% than prior robust training methods [16, 19]; and makes adversarial attack detection [52] more effective by up to 6% according to the Area Under Curve (AUC) score.

2 Related Work

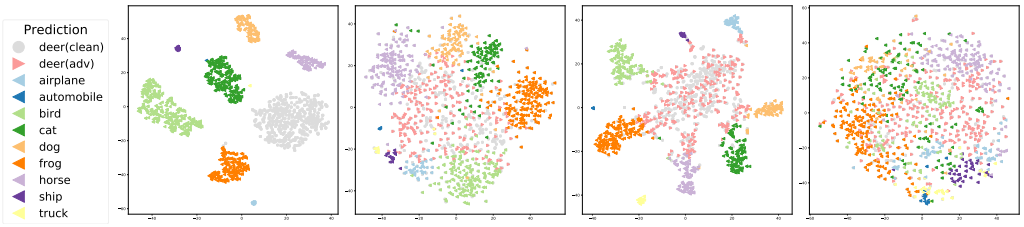
The fact that adversarial noise can fool deep networks was first discovered by Szegedy et al. [35], which started the era of adversarial attacks and defenses for deep networks. Goodfellow et al. [12] then proposed an attack — fast gradient sign method (FGSM) and also constructed a defense model by training on the FGSM adversarial examples. More effective attacks including C&W [5], PGD [19], BIM [18], MIM [9], DeepFool [23], and JSMA [27] are proposed to fool deep networks, which further encourage the research for defense methods.

Madry et al. [19] proposed adversarial training (AT) that dynamically trained the model on the generated PGD attacks, achieving the first empirical adversarial robust classifier on CIFAR-10. Since then, AT became the foundation for the state-of-the-art adversarial robust training method and went through widely and densely scrutiny [3], which achieved real robustness without relying on gradient masking [3, 13, 31, 4, 8]. Recently, Adversarial Logit Pairing (ALP) [16] is proposed with an additional loss term that matches the logit feature from a clean image $\mathbf{x}$ and its corresponding adversarial image $\mathbf{x}'$, which further improves the adversarial robustness. However, this method has a distorted loss function and is not scalable to untargeted attack [11, 22]. In contrast to the ALP loss which uses a pair of data, our method introduces an additional negative example in a triplet of data, which achieves more desirable geometric relationships between adversarial examples and clean examples in feature metric space.

Orthogonal to our method, the concurrent feature denoising method [46] achieves the state-of-the-art adversarial robustness on ImageNet. While their method adds extra denoising block in the model, our method requires no change to the model architecture. Another concurrent work, TRADES [49], achieves improved robustness by introducing Kullback-Leibler divergence loss to a pair of data. In addition, unlabeled data [39] and model ensemble [37, 25] have been shown to improve the robustness of the model. Future work can be explored by combining these methods with our proposed TLA regularization for better adversarial robustness.

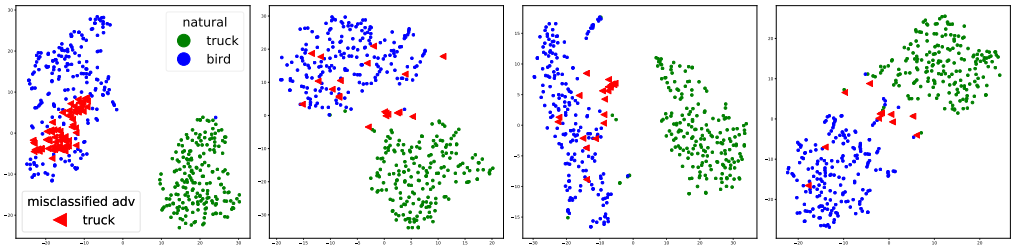
3 Qualitative Analysis of Latent Representations under Adversarial Attack

We begin our investigation by analyzing how the adversarial images are represented by different models. We call the original class of an adversarial image as *true* class and the mis-predicted class of adversarial example as *false* class. Figure 1 shows the visualization of the high dimensional latent representation of sampled CIFAR-10 images with t-SNE [40, 2]. Here, we visualize the penultimate fully connected (FC) layer of four existing models: standard undefended model (UM), model after adversarial training (AT) [19], model after adversarial logit pairing (ALP) [16], and model after our proposed TLA training. Though all the adversarial images belong to the same *true* class, UM separates them into different *false* classes with large margins. The result shows UM is highly non-robust against adversarial attacks because it is very easy to craft an adversarial image that will be mistakenly classified into a different class. With AT and ALP methods, the representations are getting closer together, but one can still discriminate them. Note that, a good robust model will bring the representations of the adversarial images closer to their original *true* class so that it will be difficult to discriminate the adversarial images from the original images. We will leverage this observation to design our approach.



(a) UM (b) AT (c) ALP (d) TLA

Figure 1: **t-SNE Visualization of adversarial images from the same *true* class which are mistakenly classified to different *false* classes.** The figure shows representations of second to last layer of 1000 adversarial examples crafted from 1000 natural (clean) test examples from CIFAR-10 dataset, where the *true* class is "deer." The different colors represent different *false* classes. The gray dots further show 500 randomly sampled natural deer images. Notice that for (a) undefended model (UM), the adversarial attacks clearly separate the images from the same "deer" category into different classes. (b) adversarial training (AT) and (c) adversarial logit pairing (ALP) method still suffer from this problem at a reduced level. In contrast, our proposed ATL (see (d)) clusters together all the examples from the same *true* class, which improves overall robustness.



(a) UM (b) AT (c) ALP (d) TLA

Figure 2: **Illustration of the separation margin of adversarial examples from the natural images of the corresponding false class.** We show t-SNE visualization of the second to last layer representation of test data from two different classes across four models. The blue and green dots are 200 randomly sampled natural images from "bird" and "truck" classes respectively. The red triangles denote adversarial (adv) truck images but mispredicted as "bird." Notice that for (a) UM, the adversarial examples are moved to the center of the false class, making it hard to separate from them. (b) AT and (c) ALP achieve some robustness by separating adversarial and natural images, but they are still close to each other. Plot (d) shows our proposed TLA training promotes the mispredicted adversarial examples to lie on the edge of the natural images false class and can still be separated, which improves the robustness.

In Figure 2, we further analyze how the representation of images of one class is attacked into the neighborhood of another class. The green and blue dots are the natural images of trucks and birds, respectively. The red triangles are the adversarial images of trucks mispredicted as birds. For UM model (Figure 2a), all the adversarial attacks successfully get into the center of the false class. The AT and ALP models achieve some robustness by separating some adversarial images from natural images, but most adversarial images are still inside the false class. A good robust model should promote the representations of adversarial examples away from the false class, as shown in Figure 2d. Such separation not only improves the adversarial classification accuracy but also helps to reject the mispredicted adversarial attacks, because the mispredicted adversaries tend to lie on edge.

Based on these two observations, we build a new approach that ensures adversarial representations will be (i) closer to the natural image representations of their true classes, and (ii) farther from the natural image representations of the corresponding false classes.

4 Approach

Inspired by the adversarial feature space analysis, we add an additional constraint to the model using metric learning. Our motivation is that the triplet loss function will pull all the images of one class, both natural and adversarial, closer while pushing the images of other classes far apart. Thus, an image and its adversarial counterpart should be on the same manifold, while all the members of the false class should be forced to be separated by a large margin.

Notations. For an image classification task, let M be the number of classes to predict, and N be the number of training examples. We formulate the deep network classifier as $F_{\boldsymbol{\theta}}(\mathbf{x}) \in \mathbb{R}^M$ as a probability distribution, where $\mathbf{x}$ is the input variable, $\mathbf{y}$ is the output ground-truth, and $\boldsymbol{\theta}$ is the network's parameters to learn (we simply use $F(\mathbf{x})$ most of time); $\mathcal{L}(F(\mathbf{x}), y)$ is the loss function.

Assume that an adversary is capable of launching adversarial attacks bounded by p-norm, i.e., the adversary can perturb the input pixel by ϵ bounded by $L_p, p = 0, 2, \infty$, let $\mathbf{I}(\mathbf{x}, \epsilon)$ denote the L_p ball centered at $\mathbf{x}$ with radius ϵ. We focus on the study of *untargeted* attack, i.e., the objective is to generate $\mathbf{x}' \in \mathbf{I}(\mathbf{x}, \epsilon)$ such that $F(\mathbf{x}') \neq F(\mathbf{x})$.

Triplet Loss. Triplet loss is a widely used strategy for metric learning. It trains on a triplet input $\left\{ \left(\mathbf{x}_a^{(i)}, \mathbf{x}_p^{(i)}, \mathbf{x}_n^{(i)}\right) \right\}$, where the elements in the positive pair $\left(\mathbf{x}_a^{(i)}, \mathbf{x}_p^{(i)}\right)$ are clean images from the same class and the elements in the negative pair $\left(\mathbf{x}_a^{(i)}, \mathbf{x}_n^{(i)}\right)$ are from different classes [32, 15]. $\mathbf{x}_p^{(i)}$, $\mathbf{x}_a^{(i)}$, and $\mathbf{x}_n^{(i)}$ are referred as *positive*, *anchor*, and *negative* examples of the triplet loss. The embeddings are optimized such that examples of the same class are pulled together and the examples of different classes are pushed apart by some margin [34]. The standard triplet loss for clean images is as follows:

$$\sum_i^N \mathcal{L}_{trip}(\mathbf{x}_a^{(i)}, \mathbf{x}_p^{(i)}, \mathbf{x}_n^{(i)}) = \sum_i^N [D(h(\mathbf{x}_a^{(i)}), h(\mathbf{x}_p^{(i)})) - D(h(\mathbf{x}_a^{(i)}), h(\mathbf{x}_n^{(i)})) + \alpha]_+$$

where, $h(\mathbf{x})$ maps from the input $\mathbf{x}$ to the embedded layer, $\alpha \in \mathbb{R}^+$ is a hyper-parameter for margin and $D(h(\mathbf{x}_i), h(\mathbf{x}_j))$ denotes the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ in the embedded representation space. In this paper, we define the embedding distance between two examples using the angular distance [42]: $D(h(\mathbf{x}_a^{(i)}), h(\mathbf{x}_{p,n}^{(j)})) = 1 - \frac{|h(\mathbf{x}_a^{(i)}) \cdot h(\mathbf{x}_{p,n}^{(j)}))|}{||h(\mathbf{x}_a^{(i)})||_2 ||h(\mathbf{x}_{p,n}^{(j)}))||_2}$, where we choose to encode the information in the angular metric space.

Metric Learning for Adversarial Robustness. We add triplet loss to the penultimate layer's representation. Different from standard triplet loss where all the elements in the triplet loss term are clean images [32, 50], at least one element in the triplet loss under our setting will be an adversarial image. Note that generating adversarial examples is more computational intensive compared with just taking the clean images. For efficiency, we only generate one adversarial perturbed image for each triplet data, using the same method introduced by Madry et al. [19]. Specifically, given a clean image $\mathbf{x}^{(i)}$, we generate the adversarial image $\mathbf{x}'^{(i)}$ based on $\nabla_{\mathbf{x}} \mathcal{L}(F(\mathbf{x}), y)$ (standard loss without the triplet loss) with PGD method. We do not add the triplet loss term into the loss of adversarial example generation due to its inefficiency.

The other elements in the triplet data are clean images. We forward the triplet data in parallel through the model and jointly optimize the cross-entropy loss and the triplet loss, which enables the model to capture the stable metric space representation (triplet loss) with semantic meaning (cross-entropy loss). The total loss function is formulated as follows:

$$\begin{aligned} \mathcal{L}_{all} = \sum_i^N \mathcal{L}_{ce}(f(\mathbf{x}_a'^{(i)}), y^{(i)}) + \lambda_1 \mathcal{L}_{trip}(h(\mathbf{x}_a'^{(i)})), h(\mathbf{x}_p^{(i)}), h(\mathbf{x}_n^{(i)})) + \lambda_2 \mathcal{L}_{norm} \\ \mathcal{L}_{norm} = ||h(\mathbf{x}_a'^{(i)})||_2 + ||h(\mathbf{x}_p^{(i)})||_2 + ||h(\mathbf{x}_n^{(i)})||_2 \end{aligned} \tag{1}$$

where λ_1 is a positive coefficient trading off the two losses; $\mathbf{x}_a'^{(i)}$ (anchor example) is an adversarial counterpart based on $\mathbf{x}_a^{(i)}$; $\mathbf{x}_p^{(i)}$ (positive example) is a clean image from the same class of $\mathbf{x}_a^{(i)}$; $\mathbf{x}_n^{(i)}$ (negative example) is a clean image from a different class; λ_2 is the weight for the feature norm decay term, which is also applied in [32] to reduce the L_2 norm of the feature.

Notice that, besides the TLA set-up in equation 1, an adversarial perturbed image can be the positive example, and a clean image can be the anchor example (i.e., switch the anchor and the positive), where we refer it as TLA-SA (Sec 5). We choose the adversarial example as the anchor for TLA according to the experimental result. Intuitively, the adversarial image is picked as the anchor because it tends to be closer to the decision boundary between the "true" class and the "false" class. As an anchor, the adversarial example is considered in both the positive pair and the negative pair, which gives more-useful gradients for the optimization. The modified triplet loss for adversarial robustness is shown in Figure 3.

Figure 3: Illustration of the triplet loss for adversarial robustness (TLA). The red circle is an adversarial example, while the green and the blue circles are clean examples. The anchor and positive belong to the same class. The negative (blue), from a different class, is the closest image to the anchor (red) in feature space. TLA learns to pull the *anchor* and *positive* from the true class closer, and push the *negative* of false classes apart.

Negative Sample Selection. In addition to the anchor selection, the selection of the negative example is crucial for the training process, because most of the negative examples are easy examples that already satisfy the margin constraint of pairwise distance and thus contribute useless gradients [32, 10]. Using the representation angular distance we predefine, we select negative samples as the nearest images to the anchor from a false class. As a result, our model is able to learn to enlarge the boundary between the adversarial samples and their closest negative samples from the other classes.

Unfortunately, finding the closest negative samples from the entire training set is computationally intensive. Besides, using very hard negative examples have been found to decrease the network's convergence speed [32] significantly. Instead, we use a semi-hard negative example, where we select the closest sample in a mini-batch. We demonstrate the advantage of this sampling strategy by comparing it with the random sampling (TLA-RN). The results are shown in Sec 5. Other strategies of sampling negative samples such as DAML [10] could also be applied here, which uses an adversarial generator to exploit hard negative examples from easy ones.

Implementation Details. We apply our proposed triplet loss on the embedding of the penultimate layer of the neural network for classification tasks. Since the following transformation only consists of a linear layer and a softmax layer, small fluctuation to this embedding only brings monotonous adjustment to the output controlled by some tractable Lipschitz constant [7, 24]. We do not apply triplet loss on the logit layer but on the penultimate layer, because the higher dimensional penultimate layer tends to preserve more information. We also construct two triplet loss terms on CIFAR-10 and Tiny ImageNet, adding another positive example while reusing the anchor and negative example, which achieves better performance [34, 6]. The details of the algorithm are introduced in the appendix.

5 Experiments

Experimental Setting. We validate our method on different model architectures across three popular datasets: MNIST, CIFAR-10, and Tiny-ImageNet. We compare the performance of our models with the following baselines: **Undefended Model (UM)** refers to the standard training without adversarial samples, **Adversarial Training (AT)** refers to the min-max optimization method proposed in [19], **Adversarial Logit Pairing (ALP)** refers to the logit matching method which is currently the state-of-the-art [16]. We use **TLA** to denote the triplet loss adversarial training mentioned in Section 4. To further evaluate our design choice, we study two variants of TLA: **Random Negative (TLA-RN)**, which refers to our proposed triplet loss training method with a randomly sampled negative example, and **Switch Anchor (TLA-SA)**, which sets the anchor to be natural example and the positive to be adversarial example (i.e., switching the anchor and the positive of our proposed method).

We conduct all of our experiments using TensorFlow v1.13 [1] on a single Tesla V100 GPU with a memory of 16GB. We adopt the untargeted adversarial attacks during all of our training processes, and evaluate the models with both white-box and black-box *untargeted* attacks instead of the targeted attacks following the suggestions in [11] (a defense robust only to targeted adversarial attacks is weaker than one robust to untargeted adversarial attacks). In order to be comparable to the original paper in AT and ALP, we mainly evaluate the model under the L_∞ bounded attacks. We also evaluate the models under other norm-bounded attacks (L_0, L_2). The PGD and 20PGD in our Table 1 refer to the PGD attacks with the random restart of 1 time and 20 times, respectively. For black-box (BB) attacks, we use the transfer based method [26]. We set $\lambda = 0.5$ for ALP method as the original paper. All the other implementation details are discussed in the appendix.

		MNIST								
	Attacks (Steps)	Clean -	FGSM (1)	BIM (40)	C&W (40)	PGD (40)	PGD (100)	20PGD (100)	MIM (200)	BB (100)
Methods	UM	99.20%	34.48%	0%	0%	0%	0%	0%	0%	81.81%
	AT	99.24%	97.31%	95.95%	96.66%	96.58%	94.82%	93.87%	95.47%	96.67%
	ALP	98.91%	97.34%	96.00%	96.50%	96.62%	95.06%	94.93%	95.41%	96.95%
	TLA-RN	99.50%	98.12%	97.17%	97.17%	97.64%	**97.07%**	96.73%	**96.84%**	97.69%
	TLA-SA	99.44%	98.14%	97.08%	**97.45%**	97.50%	96.78%	95.64%	96.45%	97.65%
	TLA	**99.52%**	**98.17%**	**97.32%**	97.25%	**97.72%**	96.96%	**96.79%**	96.64%	**97.73%**
		CIFAR-10								
	Attacks (Steps)	Clean -	FGSM (1)	BIM (7)	C&W (30)	PGD (7)	PGD (20)	20PGD (20)	MIM (40)	BB (7)
Methods	UM	**95.01%**	13.35%	0%	0%	0%	0%	0%	0%	7.60%
	AT	87.14%	55.63%	48.29%	46.97%	49.79%	45.72%	45.21%	45.16%	62.83%
	ALP	89.79%	**60.29%**	50.62%	47.59%	51.89%	48.50%	45.98%	45.97%	67.27%
	TLA-RN	81.02%	55.41%	51.44%	49.66%	52.50%	49.94%	45.55%	49.63%	65.96%
	TLA-SA	86.19%	58.80%	52.19%	49.64%	53.53%	49.70%	49.15%	49.29%	61.67%
	TLA	86.21%	58.88%	**52.60%**	**50.69%**	**53.87%**	**51.59%**	**50.03%**	**50.09%**	**70.63%**
		Tiny ImageNet								
	Attacks (Steps)	Clean -	FGSM (1)	BIM (10)	C&W (10)	PGD (10)	PGD (20)	20PGD (20)	MIM (40)	BB (10)
Methods	UM	**60.64%**	1.15%	0.01%	0.01%	0.01%	0%	0%	0%	9.99%
	AT	44.77%	21.99	19.59%	17.34%	19.79%	19.44%	19.25%	19.28%	27.73%
	ALP	41.53%	21.53%	20.03%	16.80%	20.18%	19.96%	19.76%	19.85%	**30.31%**
	TLA-RN	42.11%	21.47%	20.03%	17.00%	20.05%	19.93%	19.81%	19.91%	30.18%
	TLA-SA	41.43%	22.09%	20.77%	17.28%	20.82%	20.63%	**20.50%**	20.61%	29.96%
	TLA	40.89%	**22.12%**	**20.77%**	**17.48%**	**20.89%**	**20.71%**	20.47%	**20.69%**	29.98%

Table 1: Classification accuracy under 8 different L_∞ bounded *untargeted* attacks on MNIST (L_∞ =0.3), CIFAR-10 (L_∞ =8/255), and Tiny-ImageNet (L_∞ =8/255). The best results of each column are in **bold** and the empirical lower bound (the lowest accuracy of each row if any) for each method is underlined. TLA improves the adversarial accuracy by up to 1.86%, 4.12% , and 0.84% on MNIST, CIFAR-10, and Tiny ImageNet respectively.

5.1 Effect of TLA on Robust Accuracy

MNIST consists of a training set of 55,000 images (excluding the 5000 images for validation as in [19]) and a testing set of 10,000 images. We use a variant of LeNet CNN architecture which has batch normalization for all the methods. The details of network architectures and hyper-parameters are summarized in the appendix. We adopt the $L_\infty = 0.3$ bounded attack during the training and evaluation. We generate adversarial examples using PGD with 0.01 step size for 40 steps during the training. In addition, we conduct different types of $L_\infty = 0.3$ bounded attacks to achieve good evaluations. The adversarial classification accuracy of different models under various adversarial attacks is shown in Table 1. As shown, we improve the empirical state-of-the-art adversarial accuracy by up to **1.86%** on 20PGD attacks (100 steps PGD attacks with 20 times of random restart), along with **0.28%** improvement on the clean data.

CIFAR-10 consists of 32×32×3 color images in 10 classes, with 50k images for training and 10k images for testing. We follow the same wide residual network architecture and the same hyper-parameters settings as AT [19]. As shown in Table 1, our method achieves up to **4.12%** adversarial accuracy improvement over the baseline methods under the strongest 20PGD attacks (20 steps PGD attack with 20 times of restart). Note that our method results in a minor decrease of standard accuracy, but such loss of generic accuracy is observed in all the existing robust training models [38, 49]. The comparison with TLA-RN illustrates the effectiveness of the negative sampling strategy. According to the result of the TLA-SA, our selection of the adversarial example as the anchor also achieves better performance than the method which chooses the clean image as the anchor.

Tiny Imagenet is a tiny version of ImageNet consisting of color images with size 64×64×3 belonging to 200 classes. Each class has 500 training images and 50 validation images. Due to the GPU limit, we adapt the ResNet 50 architectures for the experiment. We adopt $L_\infty = 8/255$ for both training and validation. During training, we use 7 step PGD attack with step size $2/255$ to generate the adversarial samples. As shown in Table 1, our proposed model achieves higher adversarial accuracy under white box adversarial attacks by up to **0.84%** on MIM attacks.

		Mini-Batch TT (s)	Total TT (s)	Clean	FGSM(1)	BIM(7)	C&W(30)	PGD(20)	MIM(40)
Negative Size	1	0	1.802	81.02%	55.41%	51.44%	49.66%	49.94%	49.63%
	250	0.467	2.259	86.38%	59.05%	53.02%	50.49%	50.71%	50.31%
	500	0.908	2.688	**88.32%**	**60.02%**	53.20%	**51.30%**	50.46%	50.07%
	1000	1.832	3.621	86.71%	59.08%	**53.25%**	50.88%	**51.22%**	**50.74%**
	2000	3.548	5.992	87.45%	59.23%	52.52%	50.57%	50.20%	49.79%

Table 2: The effect of mini-batch size of *negative* samples on training time (TT) per iteration and adversarial robustness ($L_\infty = 8/255$) on CIFAR-10 dataset. The best results of each column are shown in **bold**. The number of steps for each attack is shown in the parenthesis. The training time grows linearly as the size of the mini-batch grows. The adversarial robustness peaks at size 500 to 1000, which validate that semi-hard negative examples are crucial for TLA.

		MNIST (LeNet)				**CIFAR-10 (WRN)**			
	Attacks	JSMA (L_0)	PGD (L_2)	C&W (L_2)	DeeoFool (L_2)	JSMA(L_0)	PGD (L_2)	C&W (L_2)	DeeoFool (L_2)
Methods	AT	99.08%	96.61%	99.08%	99.13%	40.4%	36.8%	50.0%	67.7%
	ALP	98.83%	96.28%	98.91%	98.95%	36.9%	38.6%	51.2%	43.5%
	TLA	**99.32%**	**97.38%**	**99.36%**	**99.35%**	**48.6%**	**41.1%**	**53.5%**	**80.8%**

Table 3: Classification accuracy of two baseline methods and TLA method on 4 unseen types of attacks (L_0 and L_2 norm bounded). All the models are only trained on the L_∞ bounded attacks. The best results of each column are shown in **bold**. TLA improves the adversarial accuracy by up to 1.10% and 13.1% on MNIST and CIFAR-10 dataset respectively. The results demonstrate that TLA generalizes better to unseen types of attacks.

Effect of the mini-batch size of negative samples of TLA. Compared with retrieving from the whole dataset, the mini-batch based method can mitigate the computational overhead by finding the nearest neighbor from a batch rather than from the whole training set. The size of the mini-batch size controls the hardness level of the negative samples, where larger mini-batch size makes harder negative ones. We train models with different mini-batch size and evaluate the robustness of the model using five untargeted, L_∞ bounded attacks. As shown in Table 2, the total training time grows linearly as the size of the mini-batch increases, which triples for size 2000 compared with size 1. The adversarial robustness first increases and then decreases after the mini-batch size reaches 1000 (very hard negative examples hurt performance). Being consistent with the observation in standard metric learning [32, 51], our results show that it is important to train TLA with semi-hard negative examples by choosing the proper mini-batch size.

Generalization to Unseen Types of Attacks. After training the models, both baselines and ours, with L_∞ bounded attacks, we evaluate them on unseen L_0-bounded [27] and L_2-bounded attacks [23, 5, 19, 5]. We set $L_0 = 0.1$ and $L_0 = 0.02$ bound for JSMA on MNIST and CIFAR-10 dataset respectively. For L_2 norm bounded PGD and C&W attacks, we set the bound as $L_2 = 0.1$ and $L_2 = 32$ on MNIST and CIFAR-10 respectively. We apply 40 steps of PGD and C&W on MNIST, and 10 steps of PGD and C&W on CIFAR-10. We apply 2 steps for DeepFool attack for both dataset. Due to the slow speed of JSMA, we only run 1000 test samples on CIFAR-10. Table 3 shows that TLA improves the adversarial accuracy by up to 1.10% and 13.1% on MNIST and CIFAR-10 respectively, which demonstrates that TLA generalizes better to unseen attacks than baseline models.

Performance on Different Model Architectures. To demonstrate that TLA is general for different model architectures, we conduct experiments using multi-layer perceptron (MLP) and ConvNet [47] architectures. Results in Table 4 show that TLA achieves better adversarial robustness by up to 4.27% and 0.55% on MNIST and CIFAR-10 respectively.

		MNIST (MLP)					**Cifar10 (ConvNet)**				
	Attacks	Clean	FGSM	BIM	C&W	PGD	Clean	FGSM	BIM	C&W	PGD
	Steps	-	1	40	40	100	-	1	7	30	20
Methods	UM	**98.27%**	5.23%	0%	0%	0%	**77.84%**	3.50%	0.09%	0.08%	0.03%
	AT	96.43%	73.25%	57.83%	62.60%	58.10%	67.60%	40.26%	36.34%	33.17%	34.83%
	ALP	95.56%	77.08%	64.39%	63.46%	64.13%	66.18%	39.45%	36.15%	32.55%	35.32%
	TLA	97.15%	**78.44%**	**65.47%**	**67.73%**	**65.88%**	67.48%	**40.76%**	**36.77%**	**33.27%**	**35.38%**

Table 4: Effect of TLA on different neural network architectures. The table lists classification accuracy under various L_∞ bounded *untargeted* attacks on MNIST (L_∞ = 0.3) and Cifar10 (L_∞ = 8/255). Overall, TLA improves adversarial accuracy.

5.2 Effect of TLA on Adversarial vs. Natural Image Separation

Recall in Figure 2b and Figure 2c, the representations of adversarial images are shifted toward the false class. A robust model should separate them apart. To quantitatively evaluate how well TLA training helps with separating the adversarial examples from the natural images of the corresponding 'false' classes, we define the following metric.

Let $\{c_k^i\}$ denote the *embedded representations* of all the natural images from class c_k, where $i = 1, \ldots, |c_k|$, and $|c_k|$ is the total number of images in class c_k. Then, the average pairwise within-class distance of these embedded images is: $\sigma_{c_k}^{ntrl} = \frac{2}{|c_k|(|c_k|-1)} \sum_{i=1}^{|c_k|-1} \sum_{j=i+1}^{|c_k|} D(c_k^i, c_k^j)$. Let $\{c_k'^q\}$ further denote embedded representations of all the adversarial examples that are misclassified to class c_k, where $q = 1, \ldots, |c_k'|$, and $|c_k'|$ is the total number of such examples. Note that, class c_k is the 'false' class to those adversarial images. Then, the distance between an adversarial images $c_k'^i$ and a natural image c_k^j is: $D(c_k'^i, c_k^j)$, and the average pair-wise distance between adversary image and natural images is: $\sigma'^{adv}_{c_k} = \frac{1}{|c_k'||c_k|} \sum_{i=1}^{|c_k'|} \sum_{j=1}^{|c_k|} D(c_k'^i, c_k^j)$. We then define the ratio $r_{c_k} = \frac{\sigma_{c_k}^{adv}}{\sigma_{c_k}^{ntrl}}$ as a metric to evaluate how close the adversarial images are w.r.t. the natural images of the 'false' class while compared with the average pairwise within-class distance of all the natural images of that class. Finally, for all classes we compute the average ratio as $r = \frac{1}{M} \sum_{k=1}^{M} (r_{c_k})$. Note that, any good robust method should increase the value of r, indicating σ^{adv} is far from σ^{ntrl}, i.e., they are better separated than the natural cluster, as shown in Figure 2d.

	Dataset	MNIST		CIFAR-10		Tiny ImageNet	
	Perturbation Level	$L_\infty = 0.03$	$L_\infty = 0.3$	$L_\infty = \frac{8}{255}$	$L_\infty = \frac{25}{255}$	$L_\infty = \frac{8}{255}$	$L_\infty = \frac{25}{255}$
Methods	AT	1.288	1.308	1.053	1.007	**0.9949**	0.9656
	ALP	1.398	1.394	1.038	1.210	0.9905	0.9722
	TLA	**1.810**	**1.847**	**1.093**	**1.390**	0.9937	**0.9724**

Table 5: Average Ratio (r) of mean distance between adversary points and natural points over the mean intra-class distance. The best results of each column are in **bold**. The results illustrate that TLA increases the relative distance of adversarial images w.r.t. the natural images of the respective false classes, which illustrates that TLA achieves more desirable geometric feature space under attacks.

For every dataset, we estimate the ratios under two different perturbation levels of PGD attacks for all the models. As shown in Table 5, stronger attacks (larger perturbation level) tend to shift their latent representation more toward the false class. For Tiny-ImageNet, the adversarial examples are even closer ($r < 1$) to the false class's manifold than the corresponding natural images to themselves, which explains the low adversarial accuracy on this dataset. In almost all the settings, TLA leads to higher r values of separation than the other baseline methods. This indicates TLA is most effective in pulling apart the misclassified adversary examples from their false class under both small and large perturbations attacks.

	Dataset	MNIST		CIFAR-10		Tiny-ImageNet	
	Type	Adv	Natural	Adv	Natural	Adv	Natural
Method	AT	93.01%	98.68%	47.46%	87.06%	20.20%	**36.6%**
	ALP	95.20%	98.43%	48.85%	**89.63%**	20.33%	35.23%
	TLA	**96.98%**	**99.47%**	**51.74%**	86.29%	**20.72%**	33.99%

Table 6: Accuracy of K-Nearest Neighbors classifier with $K = 50$, illustrating TLA has better similarity measures in embedding space even with adversarial samples. The best results of each column are in **bold**.

We further conduct the nearest neighbor analysis on the latent representations across all the models. The results illustrate the advantage of our learned representations for retrieving the nearest neighbor under adversarial attacks (See Figure 4). Table 6 numerically shows that the latent representation of TLA achieves higher accuracy using K-Nearest Neighbors classifier than baseline methods.

5.3 Effect of TLA on Adversarial Image Detection

Detecting mis-predicted adversarial inputs is another dimension to improve a model's robustness. Forward these detected adversarial examples to humans for labeling can significantly improve the reliability of the system under adversarial cases. Given that TLA separates further the adversarial examples from the natural examples of the false class, we can detect more mis-classified examples by filtering out the outliers. We conduct the following experiments.

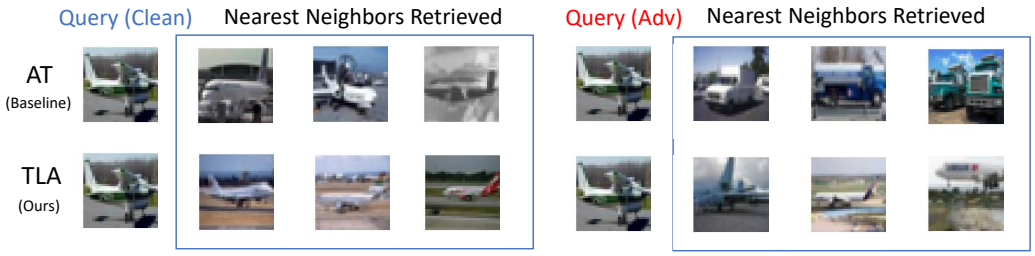
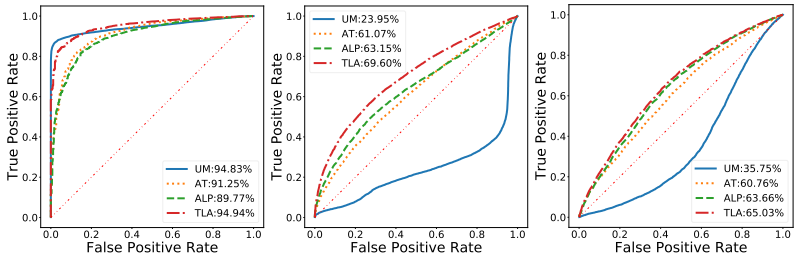


Figure 4: Visualization of nearest neighbor images while querying about a “plane" on AT and TLA trained models. For a natural query image, both methods retrieve correct images (left column). However, given an adversarial query image (right column), the AT retrieves false “truck" images indicating the perturbation moves the representation of the “plane" into the neighbors of “truck," while TLA still retrieves images from the true "plane" class.



(a) MNIST (b) CIFAR-10 (c) Tiny ImageNet

Figure 5: The ROC curve and AUC scores of detecting mis-classified adversarial examples. We train a GMM model on half clean and half adversarial examples (generated with perturbation level $\epsilon = 0.03/1$(40 steps) for MNIST, $\epsilon = 8/255$(7 steps) for CIFAR-10, and $\epsilon = 8/255$(7 steps) for Tiny-ImageNet), and then test the detection model on 10k natural test images and 10k adversary test images (generated with perturbation level $\epsilon = 0.3/1$(100 steps) for MNIST, $\epsilon = 25/255$(20 steps) for CIFAR-10, and $\epsilon = 25/255$(30 steps) for Tiny-ImageNet). The numerical results for AUC score are shown in the legend. Note that both the ROC curve of TLA is on the top and the AUC score of TLA is the highest, which shows TLA (our method) achieves higher detection efficiency for adversarial examples.

Following the adversarial detection method proposed in [52], we train a Gaussian Mixture Model for 10 classes where the density function of each class is captured by one Gaussian distribution. For each test image, we assign a confidence score of a class based on the Gaussian distribution density of the class at that image, as shown in [45]. We assign these confidence scores for all the 10 classes for each test image. We then pick the class with the largest confidence value as the assigned class of the image. We further rank all the test images based on the confidence value of their assigned class. We reject those with lower confidence scores below a certain threshold. This method serves as an additional confidence metric to detect adversarial examples in a real-world setting.

We conduct the detection experiment for mis-classified images on 10k clean images and 10k adversarial images. As shown in Figure 5, the ROC-curves and AUC score demonstrate that our learned representations are superior in adversarial example detection. Compared with other robust training models, TLA improves the AUC score by up to 3.69%, 6.45%, and 1.37% on MNIST, CIFAR-10, and Tiny ImageNet respectively. The detection results here are consistent with the visual results shown in Figure 2.

6 Conclusion

Our novel TLA regularization is the first method that leverages metric learning for adversarial robustness on deep networks, which significantly increases the model robustness and detection efficiency. TLA is inspired by the evidence that the model has distorted feature space under adversarial attacks. In the future, we plan to enhance TLA using more powerful metric learning methods, such as the N-pair loss. We believe TLA will also be beneficial for other deep network applications that desire a better geometric relationship in hidden representations.

7 Acknowledgements

We thank the anonymous reviewers, Prof. Suman Jana, Prof. Shih-Fu Chang, Prof. Janet Kayfetz, Ji Xu, and Vaggelis Atlidakis for their valuable comments, which substantially improved our paper. This work is in part supported by NSF grant CNS-15-64055; ONR grants N00014-16-1- 2263 and N00014-17-1-2788; a JP Morgan Faculty Research Award; a DiDi Faculty Research Award; a Google Cloud grant; an Amazon Web Services grant; NSF CRII 1850069; and NSF CCF-1845893.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Sanjeev Arora, Wei Hu, and Pravesh K. Kothari. An analysis of the t-sne algorithm for data visualization. In *COLT 2018*, 2018.

[3] Anish Athalye, Nicholas Carlini, and David A. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, pages 274–283, 2018.

[4] Jacob Buckman, Aurko Roy, Colin Raffel, and Ian J. Goodfellow. Thermometer encoding: One hot way to resist adversarial examples. In *6th International Conference on Learning Representations*, 2018.

[5] Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy*, pages 39–57, 2017.

[6] Weihua Chen, Xiaotang Chen, Jianguo Zhang, and Kaiqi Huang. Beyond triplet loss: a deep quadruplet network for person re-identification. *CoRR*, abs/1704.01719, 2017.

[7] Moustapha Cissé, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *International Conference on Machine Learning*, pages 854–863, 2017.

[8] Guneet S. Dhillon, Kamyar Azizzadenesheli, Zachary C. Lipton, Jeremy Bernstein, Jean Kossaifi, Aran Khanna, and Animashree Anandkumar. Stochastic activation pruning for robust adversarial defense. In *6th International Conference on Learning Representations*, 2018.

[9] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9185–9193, 2018.

[10] Yueqi Duan, Wan qing Zheng, Xudong Lin, Jiwen Lu, and Jie Zhou. Deep adversarial metric learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2780–2789, 2018.

[11] Logan Engstrom, Andrew Ilyas, and Anish Athalye. Evaluating and understanding the robustness of adversarial logit pairing. *CoRR*, abs/1807.10272, 2018.

[12] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2014.

[13] Chuan Guo, Mayank Rana, Moustapha Cissé, and Laurens van der Maaten. Countering adversarial images using input transformations. *CoRR*, abs/1711.00117, 2017.

[14] Awni Y. Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. Deep speech: Scaling up end-to-end speech recognition. *CoRR*, abs/1412.5567, 2014.

[15] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *ICLR*, 2015.

[16] Harini Kannan, Alexey Kurakin, and Ian J. Goodfellow. Adversarial logit pairing. *CoRR*, abs/1803.06373, 2018.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*, pages 1097–1105, 2012.

[18] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *CoRR*, abs/1607.02533, 2017.

[19] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

[20] Chengzhi Mao, Kangbo Lin, Tiancheng Yu, and Yuan Shen. A probabilistic learning approach to uwb ranging error mitigation. In *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018.

[21] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119, 2013.

[22] Marius Mosbach, Maksym Andriushchenko, Thomas Alexander Trost, Matthias Hein, and Dietrich Klakow. Logit pairing methods can fool gradient-based attacks. *CoRR*, abs/1810.12042, 2018.

[23] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, pages 427–436, 2015.

[24] Adam M. Oberman and Jeff Calder. Lipschitz regularized deep neural networks converge and generalize. *CoRR*, abs/1808.09540, 2018.

[25] Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. Improving adversarial robustness via promoting ensemble diversity. *CoRR*, abs/1901.08846, 2019.

[26] Nicolas Papernot, Patrick D. McDaniel, and Ian J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *CoRR*, abs/1605.07277, 2016.

[27] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. *CoRR*, abs/1511.07528, 2015.

[28] Magdalini Paschali, Sailesh Conjeti, Fernando Navarro, and Nassir Navab. Generalizability vs. robustness: Adversarial examples for medical imaging. *CoRR*, abs/1804.00504, 2018.

[29] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. Deepxplore: Automated whitebox testing of deep learning systems. *CoRR*, abs/1705.06640, 2017.

[30] Adnan Siraj Rakin, Jinfeng Yi, Boqing Gong, and Deliang Fan. Defend deep neural networks against adversarial examples via fixed anddynamic quantized activation functions. *CoRR*, abs/1807.06714, 2018.

[31] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *CoRR*, abs/1805.06605, 2018.

[32] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823, 2015.

[33] Chuanbiao Song, Kun He, Liwei Wang, and John E. Hopcroft. Improving the generalization of adversarial training with domain adaptation. *CoRR*, abs/1810.00740, 2018.

[34] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, pages 4004–4012. IEEE Computer Society, 2016.

[35] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

[36] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. *CoRR*, abs/1708.08559, 2017.

[37] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Dan Boneh, and Patrick D. McDaniel. Ensemble adversarial training: Attacks and defenses. *CoRR*, abs/1705.07204, 2017.

[38] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *stat*, 1050:11, 2018.

[39] Jonathan Uesato, Jean-Baptiste Alayrac, Po-Sen Huang, Robert Stanforth, Alhussein Fawzi, and Pushmeet Kohli. Are labels required for improving adversarial robustness? *CoRR*, abs/1905.13725, 2019.

[40] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[41] Hao Wang, Chengzhi Mao, Hao He, Mingmin Zhao, Tommi S. Jaakkola, and Dina Katabi. Bidirectional inference networks: A class of deep bayesian networks for health profiling. In *The Thirty-Third AAAI Conference on Artificial Intelligence*, pages 766–773, 2019.

[42] Jian Wang, Feng Zhou, Shilei Wen, Xiao Liu, and Yuanqing Lin. Deep metric learning with angular loss. In *ICCV*, pages 2612–2620, 2017.

[43] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Formal security analysis of neural networks using symbolic intervals. *CoRR*, abs/1804.10829, 2018.

[44] Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, 2009.

[45] Xi Wu, Uyeong Jang, Jiefeng Chen, Lingjiao Chen, and Somesh Jha. Reinforcing adversarial robustness using model confidence induced by adversarial training. In *ICML*, volume 80, pages 5330–5338, 2018.

[46] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L. Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. *CoRR*, abs/1812.03411, 2018.

[47] Ziang Yan, Yiwen Guo, and Changshui Zhang. Deep defense: Training dnns with improved adversarial robustness. In *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*, pages 417–426, 2018.

[48] Yuzhe Yang, Guo Zhang, Dina Katabi, and Zhi Xu. Me-net: Towards effective adversarial robustness with matrix estimation. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, 2019.

[49] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. *CoRR*, abs/1901.08573, 2019.

[50] Stephan Zheng, Yang Song, Thomas Leung, and Ian J. Goodfellow. Improving the robustness of deep neural networks via stability training. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 4480–4488, 2016.

[51] Wenzhao Zheng, Zhaodong Chen, Jiwen Lu, and Jie Zhou. Hardness-aware deep metric learning. *CoRR*, abs/1903.05503, 2019.

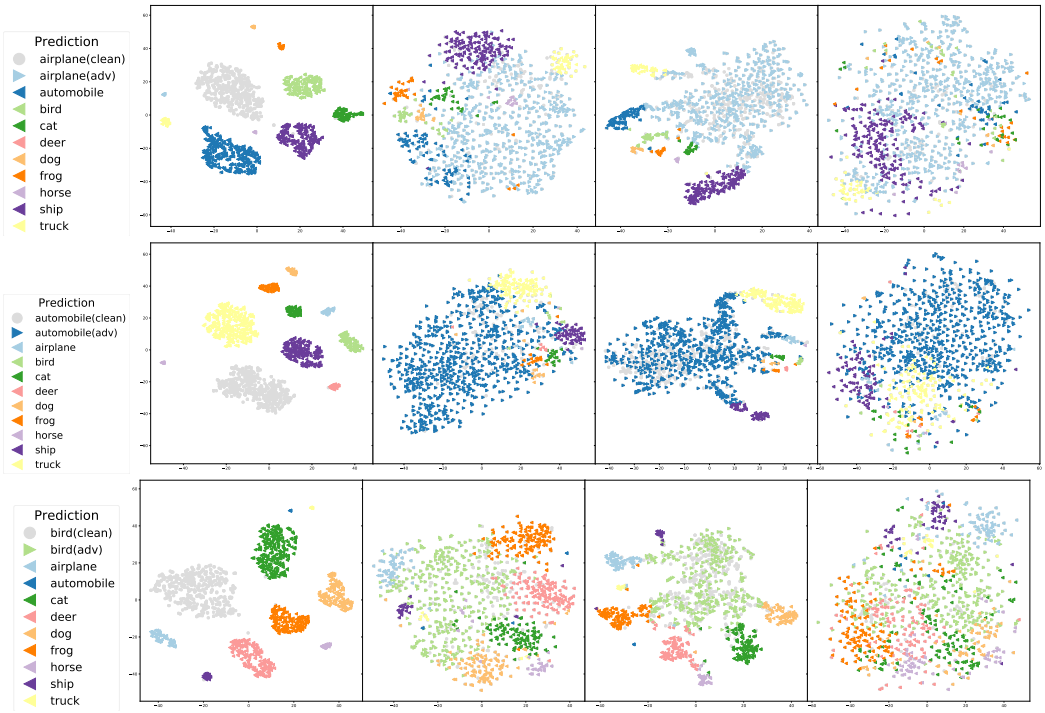
[52] Zhihao Zheng and Pengyu Hong. Robust detection of adversarial attacks by modeling the intrinsic properties of deep neural networks. In *Advances in Neural Information Processing Systems*, pages 7913–7922. 2018.

Supplementary material for "Metric Learning for Adversarial Robustness"

A Indiscrimination of Robust Representation of Adversarial Examples and the True Class

Similar to Fig. 1 in the main text, we visualize the representations of clean and adversarial examples from the same class for the remaining 9 classes on CIFAR-10 dataset across all models using t-SNE [[2]]. Visualizations are shown in Fig. 6, Fig. 7, and Fig. 8.

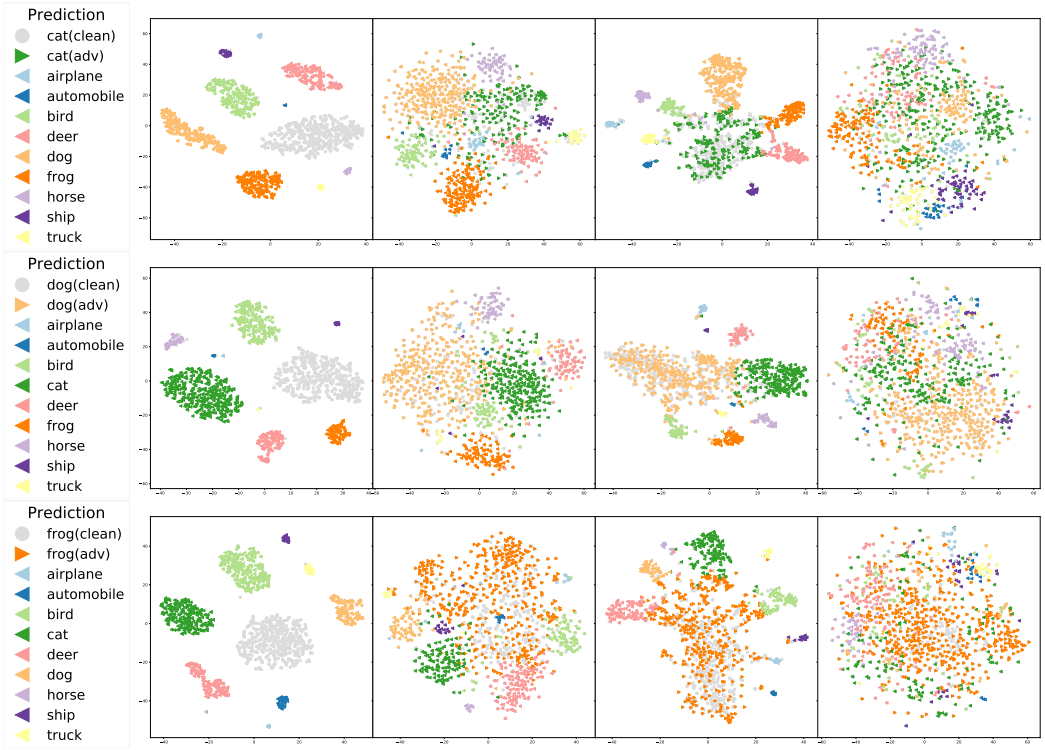
Figure 6: **t-SNE Visualizations of adversarial images from the same *true* class which are mistakenly classified to *false* classes. From left to right: UM, AT, ALP, TLA.** These are representations of second to last layer of 1000 adversarial examples crafted from 1000 clean test examples from CIFAR-10 dataset, where the *true* class is the same for all the figures in the same row and different for figures of different row. The different colors represent different *false* classes. The gray dots further show 500 randomly sampled clean *true* images. Notice that for (a) undefended model (UM), the adversarial attacks clearly separate the images from the *true* category into different classes. (b) adversarial training (AT) and (c) adversarial logit pairing (ALP) method still suffer from this problem at a reduced level. In contrast, proposed ATL (see (d)) clusters together all the examples from the same *true* class, which improves overall robustness.



B Separation of Robust Representations of Adversarial Examples to the False Class

Similar to Fig. 2 in the main text, we provide more visualizations of the representations on CIFAR-10 using t-SNE to demonstrate the separation margin of adversarial samples to the corresponding false class. We plot the representations of adversarial examples from class A which are finally misclassified as class B. We also plot clean images from both A and B. Visualizations are shown in Fig. 9, Fig. 10, and Fig. 11.

Figure 7: **t-SNE Visualizations of adversarial images from the same *true* class which are mistakenly classified to different *false* classes (Same as Fig 6).**



C Experiment

C.1 Implementation Details

We add uniform random noise to the clean images $\mathbf{x}$ within the $\mathbf{I}(\mathbf{x}, \epsilon)$ corresponding to the allowed perturbation scale for the natural images in the TLA training. We conduct all the experiments using a single V100 GPU with 16GB memory. The black-box attack is evaluated Code is appended in the zip file.

MNIST follows the setup of Madry et al. [19] and ALP [16], we use Adam with learning rate of 0.0001. For ALP, we use $\lambda = 0.5$ as suggested in papers [16]. We conduct experiments using our modified LeNet model (adding the batch normalization and replace 5×5 convolution kernel with 3×3). The architecture is shown in Table 7. All experiments are conducted with batch size 50. To be consistent with the results reported by ALP, we maintain the label smoothing with value equals to 0.1. We achieve better accuracy for AT [19] and ALP [16] (The baselines are stronger than the original paper because of the additional label smoothing and batch normalization). We set up the experiment we reported in the table with the following hyper-parameters. For the TLA method, we adopt $\lambda_1 = 0.5$, $\lambda_2 = 0.001$, margin $\alpha = 0.05$, mini-batch size for the negative sample selection as 50. We run the experiments for 200 epochs before it fully converges. We repeat the experiments for five times and observe little oscillations for the performance. We select the one randomly with middle-level performance and conduct all the evaluations above.

CIFAR10 follows the same WRN model as Madry et al [19] across all our models, as shown in Table 8. Also, we adopt the same SGD optimization method with the same learning rate decay strategy as Madry's, where we start with learning rate of 0.1 and decrease it to 0.01 at 50k iterations. We run it for 55k iterations before stopping. We train all the models with a batch size of 50. We implement the ALP on CIFAR-10 because it is not implemented in the original ALP paper, where we do improve the adversarial accuracy significantly. To achieve a fair comparison, we all follow the hyper-parameters set-up in [19]. It took a day and a half before our training converges. We set the $\lambda = 0.5$ for ALP and do not use label smoothing. For TLA method, we adopt $\lambda_1 = 2$, $\lambda_2 = 0.001$, margin $\alpha = 0.03$, mini-batch size for the negative sample selection as 500. Our TLA improves the robust accuracy over ALP baseline for **4.12%** and AT baseline for **4.82%**.

Figure 8: **t-SNE Visualizations of adversarial images from the same *true* class which are mistakenly classified to different *false* classes (Same as Fig 6).**

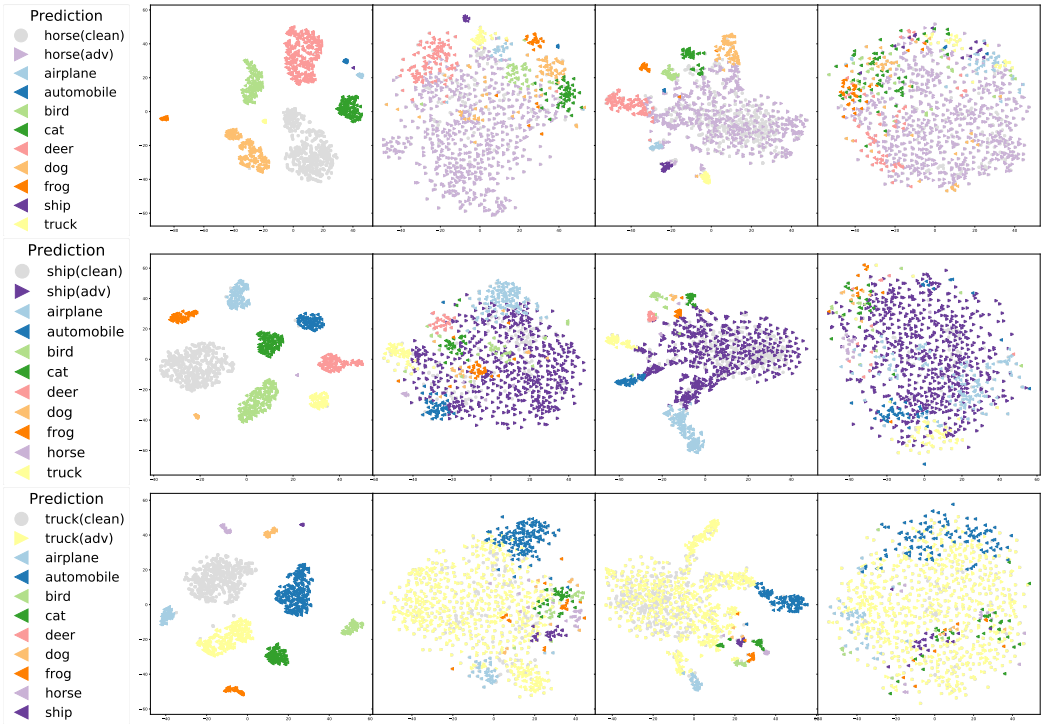
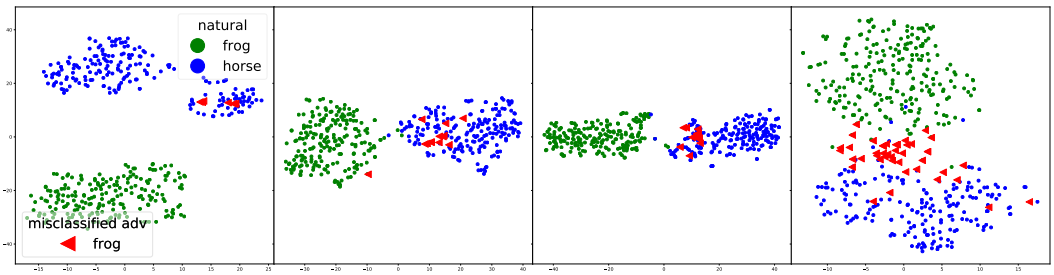


Figure 9: **Illustration of the separation margin of adversarial examples from the natural images of the corresponding false class. From left to right: UM, AT, ALP, TLA.** We show t-SNE visualization of the second to last layer representation of test data from two different classes across four models. The blue and green dots are 200 randomly sampled natural images from "frog" and "horse" classes respectively. The red triangles denote adversarial (adv) perturbed "frog" images but mispredicted as "horse". Notice that for (a) UM, the adversarial examples are moved to the center of the false class which is hard to separate from the natural images of the false class. (b) AT and (c) ALP achieve some robustness by separating adversarial and false natural images, but they are still close to each other. Plot (d) shows proposed TLA promotes the mispredicted adversarial examples to lie on edge and can still be separated from natural images of the false class, which improves the robustness.



Tiny-ImageNet follows the well studied Resnet-50 model. We apply a stride of 2 for the first convolution to reduce the computational intensity. We use the Adam optimizer for AT and ALP trained with batch size of 32. We start from learning rate of 0.1 and decrease the learning rate to 0.01 at the 110-th epochs and 0.001 at the 130th epochs. We train it for 150 epochs. For TLA training, we fintuning on the AT model with batch-size of 20 because of GPU memory budget. We use the untargeted attacks for the adversarial examples generation during both the training and testing procedure, so that it is consistent with the attack conducted in the evaluation. We use data augmentation (crop, flip, saturation, etc.) for all the models. The training time for the models requires about 2 days on a single Nvidia V100 GPU. We set the $\lambda = 0.5$ for ALP. We use label smoothing with parameter equal to 0.1 across all of our experiments. For the TLA method, we adopt $\lambda_1 = 0.2$, $\lambda_2 = 0.001$, margin $\alpha = 0.01$, mini-batch size for the negative sample selection as 50.

Figure 10: **Same as Fig 9 except the two classes are "horse" and "airplane".**

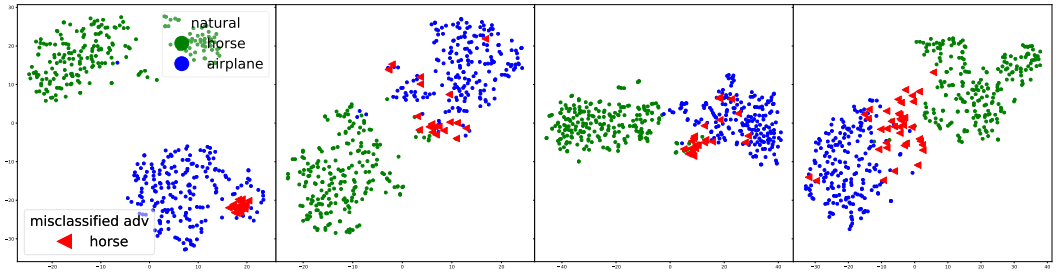
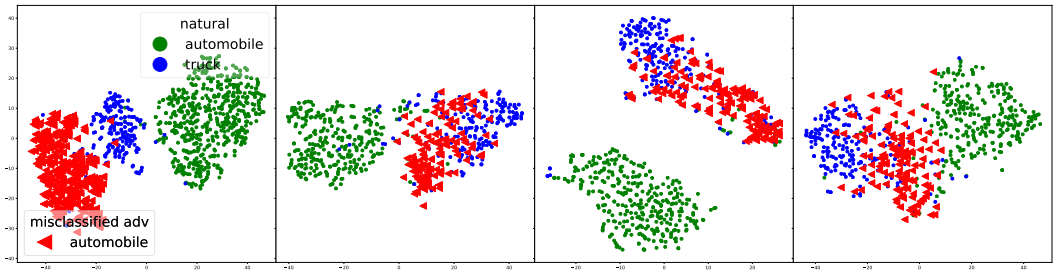


Figure 11: **Same as Fig 9 except the two classes are "horse" and "airplane".**



C.2 Effect of TLA on Bring Adversarial vs. Natural Image of the Same Class Together

We also define a similar metric to show TLA tends to pull closer the adversary images to their true class. For every dataset, we compute a complementary ratio (denoted as r') that measures how adversary images are pulled back to their true class on different models. We reformulate the definition of $\{c_k'^q\}$ to be the embedded representations of all the adversarial examples crafted based on the clean images of true class c_k in the test set. The results are shown in Table 9. Notice that lower value of r' is desirable here, indicating the examples of the same class are pulled together.

Table 7: Illustration of MNIST architecture, which shows all the details of our modified LeNet Architecture by using smaller Convolution (Conv) kernels and batch normalization. Where the Feature-In/Out for the convolution and fully connected (FC) denotes the number of the channel and hidden neurons respectively.

Layer Type	Feature-In Dimension	Feature-Out Dimension	Kernel-Size
Conv	1	32	3×3
BatchNormalization	32	32	-
ReLU	-	-	-
Conv	32	64	3×3
BatchNormalization	64	64	-
ReLU	-	-	-
Max Pooling 2×2			
Conv	64	128	3×3
BatchNormalization	128	128	-
ReLU	-	-	-
Conv	128	256	3×3
BatchNormalization	256	256	-
ReLU	-	-	-
Max Pooling 2×2			
Fully Connected	$7 \times 7 \times 256$	1024	-
BatchNormalization	1024	1024	-
ReLU	-	-	
Fully Connected	1024	10	-
Softmax			

Table 8: Illustration of wide residual network architecture which is the same as Madry et al. [19]. The matrix denotes the set up of the residual block, and $\times$ denotes to repeat this for the following number of times.

layer name	output size	layer setup
Conv	32×32	3×3, 16, stride 1
conv2_x	16×16	$\begin{bmatrix} 3 \times 3, 160 \\ 3 \times 3, 160 \end{bmatrix} \times 6$
conv3_x	8×8	$\begin{bmatrix} 3 \times 3, 320 \\ 3 \times 3, 320 \end{bmatrix} \times 6$
conv4_x	4×4	$\begin{bmatrix} 3 \times 3, 640 \\ 3 \times 3, 640 \end{bmatrix} \times 6$
classifier	1×1	average pool, 10-d fc, softmax

As we can see, adversarial attack tends to bring the representation of an image far away from its true class. For UM, the adversarial examples are far away from the clean examples of the same class. With AT and ALP (baseline) methods, the adversarial examples are getting closer to the clean images of the true class to some extent. Our method TLA brings even closer the adversarial examples to the clean examples on CIFAR-10 and Tiny-ImageNet and achieves comparable performance on MNIST. This further implies that our method promotes the adversarial and clean images from the same class to lie on the same manifold and thus improves the robustness of the model.

Table 9: Average (over all classes) ratio (r') of the mean of pairwise distance between adversary images and natural images of the same class over the mean inner-class distance. The results illustrate that TLA decreases the relative distance of adversarial images w.r.t. the natural images of the respective true classes. The best results of each column are in **bold**.

	Dataset	MNIST		CIFAR-10		Tiny-ImageNet	
	Perturbation Level	$L_\infty = 0.03$	$L_\infty = 0.3$	$L_\infty = \frac{8}{255}$	$L_\infty = \frac{25}{255}$	$L_\infty = \frac{8}{255}$	$L_\infty = \frac{25}{255}$
Methods	UM	1.071	2.159	3.604	3.682	1.319	1.480
	AT	**1.004**	**1.042**	1.342	1.714	1.053	1.204
	ALP	1.006	1.068	2.313	3.796	**1.040**	**1.151**
	TLA	1.005	1.072	**1.191**	**1.491**	1.044	1.174

D TLA Algorithm

The Triplet Loss Adversarial Training (TLA) is introduced in the Algorithm 1. It is a simple approach which can be done within one Loop.

Algorithm 1 Metric Learning for Adversarial Robustness (Triplet Loss Adversarial (TLA) method)

1: **Input:** Data $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N}$, training iterations T_t, learning rate ρ_t, initialized trainable model parameters $\boldsymbol{\theta}$. A minibatch of size K for each iteration is denoted as $\{(\mathbf{X}^{(k)}, Y^{(k)})\}_{k \in \{i_1, \ldots, i_K\}}$.
2: **for** $t = 1 : T_t$ **do**
3: Sample a minibatch of data $\mathbf{X}$ and $\mathbf{X}_{pos}$ of the same class from $\mathcal{D}$
4: Generate adversarial attack images $\mathbf{X}_{adv}$ based on $\mathbf{X}$.
5: Sample a subset of data $\mathbf{X}_{extra}$ and calculate a negative minibatch $\mathbf{X}_{neg}^{-}$corresponding to $\mathbf{X}_{adv}$ with strategy mentioned in section 3.2.
6: Calculate $\mathcal{L}_{all}$ (as defined in Sec 4.2) on the sampled batches.
7: Update parameters: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \rho_t \sum_k \nabla_{\boldsymbol{\theta}} \mathcal{L}_{all}$.
8: **end for**

E The effect of the hyper-parameter

We use MNIST dataset to explore the influence of the hyper-parameters. We conclude that a higher accuracy is usually achieved with a margin between 0.01 to 0.1, the weight λ should be between 0.5 to 2. The results for

different margin and λ plot in the following graph. Overall, our TLA algorithm does not sensitive to the specific hyper-parameters set up. In a wide range, it is able to achieve significant improvement over the baseline models.

Table 10: Adversarial accuracy under 100 steps of PGD attack when model is trained using different λ_1 parameters on MNIST. We conclude that setting the λ_1 within range of 0.5 to 2 is all reasonable.

λ	0	0.025	0.5	1	2	4
TLA	94.82%	96.31%	**96.96%**	96.57%	96.72%	96.26%

Table 11: Adversarial accuracy under 100 steps of PGD attack with $\lambda_1 = 2$ when model is trained using different α parameter on MNIST. The best accuracy is achieved with margin 0.05 according to our experiment.

α	0	0.025	0.05	0.1	0.2
TLA	96.60%	96.47%	**96.72%**	96.36%	96.35%

Table 12: Adversarial robustness accuracy under 100 steps of PGD of model trained on different representation layers with ATL on MNIST. All the models using $\lambda_1 = 2$.The result demonstrate our choice of the second to last layer achieve the best performance.

Representation Layer	Lower	Middle	Higher (Ours)	Logit (ALP)
TLA	96.14%	96.48%	**96.72%**	96.34%

F Visualization

F.1 More Visualization of the Nearest Neighbor Retrival on Learned Embeddings

We show more visualizations of the nearest neighbor retrieval based on the representation learned on different methods. The results are shown in Fig 12.

F.2 Visualization of the Loss Landscape

To demonstrate that our approach does not rely on the obfuscated gradients by having a distorted loss landscape [11], we visualize the loss landscape of the loss function on two random directions in Fig 13.

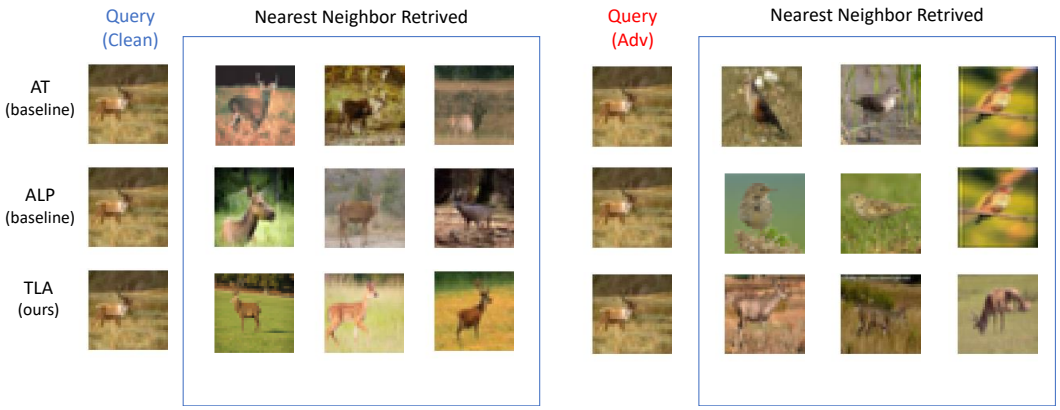


Figure 12: **Visualization of nearest neighbor images while querying about a "deer" on models using AT, ALP, and TLA training separately.** The clean image query is shown on the left column, and the adversarial perturbed image query is shown on the right column. As we can see, while both baseline methods are unable to retrieve the correct nearest neighbors under adversarial attacks, our TLA method (bottom) retrieve the correct images.

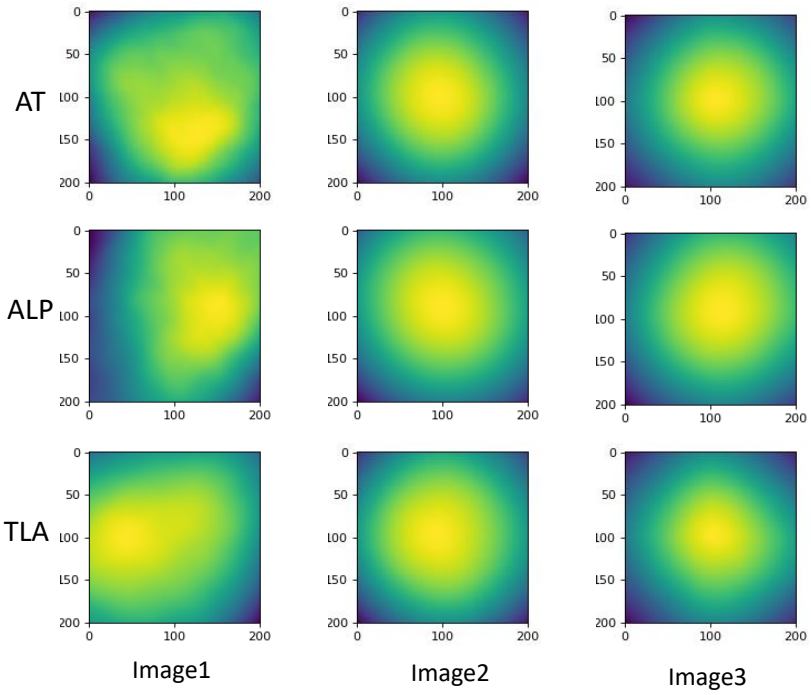


Figure 13: **Visualization of loss landscape of each model of Tiny ImageNet.** We visualize the loss using heatmap of three randomly sampled example (each column has the same direction). For each line, we show the result of baseline methods and our methods. As we can see, TLA (last row) has a slightly smoother loss landscape compared with AT and ALP baselines.