IMPROVING THE GENERALIZATION OF ADVERSARIAL TRAINING WITH DOMAIN ADAPTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

By injecting adversarial examples into training data, the adversarial training method is promising for improving the robustness of deep learning models. However, most existing adversarial training approaches are based on a specific type of adversarial attack. It may not provide sufficiently representative samples from the adversarial domain, leading to a weak generalization ability on adversarial examples from other attacks. To scale to large datasets, perturbations on inputs to generate adversarial examples are usually crafted using fast single-step attacks. This work is mainly focused on the adversarial training with the single-step yet efficient FGSM adversary. In this scenario, it is difficult to train a model with great generalization due to the lack of representative adversarial samples, aka the samples are unable to accurately reflect the adversarial domain. To address this problem, we propose a novel Adversarial Training with Domain Adaptation (ATDA) method by regarding the adversarial training with FGSM adversary as a domain adaption task with limited number of target domain samples. The main idea is to learn a representation that is semantically meaningful and domain invariant on the clean domain as well as the adversarial domain. Empirical evaluations demonstrate that ATDA can greatly improve the generalization of adversarial training and achieves state-of-the-art results on standard benchmark datasets.

1 INTRODUCTION

Deep learning techniques have shown impressive performance on image classification and many other computer vision tasks, however, recent works have revealed that deep learning models are often vulnerable to adversarial examples (Szegedy et al., 2014; Goodfellow et al.; Papernot et al., 2016), which are maliciously designed to deceive the target model by generating carefully crafted adversarial perturbations on original clean inputs. Moreover, adversarial examples can transfer across models to mislead other models with a high probability (Papernot et al., 2017; Liu et al., 2017). How to effectively defense against adversarial attacks is crucial for security-critical computer vision systems, such as autonomous driving.

One promising approach, adversarial training, defends from adversarial perturbations by training a target classifier with adversarial examples. Researchers have found (Goodfellow et al.; Kurakin et al., 2016b; Madry et al., 2018) that adversarial training could increase the robustness of neural networks. However, adversarial training often obtain adversarial examples by taking a specific attack into consideration, so the defense is specific to such attack and the trained model has weak generalization ability on adversarial examples from other attacks. Tramèr et al. (2018) showed that the robustness of adversarial training can be easily circumvented by the attack that combines with random perturbation from other models. Thus, for most existing adversarial training methods, there is a risk of overfitting to adversarial examples crafted on the original model with the specific attack.

In this paper, we propose a novel adversarial training method that is able to improve the generalization of adversarial training. From the perspective of domain adaptation (DA) (tor, 2011), there is a big domain gap between the distribution of clean examples and the distribution of adversarial examples in the high-level representation space, even though adversarial perturbations are imperceptible to humans. Liao et al. (2018) showed that adversarial perturbations are progressively amplified along the layer hierarchy of neural networks, which maximizes the distance between the original and adversarial subspace representations. In addition, adversarial training simply injects adversarial

examples from a specific attack into the training set, but there is still a large sample space for adversarial examples. Thus, training with the classification loss on such a training set will probably lead to overfitting on the adversarial examples from the specific attack. Even though Wong & Kolter (2018) showed that adversarial training with iterative noisy attack has better robustness than the adversarial training with single-step attack, the iterative attack is computationally expensive and there is no theoretical analysis to justify the adversarial examples sampled in such way are sufficiently representative for adversarial domain.

The key idea of our approach is to formulate the learning procedure as a domain adaptation problem with limited number of target domain samples, where target domain denotes adversarial domain. Specifically, we introduce unsupervised as well as supervised domain adaptation into adversarial training to minimize the gap and increase the similarity between the distributions of clean examples and adversarial examples. In this way, the learned classifier generalizes well on adversarial examples from different attacks. We evaluate our ATDA method on standard benchmark datasets. Empirical results show that despite a small decay of accuracy on clean data, ATDA significantly improves the generalization ability of adversarial training. In addition, as compared with adversarial training with iterative attack method that is computationally expensive and only runs on small data, our method is fast and it can handle big data like CIFAR-100 with a deep network having seven convolutional layers within reasonable time.

2 Background and Related Work

This section will introduce some notations and provide a brief overview of current advanced attack methods, and defenses based on adversarial training.

2.1 Notation

Denote the clean data domain and adversarial data domain by $\mathcal{D}$ and $\mathcal{A}$ respectively, we consider a classifier based on a neural network $f(x) : \mathbb{R}^d \rightarrow \mathbb{R}^k$. $f(x)$ outputs the probability distribution for input $x \in [0,1]^d$, and k denotes number of classes in the classification task. Let ϕ be the mapping at the logits layer (the last neural layer before the final softmax function), so that $f(x) = softmax(\phi(x))$. Let ϵ be the ℓ_∞ norm of the perturbation. Let x^{adv} be the adversarial image computed by perturbing the original image x. The cost function of image classification is denoted as $J(x, y)$. We define the *logits* as the logits layer representation, and define the *logit space* as the semantic space of the logits representation.

There are two types of attacks. White-box attacks have the complete knowledge of the target model and can fully access the model. Black-box attacks have limited knowledge of the target classifier (e.g. its architecture) and can not access the model weights.

2.2 Attack Methods

We consider four attack methods to generate adversarial examples. For all attacks, the components of adversarial examples are clipped in $[0, 1]$.

Fast Gradient Sign Method (FGSM). Goodfellow et al. introduced the Fast Gradient Sign Method (FGSM) to generate adversarial examples by applying perturbations in the direction of the gradient.

$$x^{adv} = x + \epsilon \cdot \text{sign}(\nabla_x J(x, y_{true})) \tag{1}$$

As compared to other attack methods, FGSM is simple and efficient, so it is particularly amenable to adversarial training.

Basic Iterative Method (BIM). Kurakin et al. (2016a) introduced an iterative extension of FGSM, called the Basic Iterative Method (BIM). This method applies FGSM iteratively for k times with a budget $\alpha = \epsilon/k$ instead of a single step.

$$\begin{aligned} x^{adv_0} &= x \\ x^{adv_{t+1}} &= x^{adv_t} + \alpha \cdot \text{sign}(\nabla_x J(x^{adv_t}, y_{true}) \\ x^{adv} &= x^{adv_k} \end{aligned} \tag{2}$$

BIM usually yields a higher success rate than FGSM does in white-box attacks but shows weaker capability in black-box attacks.

RAND+FGSM (R+FGSM). Tramèr et al. (2018) proposed R+FGSM against adversarially trained models by applying a small random perturbation of step size α before applying FGSM.

$$\begin{aligned} x' &= x + \alpha \cdot \text{sign}(\mathcal{N}(\mathbf{0}^d, \mathbf{I}^d)) \\ x^{adv} &= x' + (\epsilon - \alpha) \cdot \text{sign}(\nabla_x J(x^{adv}, y_{true}) \end{aligned} \tag{3}$$

Momentum Iterative Method (MIM). MIM (Dong et al., 2018) is a modification of the iterative FGSM and it won the first place of NIPS 2017 Adversarial Attacks Competition. Its basic idea is to utilize the gradients of the previous t steps with a decay factor μ to update the gradient at step $t+1$ before applying FGSM with budget α.

$$\begin{aligned} x^{adv_0} &= x,\ g_0 = 0 \\ g_{t+1} &= \mu \cdot g_t + \frac{\nabla_x J(x^{adv_t}, y_{true})}{\|\nabla_x J(x^{adv_t}, y_{true})\|_1} \\ x^{adv_{t+1}} &= x^{adv_t} + \alpha \cdot \text{sign}(g_{t+1}) \\ x^{adv} &= x^{adv_k} \end{aligned} \tag{4}$$

2.3 Progress on adversarial training

An intuitive technique for defending a deep model against adversarial examples is adversarial training, which injects adversarial examples into the training data during the training process. First, Goodfellow et al. propose to increase the robustness of a model by feeding the model with both original and adversarial examples generated by FGSM and by learning with the modified objective function.

$$\hat{J}(x, y_{true}) = \alpha J(x, y_{true}) + (1-\alpha) J(x + \epsilon\, \text{sign}(\nabla_x J(x, y_{true}), y_{true}) \tag{5}$$

Kurakin et al. (2016b) scaled the adversarial training to ImageNet (Russakovsky et al., 2015) and show better results by replacing half the clean example at each batch with the corresponding adversarial examples. Meanwhile, Kurakin et al. (2016b) discovered the *label leaking* effect and suggest not to use the FGSM defined with respect to the true label y_{true}. However, their approach is not robust against the RAND+FGSM adversary. Tramèr et al. (2018) propose an *ensemble adversarial training* (ENS) to improve robustness on black-box attacks by injecting adversarial examples transferred from a number of fixed pre-trained models into training data. Another approach is to train only with adversarial examples. Nøkland (2015) proposes a specialization of the method (Goodfellow et al.) that learns only with the objective function of adversarial examples. Madry et al. (2018) demonstrate successful defenses based on adversarial training with iterative noisy attack. However, their approach is fairly expensive to scale the process to large-scale neural networks as the iterative attack has a large cost of computing time during the training. Wong & Kolter (2018) developed a robust training method by linear programming that minimizes the loss for the worst case within the perturbation ball around each clean data point. However, their approach achieves high test error on clean data and it is still challenging to scale to deep or wide neural networks.

As described above, though adversarial training is promising, it is difficult to select a simple and efficient attack method to train on and most existing methods are weak in generalization for various attacks because the region of the adversarial examples for each clean data is large and contiguous (Tramèr et al., 2017; Tabacof & Valle, 2016).

3 Adversarial Training with Domain Adaptation

In this work, instead of focusing on a better sampling strategy to obtain representative adversarial data from adversarial domain, we are especially concerned with the problem of how to train with clean data and adversarial examples from the efficient FGSM, so that the adversarially trained model is strong in generalization for different attacks and has a low cost of computing time for the training.

We propose an Adversarial Training with Domain Adaptation (ATDA) method to defense adversarial attacks and expect the learned model generalizes well for various adversarial examples. Our motivation is to treat the adversarial training with FGSM as a domain adaptation task with limited

number of target domain samples, where the target domain denotes adversarial domain. We combine the original adversarial training with the domain adaptor, which minimizes the domain gap between clean examples and adversarial examples. In this way, our adversarially trained model is effective on adversarial examples crafted by FGSM but also shows great generalization on different attacks.

3.1 Domain Adaptation on Logit Space

3.1.1 Unsupervised domain adaptation

Suppose we are given some clean training examples $\{x_i\}$ ($x_i \in \mathbb{R}^d$) with labels $\{y_i\}$ from the clean data domain $\mathcal{D}$, and adversarial examples $\{x_i^{adv}\}$ ($x_i^{adv} \in \mathbb{R}^d$) from adversarial data domain $\mathcal{A}$. The adversarial examples are obtained by sampling (x_i, y_{true}) from $\mathcal{D}$, computing small perturbations on x_i to generate adversarial perturbations, and outputting (x_i^{adv}, y_{true}).

It's known that there is a huge shift in the distributions of clean data and adversarial data in high-level representation space. Assume that in the logit space, examples from either clean domain or adversarial domain follow a multivariate normal distribution, i.e., $\mathcal{D} \sim \mathcal{N}(\mu_{\mathcal{D}}, \Sigma_{\mathcal{D}})$, $\mathcal{A} \sim \mathcal{N}(\mu_{\mathcal{A}}, \Sigma_{\mathcal{A}})$. Our goal is to learn the logits representation that minimizes the domain shift by aligning the covariance matrix and the mean vector of the clean distribution and the adversarial distribution.

To implement the CORrelation ALignment (CORAL), we define a covariance distance between the clean data and the adversarial data as follows:

$$\mathcal{L}_{CORAL}(\mathcal{D}, \mathcal{A}) = \frac{1}{k^2} \left\| C_{\phi(\mathcal{D})} - C_{\phi(\mathcal{A})} \right\|_{\ell_1} \tag{6}$$

where $C_{\phi(\mathcal{D})}$ and $C_{\phi(\mathcal{A})}$ are the covariance matrices of the clean data and the adversarial data in the logit space respectively, and $\| \cdot \|_{\ell_1}$ denotes the L_1 norm of a matrix. Note that $\mathcal{L}_{CORAL}(\mathcal{D}, \mathcal{A})$ is slightly different from the CORAL loss proposed by Sun & Saenko (2016).

Similarly, we use the standard distribution distance metric, Maximum Mean Discrepancy (MMD) (Borgwardt et al., 2006), to minimize the distance of the mean vectors of the clean data and the adversarial data.

$$\mathcal{L}_{MMD}(\mathcal{D}, \mathcal{A}) = \frac{1}{k} \left\| \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{A}} \phi(x) - \frac{1}{|\mathcal{A}|} \sum_{x^{adv} \in \mathcal{A}} \phi(x^{adv}) \right\|_1 \tag{7}$$

The loss function for unsupervised domain adaptation (UDA) can be calculated as follows.

$$\mathcal{L}_{UDA}(\mathcal{D}, \mathcal{A}) = \mathcal{L}_{CORAL}(\mathcal{D}, \mathcal{A}) + \mathcal{L}_{MMD}(\mathcal{D}, \mathcal{A}) \tag{8}$$

3.1.2 Supervised domain adaptation

Even though the unsupervised domain adaptation achieves perfect confusion alignment, there is no guarantee that samples of the same label from clean domain and adversarial domain would map nearby in the logit space. To effectively utilize the labeled data in the adversarial domain, we introduce a supervised domain adaptation (SDA) by proposing a new loss function, denoted as margin loss, to minimize the intra-class variations and maximize the inter-class variations on samples of different domains. The SDA loss is shown in Eq. (9).

$$\begin{aligned} \mathcal{L}_{SDA}(\mathcal{D}, \mathcal{A}) &= \mathcal{L}_{margin}(\mathcal{D}, \mathcal{A}) \\ &= \frac{1}{(k-1)(|\mathcal{D}| + |\mathcal{A}|)} \cdot \\ &\sum_{x \in \mathcal{D} \cup \mathcal{A}} \sum_{c^n \in C \setminus \{c_{y_{true}}\}} softplus(\|\phi(x) - c_{y_{true}}\|_1 - \|\phi(x) - c^n\|_1) \end{aligned} \tag{9}$$

Here $softplus$ denotes a function $ln(1 + exp(\cdot))$; $c_{y_{true}} \in \mathbb{R}^k$ denotes the center of y_{true} class in the logit space; $C = \{ c_j \mid j = 1, 2, ..., k\}$ is a set consisting of the logits center for each class, which will be updated as the logits changed. Similar to the center loss (Wen et al., 2016), we update center c_j for each class j:

$$\begin{aligned} \Delta c_j^t &= \frac{\sum_{x \in \mathcal{D} \cup \mathcal{A}} \mathbf{1}_{\mathrm{y_{true}=j}} \cdot (c_j^t - \phi(x))}{1 + \sum_{x \in \mathcal{D} \cup \mathcal{A}} \mathbf{1}_{\mathrm{y_{true}=j}}} \\ c_j^{t+1} &= c_j^t - \alpha \cdot \Delta c_j^t \end{aligned} \tag{10}$$

where $\mathbf{1}_{\text{condition}} = 1$ if the *condition* is true, otherwise $\mathbf{1}_{\text{condition}} = 0$; α denotes the learning rate of the centers. In the training process, the logits center for each class can integrate supervised information from both the clean domain and the adversarial domain. We set $\alpha = 0.1$ to stabilize the training process.

3.2 Adversarial Training

For adversarial training, iterative attacks are fairly expensive to compute and single-step attacks are fast to compute. Thus, we use a variant of FGSM attack (Kurakin et al., 2016b) that avoids the *label leaking* effect to generate a new adversarial example x_i^{adv} for each clean example x_i:

$$x_i^{adv} = x_i + \epsilon \cdot \text{sign}(\nabla_x J(x_i, y_{target})) \tag{11}$$

where y_{target} denotes the predicted class $\arg\max\{\varphi(x_i)\}$ of the model.

However, in this case, the sampled adversarial examples are aggressive but not sufficiently representative due to the fact that the sampled adversarial examples always lie at the boundary of the ℓ_∞ ball of radius ϵ (see Figure 1) and the adversarial examples within the boundary are ignored. For adversarial training, if we train a deep neural network only on the clean data and the adversarial data from the FGSM attack, the adversarially trained model will overfit on these two kinds of data and exhibits weak generalization ability on the adversarial examples sampled from other attacks. From a different perspective, such problem can be viewed as a domain adaptation problem with limited number of labeled target domain samples, as only some special data point can be sampled in the adversarial domain by the FGSM attack.

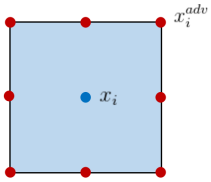


Figure 1: Illustration of adversarial sampling by the FGSM attack for $x_i \in \mathbb{R}^2$. The blue dot (in the center) represents clean data and the red dots (along the boundary) represent adversarial data.

Consequently, it is natural to combine the adversarial training with domain adaptation to improve the generalization ability on adversarial data. We generate new adversarial examples by a variant of FGSM attack, as shown in Eq. (11). Then we use the following loss function to meet the criteria of domain adaptation while training a strong classifier.

$$\begin{aligned} Loss = &\ \mathcal{L}_C(\mathcal{D}) + \mathcal{L}_C(\mathcal{A}) + \lambda \cdot \mathcal{L}_{DA}(\mathcal{D}, \mathcal{A}) \\ = &\ \mathcal{L}_C(\mathcal{D}) + \mathcal{L}_C(\mathcal{A}) + \lambda \cdot (\mathcal{L}_{UDA}(\mathcal{D}, \mathcal{A}) + \mathcal{L}_{SDA}(\mathcal{D}, \mathcal{A})) \\ = &\ \frac{1}{m} \sum_{x \in \mathcal{D}} \mathcal{L}_C(x|y_{true}) + \frac{1}{m} \sum_{x^{adv} \in \mathcal{A}} \mathcal{L}_C(x^{adv}|y_{true}) \\ &+ \lambda \cdot (\mathcal{L}_{CORAL}(\mathcal{D}, \mathcal{A}) + \mathcal{L}_{MMD}(\mathcal{D}, \mathcal{A}) + \mathcal{L}_{margin}(\mathcal{D}, \mathcal{A})) \end{aligned} \tag{12}$$

Here λ is to balance the regularization term; m is the number of input clean examples; $\mathcal{D}$ indicates the input clean examples $\{x_i\}$, and $\mathcal{A}$ the corresponding adversarial examples $\{x_i^{adv}\}$; L_C denotes the classification loss.

The training process is summarized in Algorithm 1. Note that the adversarial training process with $\mathcal{L}_C$ and $\mathcal{L}_{margin}$, $\mathcal{L}_{CORAL}$ and $\mathcal{L}_{MMD}$ can be seen as supervised learning and unsupervised learning, respectively. We combine these two types of learning methods to improve the generalization ability of adversarial training.

4 Experimental Results

To demonstrate the effectiveness of the proposed method, in conjunction with different white-box and black-box attacks with bounded ℓ_∞ norm, we consider four popular datasets, namely Fashion-MNIST (Xiao et al., 2017), Street View House Numbers (SVHN) (Netzer et al., 2011), CIFAR-10 and CIFAR-100 (Krizhevsky & Hinton, 2009). For all experiments, we normalize the pixel value in $[0, 1]$ by dividing 255. Code for these experiments is available at `http://github.com/cxmscb/ATDA`.

Algorithm 1 Adversarial training with domain adaptation on network $f(x) : \mathbb{R}^d \rightarrow \mathbb{R}^k$.
Parameters: Size of the training minibatch is m.

Randomly initialize network $f(x)$ and logits centers $\{c_j \mid j = 1, 2, ..., k\}$;
Number of iterations $t \leftarrow 0$;
repeat
  $t \leftarrow t + 1$;
  Read a minibatch of data $\mathcal{D}_b = \{x_1, ..., x_m\}$ from the training set;
  Use the current state of network f to generate adversarial examples $\mathcal{A}_b = \{x_1^{adv}, ..., x_m^{adv}\}$ from by the FGSM variant that avoids label leaking;
  Extract logits for examples $\mathcal{D}_b$, $\mathcal{A}_b$ by performing forward-backward propagation from the input layer to the logits layer $\phi(x)$;
  Update parameters c_j for each class j by $c_j^{t+1} = c_j^t - 0.1 \cdot \Delta c_j^t$;
  Compute the loss by Eq. (12) and update parameters of network f by back propagation;
until the training converges.

To evaluating the generalization power of the learned models on adversarial examples in both white-box and black-box settings, we report the clean test error, the standard errors achieved by FGSM, BIM, R+FGSM and MIM in the non-targeted way. The common settings for these attacks are shown in Table 1. We compare our ATDA method with normal training method as well as several adversarial training method:

- Normal Training (NT). Training with cross-entropy loss on the clean training data.
- Original Adversarial Training (OAT) (Goodfellow et al.). Training with the cross-entropy on clean training data and the adversarial examples from the FGSM variant that avoids label leaking with perturbation ϵ.
- Ensemble Adversarial Training (ENS) (Tramèr et al., 2018). Training with cross-entropy on clean training data and the adversarial examples crafted from the currently trained model and the static pre-trained models by the FGSM variant that avoids label leaking with the perturbation ϵ.
- Provably Robust Training (PRT) (Wong & Kolter, 2018). Training with cross-entropy loss on the worst case in the ℓ_∞ ball of radius ϵ around each clean training data point.

To have a fair comparison with the above methods, we conduct experiments using the source code provided by the authors. For all experiments, we set the hyper-parameter λ in Eq. (12) to $1/3$. All experiments are implemented on a single Titan X GPU.

Table 1: **Common settings of attacks for all experiments**

Attack	Parameter	Norm
FGSM	N/A	ℓ_∞
BIM	Iterated step $k = 10$, $\alpha = \epsilon/k$	ℓ_∞
R+FGSM	Random perturbation $\alpha = \epsilon/2$	ℓ_∞
MIM	Iterated step $k = 10,\ \alpha = \epsilon/5,\ \mu = 1.0$	ℓ_∞

4.1 Evaluation on Fashion-MNIST

We train a normal model and several adversarial models with perturbation $\epsilon = 0.1$ on a main model with ConvNet architecture, shown in Table 2. For the method of Ensemble Adversarial Training (ENS), we use two different models as static pre-trained models. For black-box attacks, we test trained models on the adversarial examples transferred from a model held out during the training.

In the training phase, we use Adam optimizer with a learning rate of 0.001 and set batch size to 64. The classification accuracies on the Fashion-MNIST dataset are summarized in Table 3. For the normal training method, it yields the best performance on clean data, but generalizes poorly on adversarial examples. The Original Adversarial Training (OAT) method and the Ensemble Adversarial Training (ENS) method overfit on the clean data and the adversarial data from the FGSM attack. The Provably Robust Training (PRT) method achieves lower error against attacks, but higher error

on clean test data. Overall, our learned model achieves higher robustness against different attacks, especially the R+FGSM attack, despite a slightly lower performance on clean data.

4.2 EVALUATION ON SVHN

SVHN is a real-world image dataset for recognizing digits and numbers in natural scene images. We train a normal model and various adversarial models with perturbation $\epsilon = 0.02$ and use the same architectures and settings as in Fashion-MNIST.

In the training phase, we use Adam optimizer with a learning rate of 0.001 and set batch size to 24. The classification accuracies on the SVHN dataset are summarized in Table 4. The proposed method achieves higher generalization ability on adversarial examples from various attacks, at the same time it only loses a small amount of accuracy on clean data.

Table 2: **Neural network architectures used for the Fashion-MNIST and SVHN datasets**. Conv: convolutional layer with Relu, FC: fully connected layer.

Main model	Pre-trained $model_A$	Pre-trained $model_B$	Holdout model
Conv(16, 4x4)	Conv(32, 5x5)	Dropout(0.2)	Conv(64, 3x3)
Conv(32, 4x4)	Conv(32, 5x5)	Conv(32, 3x3)	FC(300) + Relu
FC(100) + Relu	Dropout(0.1)	Conv(32, 3x3)	Dropout(0.5)
FC(10)	FC(128) + Relu	FC(128) + Relu	FC(300) + Relu
	Dropout(0.5)	Dropout(0.5)	Dropout(0.5)
	FC(10)	FC(10)	FC(10)

Table 3: **The classification accuracy on testing sets obtained by different defenses against various attacks on Fashion-MNIST dataset.** The magnitude of perturbations are 0.1 in ℓ_∞ norm.

Defense	Clean (%)	White-Box Attack (%)				Black-Box Attack (%)			
		FGSM	BIM	R+FGSM	MIM	FGSM	BIM	R+FGSM	MIM
NT	90.5	8.3	0.5	15.0	0.1	52.1	60.1	68.2	44.3
OAT	**90.9**	88.8	13.9	31.2	9.4	79.8	80.0	81.8	80.0
ENS	90.8	**89.0**	12.8	31.6	6.6	80.8	83.1	82.3	78.8
PRT	76.9	67.4	66.9	72.2	66.7	75.5	75.5	76.4	75.4
ATDA	85.5	78.2	**70.2**	**77.0**	**68.8**	**83.8**	**83.8**	**84.5**	**83.3**

Table 4: **The classification accuracy on testing sets obtained by different defenses against various attacks on SVHN dataset.** The magnitude of perturbations are 0.02 in ℓ_∞ norm.

Defense	Clean (%)	White-Box Attack (%)				Black-Box Attack (%)			
		FGSM	BIM	R+FGSM	MIM	FGSM	BIM	R+FGSM	MIM
NT	84.9	19.6	7.0	33.3	4.6	64.3	69.9	76.5	64.8
OAT	86.6	52.1	47.5	70.4	46.1	79.0	**80.5**	83.3	78.7
ENS	**88.6**	47.1	38.8	67.6	36.6	**80.4**	80.2	**85.3**	**80.2**
PRT	58.5	41.7	41.3	49.5	41.2	53.7	54.7	56.1	54.0
ATDA	82.9	**57.2**	**54.7**	**70.6**	**53.9**	75.3	77.1	79.5	75.4

4.3 MORE COMPARISONS

CIFAR-10. Compared to SVHN, CIFAR-10 is a more difficult dataset for the classification. We train a normal model and different adversarial models with $\epsilon = 4/255$ on the architectures depicted in Table 7 in Appendix A. To enhance the expressive power of deep neural networks, we use Exponential Linear Unit (ELU) (Clevert et al., 2015) as the activation function and introduce Group Normalization (Wu & He, 2018) into the architectures.

In the training phase, we use Adam optimizer with a learning rate of 0.001 and set batch size to 32. For the PRT method, it is challenging and expensive to scale to large deep neural networks due to the complexity. Thus, the results of PRT are not reported. The classification accuracies on CIFAR-10 are summarized in Table 5. Our proposed method outperforms all the competing methods on most attack methods.

Table 5: **The classification accuracy on testing sets obtained by different defenses against various attacks on CIFAR-10 dataset.** The magnitude of perturbations are $4/255$ in ℓ_∞ norm.

Defense	Clean (%)	White-Box Attack (%)				Black-Box Attack (%)			
		FGSM	BIM	R+FGSM	MIM	FGSM	BIM	R+FGSM	MIM
NT	**86.9**	4.3	1.1	19.6	0.9	67.0	65.6	77.8	64.4
OAT	86.2	52.4	50.7	70.2	50.5	**80.4**	**80.6**	**83.6**	80.3
ENS	85.3	45.4	43.1	66.7	43.0	79.1	79.3	82.5	79.0
PRT	-	-	-	-	-	-	-	-	-
ATDA	84.8	**60.7**	**59.1**	**73.2**	**59.0**	**80.4**	**80.6**	82.8	**80.4**

Table 6: **The classification accuracy on testing sets obtained by different defenses on various attacks on CIFAR-100 dataset.** The magnitude of perturbations are $4/255$ in ℓ_∞ norm.

Defense	Clean (%)	White-Box Attack (%)				Black-Box Attack (%)			
		FGSM	BIM	R+FGSM	MIM	FGSM	BIM	R+FGSM	MIM
NT	59.0	1.7	0.4	6.1	0.4	52.8	54.9	56.2	52.7
OAT	58.7	19.8	18.0	34.8	17.9	56.9	57.2	57.8	56.8
ENS	58.6	15.1	13.5	31.1	13.4	56.6	57.0	57.5	56.6
PRT	-	-	-	-	-	-	-	-	-
ATDA	**61.2**	**28.8**	**27.2**	**43.0**	**27.0**	**59.5**	**59.8**	**60.3**	**59.5**

CIFAR-100. The CIFAR-100 dataset contains 100 image classes, with 600 images per class. We train a normal model and different adversarial models with $\epsilon = 4/255$ on the architectures shown in Table 8 in Appendix B, which are similar to the architectures in CIFAR-10. The goal here is not to achieve state of the art performance on CIFAR-100 dataset, but to compare the generalization ability of different training methods on large dataset.

In the training phase, we use Adam optimizer with a learning rate of 0.001 and set the batch size to 32. The classification accuracies on CIFAR-100 are summarized in Table 6. Compared to other training methods, our method achieves higher performance on clean data and exhibits better generalization on adversarial examples from various attacks. It is interesting that we do not observe the trade-off between generalization and accuracy again.

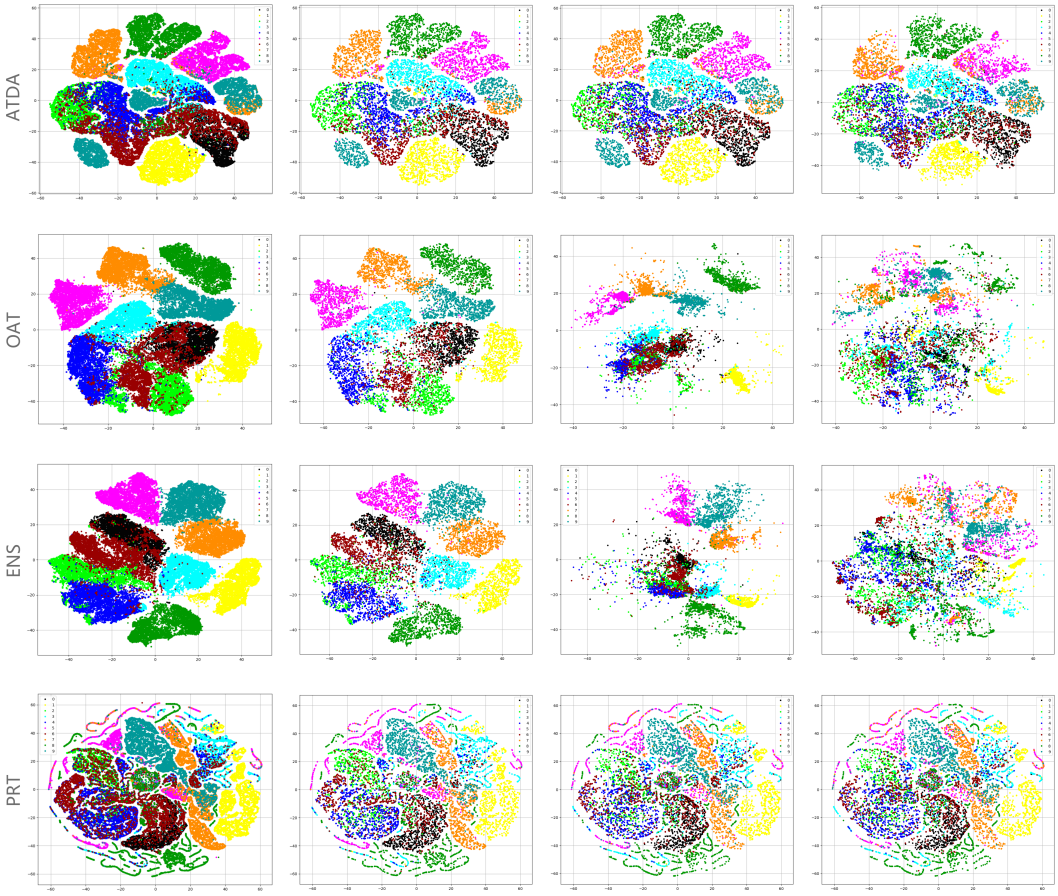
4.4 Discussion

To have a better understanding on the proposed method, we compare our learned logits embeddings with the logits embeddings of the competing adversarial training methods on Fashion-MNIST, through t-SNE (Maaten & Hinton, 2008) for the training data, testing data and adversarial data from the FGSM or BIM attacks. The comparisons are illustrated in Figure 2. It shows that our learned logits representation could minimize the distance between the clean domain and the adversarial domain, and exhibits better performance on domain invariance.

It is interesting that the logits embeddings of PRT show that it can learn domain invariance between the clean domain and the adversarial domain by training with the worst case in the perturbation ball. Maybe there are unknown connection between adversarial training with domain adaptation and adversarial training with the worst case in the perturbation ball, which deserves to be further explored in future work. To sum up, our ATDA method has better performance on both scalability and generalization ability.

5 Conclusion

To overcome the weak generalization problem in adversarial training for adversarial examples from different attacks, we regard the adversarial training as a domain adaptation task with limited number of target labeled data. By combining unsupervised and supervised domain adaptation with adversarial training, the generalization ability on adversarial examples from various attacks can be highly improved for robust defense. The experimental results on several benchmark datasets suggest that the proposed approach achieves significantly better generalization results when compared to current competing adversarial training methods.



(a) training data (b) testing data (c) FGSM adversarial data (d) BIM adversarial data

Figure 2: t-SNE visualizations for the embeddings of training data, testing data, and adversarial data from FGSM and BIM in the logits space for Fashion-MNIST. The first row to the forth row correspond to ATDA, OAT, ENS and PRT, respectively

REFERENCES

The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011, 2011. IEEE Computer Society.

Karsten M Borgwardt, Arthur Gretton, Malte J Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alex J Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14):e49–e57, 2006.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples (2014). *arXiv preprint arXiv:1412.6572*.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016a.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016b.

Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Jun Zhu, and Xiaolin Hu. Defense against adversarial attacks using high-level representation guided denoiser. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1778–1787, 2018.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. In *International Conference on Learning Representations(ICLR)*, 2017.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations(ICLR)*, 2018.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, pp. 5, 2011.

Arild Nøkland. Improving back-propagation by adding an adversarial gradient. *CoRR*, abs/1510.04189, 2015.

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pp. 372–387, 2016.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pp. 506–519, 2017.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European Conference on Computer Vision*, pp. 443–450, 2016.

Christian Szegedy, Google Inc, Wojciech Zaremba, Ilya Sutskever, Google Inc, Joan Bruna, Dumitru Erhan, Google Inc, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations(ICLR)*, 2014.

Pedro Tabacof and Eduardo Valle. Exploring the space of adversarial images. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pp. 426–433. IEEE, 2016.

Florian Tramèr, Nicolas Papernot, Ian J. Goodfellow, Dan Boneh, and Patrick D. McDaniel. The space of transferable adversarial examples. *CoRR*, abs/1704.03453, 2017.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations*, 2018.

Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European Conference on Computer Vision*, pp. 499–515, 2016.

Eric Wong and J. Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 5283–5292, 2018.

Yuxin Wu and Kaiming He. Group normalization. *arXiv preprint arXiv:1803.08494*, 2018.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

A NEURAL NETWORK ARCHITECTURES FOR CIFAR-10

We use the following neural network architectures on CIFAR-10 for the main model, the static pre-trained models and the model held out during training.

Table 7: **Neural network architectures used for the CIFAR-10 dataset**. Conv: convolutional layer with Group Normalization and ELU; GAP: global average pooling.

Main model	Pre-trained $model_A$	Pre-trained $model_B$	Holdout model
Conv(96, 3x3)	Conv(96, 5x5)	Conv(192, 5x5)	Conv(96, 3x3)
Conv(96, 3x3)	Conv(96, 1x1)	Conv(96, 1x1)	Dropout(0.2)
Conv(96, 3x3)	MaxPooling(3x3, 2)	MaxPooling(3x3, 2)	Conv(96, 3x3) x 2
Dropout(0.5)	Dropout(0.5)	Dropout(0.5)	Dropout(0.5)
Conv(192, 3x3) x 3	Conv(192, 5x5)	Conv(192, 5x5)	Conv(192, 3x3) x 2
Dropout(0.5)	Conv(192, 1x1)	Conv(192, 1x1)	Dropout(0.5)
Conv(192, 3x3)	MaxPooling(3x3, 2)	MaxPooling(3x3, 2)	Conv(256, 3x3)
Conv(192, 1x1)	Dropout(0.5)	Dropout(0.5)	Conv(256, 1x1)
Conv(10, 1x1)	Conv(256, 3x3)	Conv(256, 3x3)	Conv(10, 1x1)
GAP	Conv(256, 1x1)	Conv(256, 1x1)	GAP
	Conv(10, 1x1)	Conv(10, 1x1)	
	GAP	GAP	

B NEURAL NETWORK ARCHITECTURES FOR CIFAR-100

We use the following neural network architectures on CIFAR-100 for the main model, the static pre-trained models and the model held out during training.

Table 8: **Neural network architectures used for the CIFAR-100 dataset**. Conv: convolutional layer with Group Normalization and ELU; GAP: global average pooling.

Main model	Pre-trained $model_A$	Pre-trained $model_B$	Holdout model
Conv(96, 3x3)	Conv(96, 5x5)	Conv(192, 5x5)	Conv(96, 3x3)
Conv(96, 3x3)	Conv(96, 1x1)	Conv(96, 1x1)	Dropout(0.2)
Conv(96, 3x3)	MaxPooling(3x3, 2)	MaxPooling(3x3, 2)	Conv(96, 3x3) x 2
Dropout(0.5)	Dropout(0.5)	Dropout(0.5)	Dropout(0.5)
Conv(192, 3x3) x 3	Conv(192, 5x5)	Conv(192, 5x5)	Conv(192, 3x3) x 2
Dropout(0.5)	Conv(192, 1x1)	Conv(192, 1x1)	Dropout(0.5)
Conv(192, 3x3)	MaxPooling(3x3, 2)	MaxPooling(3x3, 2)	Conv(256, 3x3)
Conv(192, 1x1)	Dropout(0.5)	Dropout(0.5)	Conv(256, 1x1)
Conv(100, 1x1)	Conv(256, 3x3)	Conv(256, 3x3)	Conv(100, 1x1)
GAP	Conv(256, 1x1)	Conv(256, 1x1)	GAP
	Conv(100, 1x1)	Conv(100, 1x1)	
	GAP	GAP	