Particle-based Adversarial Local Distribution Regularization

Thanh Nguyen-Duc[†] **Trung Le**[†‡]
He Zhao **Jianfei Cai** **Dinh Phung**
Department of DSAI, Faculty of Information Technology, Monash University
[†]{thanh.nguyen4, trunglm}@monash.edu [‡]corresponding author

Abstract

Adversarial training defense (ATD) and virtual adversarial training (VAT) are the two most effective methods to improve model robustness against attacks and model generalization. While ATD is usually applied in robust machine learning, VAT is used in semi-supervised learning and domain adaption. In this paper, we introduce a novel adversarial local distribution regularization. The adversarial local distribution is defined by a set of all adversarial examples within a ball constraint given a natural input. We illustrate this regularization is a general form of previous methods (e.g., PGD, TRADES, VAT and VADA). We conduct comprehensive experiments on MNIST, SVHN and CIFAR10 to illustrate that our method outperforms well-known methods such as PGD, TRADES and ADT in robust machine learning, VAT in semi-supervised learning and VADA in domain adaption. Our implementation is on Github: `https://github.com/PotatoThanh/ALD-Regularization`.

1 Introduction

Generalization is defined by model's ability to react to unseen input data, which is one of the most challenging problems in machine learning. For examples, the model should be robust to adversarial example inputs from attacks. The model from semi-supervised learning and domain adaptation applications should not be overfitted to finite training data samples in order to generalize well on unseen data. State-of-the-art deep neural networks are reported to be susceptible to attacks [Szegedy et al., 2013, Goodfellow et al., 2014]. These attacks add crafted perturbations to clean inputs to create adversarial examples (e.g., Fast Gradient Sign Method (FGSM) [Goodfellow et al., 2014], Projected Gradient Descent (PGD) [Madry et al., 2018] and Auto-Attack [Croce and Hein, 2020]. The most common way to find the perturbations is using adversarial direction which leverages gradients to maximize the loss of a model on a particular input while keeping the size of the perturbation smaller than a specified amount referred to a radius constraint epsilon. Due to the threats, many methods have been proposed robust regularization using adversarial examples to defense such as Madry et al. [2018], Qin et al. [2019], Xie et al. [2019], Zhang et al. [2019]. In addition, overfitting problem occurs when model performs well on training dataset with low error but the true expected error (test error) is large. Regularization is the most common way to reduce the gap between the training error and the test error in real world applications. In term of using adversarial examples as regularization to improve model generalization, VAT has been introduced by Miyato et al. [2015] to tackle the problem which promotes the smoothness of model output distribution named local distributional smoothness. This regularization shows its effectiveness to reduce overfitting and improve generalization in semi-supervised learning [Miyato et al., 2018]. Then it is adopted to regularize source and target models in domain adaption in order to boost the generalization on the target domain [Shu et al., 2018].

Among defense techniques, adversarial training defense (ATD) is one of the most effective [Athalye et al., 2018, Dong et al., 2020b]. However, ATD heavily relies on attack algorithms to find adversarial samples during the training, which shows poor generalization for other unseen attacks [Song et al., 2018]. Moreover, a single attack algorithm can only create one adversarial sample in a run, which could be insufficient to completely explore the space of possible perturbations. Even PGD attack with random initialization can also lie together and lose diversity [Tashiro et al., 2020].

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151. Copyright 2022 by the author(s).

Recently, Dong et al. [2020a] proposed a technique to form the perturbation distributions named (ADT) to improve model robustness. However, ADT makes a strong assumption that the distribution of perturbations follows certain parameterization forms (e.g., Gaussian distributions).

Training a deep learning model that can generalize well on unseen data is a challenging problem. When neural networks have a lot of parameters to be tuned by finite training samples, overfitting is a common problem. Several regularization techniques have been proposed to overcome the overfitting such as l_2 weight decay and Dropout [Srivastava et al., 2014]. In this paper, we focus on using adversarial examples to reduce overfitting and improve model generalization. Virtual adversarial training (VAT) introduced local distribution smoothness based regularization [Miyato et al., 2018]. With the success of VAT, it has been adopted to domain adaptation to improve generalization on new domains, the corresponding method of which is named VADA [Shu et al., 2018]. However, the drawback of this technique is that VAT cannot explore well the local distribution and generate diverse adversarial examples (see more details about VAT and VADA in following sections).

In this paper, we introduce a novel regularization method using adversarial examples to overcome the drawbacks of previous approaches. Our contributions are summarized as follows: **1)** We propose an adversarial local distribution based regularization to encourage model generalization. The adversarial local distribution is defined by a set of all adversarial examples within a ball constraint that can maximize loss function given a natural input. We also show that this regularization is a general form of well-known previous approaches such PGD [Madry et al., 2018], TRADES [Zhang et al., 2019], VAT [Miyato et al., 2018]) and VADA [Shu et al., 2018]. **2)** We sufficiently approximate the adversarial local distribution without any assumptions using a multiple particle-based search named Stein Variational Gradient Decent (SVGD) [Liu and Wang, 2016]. The SVGD can create more diverse adversarial examples which significantly help to improve model performance. **3)** We show that our method can be adapted well to various applications such as semi-supervised learning, robust machine learning and domain adaptation. We conduct comprehensive experiments on MNIST, SVHN and CIFAR10 datasets to demonstrate that our method outperforms previous well-known approaches in the above applications, such as PGD [Madry et al., 2018], TRADES [Zhang et al., 2019], ADT Dong et al. [2020a] in robust machine learning, VAT [Miyato et al., 2018] in semi-supvervised learning and VADA [Shu et al., 2018] in domain adaptation.

2 Related work

Adversarial training defense (ATD) is one of the most effective techniques to protect deep neural networks from attacks [Athalye et al., 2018, Dong et al., 2020b]. ATD can be formulated as a minmax optimization[Madry et al., 2018]. While the inner maximization of ATD tries to find an adversarial example within a ball constraint that maximizes the classification loss given a natural input, the outer minimization aims to train a robust classifier using the generated adversarial examples. In order to solve the inner maximization problem of ATD, previous works usually used a specific attack algorithm to find adversarial examples such as FGSM [Goodfellow et al., 2014], PGD [Madry et al., 2018] and TRADES [Zhang et al., 2019]. The quality of ATD significantly depends on the strength of injected perturbations of adversarial examples. For example, ATD uses non-iterative method FGSM [Goodfellow et al., 2014], which cannot robust to iterative PGD [Madry et al., 2018] attack. Previous work proposed by Athalye et al. [2018] suggests that the adversarial training defense with PGD can perform well against attacks. Therefore, many works attempt to improve ATD with PGD such as Kannan et al. [2018], Hoang et al. [2020], Bui et al. [2020, 2022], Le et al. [2022]. Recently, contrastive learning [Bui et al., 2021a] and ensemble method [Bui et al., 2021b] have been used to archive state-of-the-art performance. However, these methods only generate only one adversarial example, which could be insufficient to explore entire space of possible perturbations. Moreover, the work proposed by Tashiro et al. [2020] shows even the attacks with random initialization can also lie together and lose diversity that reduce the quality of ATD. Recently, Dong et al. [2020a] proposed a technique to form the perturbation distribution named adversarial distributional training (ADT), where the inner maximization aims to find adversarial distribution for each natural input. However, ADT makes a strong assumption that the perturbation distribution follows Gaussian distribution. This assumption could be insufficient in practice. Therefore, our method addresses a strong diversity of adversarial examples and sufficiently forms the adversarial distribution without any assumption.

Virtual adversarial training (VAT) propopsed by Miyato et al. [2015] is a well-known regularization for semi-supervised learning [Miyato et al., 2018] and domain adaptation [Shu et al., 2018] which can be defined by a minmax optimization problem similar to the adversarial training defense. The inner maximization of VAT aims to find an adversarial example that maximizes

KL divergence loss between model outputs of a natural input and the adversarial example input. The outer minimization aims to smooth the local distribution output of a model given a natural input to reduce overfitting and improve generalization. This technique is called local distribution smoothness based regularization. Similar to ATD, VAT cannot sufficiently explore the local distribution. It is worth to note that there is a strong connection between ATD and VAT (e.g., solving minmax optimization problem and KL divergence loss). For example, TRADES [Zhang et al., 2019] used in ATD solves the mimax problem and leverages KL divergence loss to solve the inner maximization. In this paper, we form a generalization regularization for both ATD and VAT.

Semi-supervised learning is a method to machine learning that combines a small amount labeled data with a large unlabeled data during training. There are several approaches proposed to solve this problem such as entropy minimization [Grandvalet et al., 2005], pseudo-labeling [Lee et al., 2013], MixMatch [Berthelot et al., 2019] and VAT [Miyato et al., 2018]. In this paper, we focus on approaches using adversarial examples to improve performance by smoothing the model output distribution such as VAT [Miyato et al., 2015, 2018].

Domain adaptation is a subcategory of transfer learning, which is addressed as the problem of leveraging labeled data in a source domain to learn an accurate model in an unlabeled dataset of target domain. Many approaches have been proposed to solve the domain adaptation problem such as PixelDA [Bousmalis et al., 2017], ATT [Saito et al., 2017], kNN-Ad [Sener et al., 2016], TIDOT [Nguyen et al., 2021a], MOST [Nguyen et al., 2021b], LAMDA [Le et al., 2021], and STEM [Nguyen et al., 2021c]. The work from Ganin and Lempitsky [2015] attempts to use adversarial training, where it induces a feature extractor to match the source and target features. Recently, Shu et al. [2018] introduced VADA, which adopted the virtual adversarial training (VAT) [Miyato et al., 2018] to boost performance by regularizing both the source and target domains. However, VADA essentially has drawbacks from VAT (e.g., lack of ability to explore the local distribution).

3 Proposed Framework

In this section, we first recall the minmax optimization problem of adversarial training defense (ATD) [Madry et al., 2018] and virtual adversarial training (VAT) [Miyato et al., 2018]. We then formulate a novel adversarial local distribution which is a general distribution for ATD and VAT. The adversarial local distribution (ALD) is efficiently approximated without any comsumption by using multiple particle-based Stein Variational Gradient Descent (SVGD) [Liu and Wang, 2016]. We also show that our method can be adapted in defending against adversarial attacks, semi-supervised learning and domain adaption.

3.1 Minmax optimization of ATD and VAT

ATD and VAT have a common minmax optimization problem but aim to achieve different goals. For example, ATD is used to improve the adversarial robustness of models, while VAT is applied to improve the performance of semi-supervised learning and domain adaptation. Let $\boldsymbol{x} \in \mathbb{R}^d$ be our d-dimensional natural input data in a space $\boldsymbol{X}$. Given an input $(\boldsymbol{x}, y) \sim P_{\mathbb{D}}$ (i.e., the data-label distribution), we denote $B_\epsilon(\boldsymbol{x}) = \{\boldsymbol{x}' \in \boldsymbol{X} : ||\boldsymbol{x}' - \boldsymbol{x}||_p \leq \epsilon\}$ is the ball constraint around the natural sample $\boldsymbol{x}$ with a radius ϵ with respect to a norm $||\cdot||_p$. Given a classifier f_θ parameterized by θ, we define the minmax optimization problem [Madry et al., 2018] as

$$\min_\theta \mathbb{E}_{(\boldsymbol{x},y)\sim P_{\mathbb{D}}} \Big[\max_{\boldsymbol{x}' \in B_\epsilon(\boldsymbol{x})} \ell(\boldsymbol{x}', \boldsymbol{x}, y; \theta) \Big], \tag{1}$$

where $\ell(\boldsymbol{x}', \boldsymbol{x}, y; \theta)$ depends on a particular method. For example, FGSM [Goodfellow et al., 2014], PGD [Madry et al., 2018] use the cross-entropy loss (CE)

$$\ell(\boldsymbol{x}', \boldsymbol{x}, y; \theta) = CE(f_\theta(\boldsymbol{x}'), y), \tag{2}$$

where y is the one-hot ground-truth label of x and $f_\theta(\boldsymbol{x}')$ is the prediction probabilities. Another example is TRADES [Zhang et al., 2019] and VAT [Miyato et al., 2018], which use the Kullback-Leibler divergence loss (D_{KL}) in Eq.(3)

$$\ell(\boldsymbol{x}', \boldsymbol{x}, y; \theta) = D_{\text{KL}}(f_\theta(\boldsymbol{x}'), f_\theta(\boldsymbol{x})). \tag{3}$$

3.2 Adversarial local distribution regularization

Recall that the maximization problem in Eq. (1) is usually solved by the relevant methods such as FGSM, PGD, TRADES, and VAT. However, these methods only find one adversarial example $\boldsymbol{x}'$ given a natural input $\boldsymbol{x}$. In this section, we introduce our proposed adversarial local distribution (ALD) regularization. ALD regularization considers an adversarial local distribution $P_\theta(\boldsymbol{x}'|\boldsymbol{x}, y)$ within a ball constraint B_ϵ which is relevant the the loss function $\ell(\boldsymbol{x}', \boldsymbol{x}, y; \theta)$ as shown in (4).

$$P_\theta(\boldsymbol{x}'|\boldsymbol{x}, y) := \frac{e^{\ell(\boldsymbol{x}', \boldsymbol{x}, y; \theta)}}{\int_{B_\epsilon(\boldsymbol{x})} e^{\ell(\boldsymbol{x}'', \boldsymbol{x}, y; \theta)} d\boldsymbol{x}''} = \frac{e^{\ell(\boldsymbol{x}', \boldsymbol{x}, y; \theta)}}{Z(\boldsymbol{x}, y; \theta)}, \tag{4}$$

where $P_\theta(\cdot|\boldsymbol{x}, y)$ is the conditional local distribution over $B_\epsilon(\boldsymbol{x})$ and $Z(\boldsymbol{x}, y; \theta)$ is a normalization function. Instead of solving directly the inner maximization as in the aforementioned approaches, we sample a set of adversarial examples or particles from this local distribution with the aim to reach its modes and avoid the particle collapse to increase the particle diversity. We note that depending on the loss function ℓ, y could be the one-hot ground-truth label of x or the prediction probabilities $f_\theta(x)$.

Given Eq. (4), we propose the adversarial local distributional regularization term at the position x

$$R(\theta, \boldsymbol{x}, y) := \mathbb{E}_{\boldsymbol{x}' \sim P_\theta(\cdot|\boldsymbol{x}, y)}[\log P_\theta(\boldsymbol{x}'|\boldsymbol{x}, y)] = -H(P_\theta(\cdot|\boldsymbol{x}, y)), \tag{5}$$

where H indicates the entropy of a given distribution.

For $\boldsymbol{x}$ and y, when minimizing $R(\theta, \boldsymbol{x}, y)$ or equivalently $-H(P_\theta(\cdot|\boldsymbol{x}, y))$ w.r.t. θ, we pointwisely maximize $H(P_\theta(\cdot|\boldsymbol{x}, y))$, which is equivalent to encourage $P_\theta(\cdot|\boldsymbol{x}, y)$ to be more uniform distribution. This further enforces $\ell(\boldsymbol{x}', \boldsymbol{x}, y; \theta) = \ell(\boldsymbol{x}'', \boldsymbol{x}, y; \theta) = c(x, y; \theta)$, where $\boldsymbol{x}', \boldsymbol{x}'' \sim P_\theta(\cdot|\boldsymbol{x}, y)$. In other words, it implies that $\ell(\boldsymbol{x}', \boldsymbol{x}, y; \theta)$ is close to a constant $c(x, y; \theta)$ and smooth over $\boldsymbol{x}' \in B_\epsilon(\boldsymbol{x})$. Therefore, minimizing the adversarial local distribution regularization loss leads to an enhancement in the model output smoothness (i.e., the classifier does not change outputs with any input $\boldsymbol{x}' \in B_\epsilon(\boldsymbol{x})$) to encourage model robustness. As demonstrated later, not only strengthening the model robustness in adversarial defense, this also encourages the model generalization in semi-supervised learning and domain adaptation settings.

At this outset, it is worth noting that when sampling only one adversarial example from $P_\theta(\cdot|\boldsymbol{x}, y)$, the Eq. (5) reduces to FGSM, PGD, TRADES, and VAT respectively (see our asymptotic analysis when we assume using the RBF kernel and consider the behaviors when letting the kernel width $\sigma \to 0$ or ∞ below).

3.3 Multiple particle-based search to approximate the adversarial local distribution

In Eq. (4), $Z(\boldsymbol{x}, y; \theta)$ is intractable to find, we thus use a particle-based method to sample $\boldsymbol{x}'_1, \boldsymbol{x}'_2, \ldots, \boldsymbol{x}'_n \sim P_\theta(\cdot|\boldsymbol{x}, y))$, where n is the number of samples (or *adversarial particles*) to solve the optimization problem of finding $P_\theta(\cdot|\boldsymbol{x}, y)$. Here we show that our method can sufficiently explore the adversarial local distribution more efficiently compared to previous methods (e.g., FGSM, PGD, TRADES, ADT, and VAT).

Stein Variational Gradient Decent (SVGD) [Liu and Wang, 2016] is a particle-based inference method using a functional gradient decent to approximate a ground-truth distribution without explicit parametric assumptions. To this end, SVGD is leveraged to be our solver to approximate the adversarial local distribution $P_\theta(\cdot|\boldsymbol{x}, y)$. The core idea is to find a set of adversarial particles to approximate the local distribution using Alg. 1. More specifically, a set of adversarial particles $\{\boldsymbol{x}'_1, \boldsymbol{x}'_2, \ldots, \boldsymbol{x}'_n\}$ is initialized by adding uniform noises, then projected onto the ball B_ϵ. Furthermore, these adversarial particles are then iteratively updated as well as projecting onto the ball B_ϵ (line 4 in Alg. 1) until reaching termination condition. Note that k is a positive definite kernel for which in our experiments, we use radial basic function (RBF) kernel defined in Eq. (6), where the kernel width σ is empirically set by proportional to the number of particles n (i.e., $\sigma = 10^{1-n}$). Additionally, two terms of ϕ (line 5 in Alg. 1) have different roles: (i) the first one enforces the particles move towards to the high density areas of $P_\theta(\cdot|\boldsymbol{x}, y)$ and (ii) the second one prevents all the particles to collapse into local modes of $P_\theta(\cdot|\boldsymbol{x}, y)$.

```
Input: A natural sample $(\boldsymbol{x}, y) \sim P_{\mathbb{D}}$; $n$ number of adversarial particles; $\epsilon$ for the constraint $B_\epsilon$; $r$ normalization function; $\eta$ initial noise factor; $\tau$ step size updating; $N$ number of iterations; $k$ kernel function
Output: Set of adversarial particles $\{\boldsymbol{x}'_1, \boldsymbol{x}'_2, \ldots, \boldsymbol{x}'_n\} \sim P_\theta(\cdot|\boldsymbol{x}, y)$
1 Initialise a set of $n$ particles and project to the $B_\epsilon$ constraint $\{\boldsymbol{x}'_i \in \mathbb{R}^d, i \in \{1, 2, \ldots, n\} | \boldsymbol{x}'_i = \prod_{B_\epsilon}(\boldsymbol{x} + \eta * Uniform\_noise)\}$;
2 for $l = 1$ to $N$ do
3     for each particle $\boldsymbol{x}_i'^{(l)}$ do
4         $\boldsymbol{x}_i'^{(l+1)} = \prod_{B_\epsilon}\left(\boldsymbol{x}_i'^{(l)} + \tau * r(\phi(\boldsymbol{x}_i'^{(l)}))\right)$ ;
5         where $\phi(\boldsymbol{x}') = \frac{1}{n}\sum_{j=1}^{n}[k(\boldsymbol{x}_j'^{(l)}, \boldsymbol{x}')\nabla_{\boldsymbol{x}_j'^{(l)}} \log P(\boldsymbol{x}_j'^{(l)}|\boldsymbol{x}, y) + \nabla_{\boldsymbol{x}_j^{(l)}} k(\boldsymbol{x}_j'^{(l)}, \boldsymbol{x}')]$ ;
6     end
7 end
8 return $\{\boldsymbol{x}_1'^N, \boldsymbol{x}_2'^N, \ldots, \boldsymbol{x}_n'^N\}$ ;
```

Algorithm 1: Approximating the conditional adversarial local distribution given $\boldsymbol{x}$ by using Stein Variational Gradient Decent

$$k(\boldsymbol{x}', \boldsymbol{x}) = \exp\left\{\frac{-||\boldsymbol{x}' - \boldsymbol{x}||^2}{2\sigma^2}\right\}. \tag{6}$$

Asymptotic analysis of adversarial local distribution approximation. Considering the RBF kernel, the update function ϕ can be rewritten as

$$\phi(\boldsymbol{x}') = \frac{1}{n}\sum_{j=1}^{n}\Big[k(\boldsymbol{x}_j'^{(l)}, \boldsymbol{x}')\nabla_{\boldsymbol{x}_j'^{(l)}}\ell(\boldsymbol{x}_j'^{(l)}, \boldsymbol{x}, y; \theta) - k(\boldsymbol{x}_j'^{(l)}, \boldsymbol{x}')\frac{(\boldsymbol{x}_j'^{(l)} - \boldsymbol{x}')}{\sigma^2}\Big]. \tag{7}$$

When $\sigma \to \infty$, it is obvious that

$$\phi(\boldsymbol{x}') \to \frac{1}{n}\sum_{j=1}^{n}\nabla_{\boldsymbol{x}_j'^{(l)}}\ell(\boldsymbol{x}_j'^{(l)}, \boldsymbol{x}, y; \theta). \tag{8}$$

Therefore, our approach reduces exactly to FGSM, PGD, TRADES, and VAT with n independent particles, where in the update quantity is the average of the gradients at each particle as shown in Eq. (10). Evidently, in the update rule in Eq. (10), there does not exist any term that promotes the particle diversity. In addition, when using a single particle (i.e., $n = 1$), our approach under its asymptotic case reduces exactly to the aforementioned approaches.

Particularly, in our update formula in Eq. (13), the first term encourages the particles to seek the optimal values of the loss surface as in FGSM, PGD, TRADES, and VAT, while the second term plays a role of a repulsive term to push the particles away for enhancing the particle diversity. The reason is that when $\boldsymbol{x}_j'^{(l)}$ moves closer to $\boldsymbol{x}'$, the weight $k(\boldsymbol{x}_j'^{(l)}, \boldsymbol{x}')$ becomes larger to push them further away from each other.

We present the asymptotic analysis when $\sigma \to 0$. Considering the RBF kernel, the update function ϕ can be rewritten as

$$\phi(\boldsymbol{x}') = \frac{1}{n}\sum_{j=1}^{n}\Big[k(\boldsymbol{x}_j'^{(l)}, \boldsymbol{x}')\nabla_{\boldsymbol{x}_j'^{(l)}}\ell(\boldsymbol{x}_j'^{(l)}, \boldsymbol{x}, y; \theta) - k(\boldsymbol{x}_j'^{(l)}, \boldsymbol{x}')\frac{(\boldsymbol{x}_j'^{(l)} - \boldsymbol{x}')}{\sigma^2}\Big]. \tag{9}$$

When $\sigma \to 0$, it is obvious that

$$\phi(\boldsymbol{x}') \to \frac{1}{n}\sum_{j=1}^{n}1_{\boldsymbol{x}'=\boldsymbol{x}_j'^{(l)}}\nabla_{\boldsymbol{x}_j'^{(l)}}\ell(\boldsymbol{x}_j'^{(l)}, \boldsymbol{x}, y; \theta), \tag{10}$$

where 1_A is the indicator function which returns 1 if A is true and 0 if otherwise. Here we note that we have used the following equations in the above derivation.

$$\lim_{\sigma\to 0} k(\boldsymbol{x}_j'^{(l)}, \boldsymbol{x}')\frac{(\boldsymbol{x}_j'^{(l)} - \boldsymbol{x}')}{\sigma^2} = 0. \tag{11}$$

$$\lim_{\sigma\to 0} k(\boldsymbol{x}_j'^{(l)}, \boldsymbol{x}') = 0 \tag{12}$$

if $\boldsymbol{x}' \neq \boldsymbol{x}_j'^{(l)}$.

$$\lim_{\sigma\to 0} k(\boldsymbol{x}_j'^{(l)}, \boldsymbol{x}') = 1 \tag{13}$$

if $\boldsymbol{x}' = \boldsymbol{x}_j'^{(l)}$.

Therefore, the update amount $\phi(\boldsymbol{x}')$ in Eq. (10) reduces to only one gradient. It is evident that when $n = 1$, our approach reduces exactly to PGD, TRADES, or VAT.

4 Robust, semi-supervised learning and domain adaptation

In this section, we adapt our adversarial local distribution regularization Eq. (5) to various applications such as robust machine learning, semi-supervised learning and domain adaptation. The previous methods (e.g., PGD, TRADES or VAT) can be addressed as sampling only one adversarial particle in $P_\theta(\cdot|\boldsymbol{x}, y)$. More specifically, our approach with a single particle under its asymptotic setting can reduce to these methods as analysed in the previous section.

We now illustrate how to apply our adversarial local distribution regularization to specific problems such as semi-supervised learning, robust machine learning, and domain adaptation. Generally, our adversarial local distribution (ADL) regularization can be applied to a data example x with or without label to make its local vicinity more smoothly. More specifically, if $\boldsymbol{x}$ has a label y, we can flexibly apply $R(\theta, \boldsymbol{x}, y)$ as in Eq. (2) or $R(\theta, \boldsymbol{x}, f_\theta(\boldsymbol{x}))$ as in Eq. (3). In contrast, if $\boldsymbol{x}$ is an unlabeled data example, we can use $R(\theta, \boldsymbol{x}, f_\theta(\boldsymbol{x}))$ as in Eq. (3). We hence can apply our ALD regularization to both labeled and unlabeled portions in the semi-supervised setting, both source and target datasets in domain adaptation, and labeled dataset in robust machine learning. Moreover, our ADL regularization for each data example x can be estimated conveniently by sampling a set of adversarial particles $\{\boldsymbol{x}_1', \boldsymbol{x}_2', \ldots, \boldsymbol{x}_n'\} \sim P_\theta(\cdot|\boldsymbol{x}, y)$ using Alg. 1.

For semi-supervised learning, $(\boldsymbol{x}_l, y) \sim P_{\mathbb{D}_l}$ and $\boldsymbol{x}_{ul} \sim P_{\mathbb{D}_{ul}}$, where $\mathbb{D}_l$ and $\mathbb{D}_{ul}$ is the labeled and unlabeled dataset respectively. Based on the VAT loss, the loss function is adapted to semi-supervised learning using cross-entropy loss (CE) and adversarial local distribution regularizations weighted by λ_1 and λ_2, as shown in (14).

$$\min_\theta \Big\{\mathbb{E}_{(\boldsymbol{x}_l, y)\sim P_{\mathbb{D}_l}}\Big[CE(f_\theta(\boldsymbol{x}_l), y) + \lambda_1 R(\theta, \boldsymbol{x}_l, f_\theta(\boldsymbol{x}_l))\Big] + \lambda_2 \mathbb{E}_{\boldsymbol{x}_{ul}\sim P_{\mathbb{D}_{ul}}}\Big[R(\theta, \boldsymbol{x}_{ul}, f_\theta(\boldsymbol{x}_{ul}))\Big]\Big\}, \tag{14}$$

where $R(\theta, \boldsymbol{x}, f_\theta(\boldsymbol{x}))$ is relevant to the loss in Eq. (3).

For robust machine learning, $(\boldsymbol{x}, y) \sim P_{\mathbb{D}}$, where $\mathbb{D}$ is the dataset. The loss function is adapted to this problem using cross-entropy loss (CE) and adversarial local distribution regularization weighted by λ. Based on PGD or TRADES, we can adapt the loss function

$$\min_{\theta}\left\{\mathbb{E}_{(\boldsymbol{x},y)\sim P_{\mathbb{D}}}\left[CE(f_\theta(\boldsymbol{x}),y)\right]+\lambda R(\theta,\boldsymbol{x},\tilde{y})\right]\right\}, \tag{15}$$

where in $R(\theta,\boldsymbol{x},\tilde{y})$, we set $\tilde{y}=y$ (cf. Eq. (2)) for our PGD version and $\tilde{y}=f_\theta(\boldsymbol{x})$ (cf. Eq. (3)) for our TRADES version.

For domain adaptation, $(\boldsymbol{x}_s,y)\sim P_{\mathbb{D}_s}$ and $\boldsymbol{x}_t\sim P_{\mathbb{D}_t}$, where $\mathbb{D}_s$ and $\mathbb{D}_t$ is the source and target dataset respectively. Based on VADA loss[1], our regularization terms for both source classifier parameterized by θ_s and target classifier parameterized by θ_t using adversarial local distribution regularization. The λ_1 and λ_2 are the trade-off parameters for source and target regularization respectively, as shown in (16).

$$\begin{aligned}\min_{\theta}\Big\{&\lambda_1\mathbb{E}_{(\boldsymbol{x}_s,y)\sim P_{\mathbb{D}_s}}\left[R(\theta_s,\boldsymbol{x}_s,f_\theta(\boldsymbol{x}_s))\right]\\&+\lambda_2\mathbb{E}_{\boldsymbol{x}_t\sim P_{\mathbb{D}_t}}\left[R(\theta_t,\boldsymbol{x}_t,f_\theta(\boldsymbol{x}_t))\right]\Big\},\end{aligned} \tag{16}$$

where $R(\theta,\boldsymbol{x},f_\theta(\boldsymbol{x}))$ is relevant to the loss in Eq. (3).

5 Experiments

In this section, we conducted several comprehensive experiments using MNIST [LeCun et al., 1998], SVHN [Netzer et al., 2011] and CIFAR10 [Krizhevsky et al., 2009] datasets. We first analyze the adversarial particles generated by SVGD compared to adversarial examples from PGD with random initializations. We compare performance of our method to several well-known approaches such as PGD [Madry et al., 2018], TRADES [Zhang et al., 2019], ADT [Dong et al., 2020a] in robust machine learning, VAT [Miyato et al., 2018] in semi-supvervised learning and VADA [Shu et al., 2018] in domain adaptation. Please refer all experimental setup details given in the supplementary material.

5.1 Diversity of adversarial particles vs. random initialization

General setup. The pretrained model of MNIST and CIFAR10 used in this experiment is LeNet [LeCun et al., 1998] and ResNet18 [He et al., 2016] respectively. The LeNet achieves 0.99 accuracy on MNIST, while ResNet18 achieves 0.93 accuracy on CIFAR10. We fix all pretrained models in order to generate adversarial examples using PGD with random initialization and adversarial particles using our method. We set the same ϵ (e.g., 0.1 for MNIST, 8/255 for CIFAR10) and number of iterations $N=200$. Note that all adversarial examples and particles fool the classifiers with 1.0 confident.

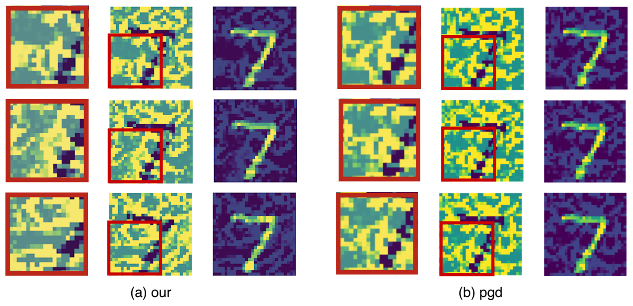
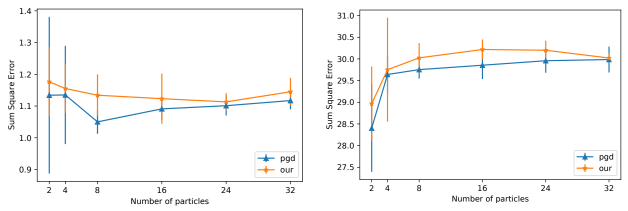


Figure 1: Comparison of three adversarial examples generated by (a) our method with SVGD and (b) PGD with random initialization. The first, second and third column of each sub-figure is region of interest of adversarial perturbations (ROIs), adversarial perturbations and adversarial particles respectively.



(a) MNIST (b) CIFAR10

Figure 2: Diversity comparison of our method and PGD with random initialization using sum of square error (SSE). The figure illustrates the average of mean (point) and standard deviation (bar) of the three different runs.

Experimental setup. In Fig. 1, we generate 3 adversarial examples with random initializations using PGD for an MNIST image (e.g., digit 7). Given the same image, we also sample 3 adversarial particles in the adversarial local distribution $P_\theta(\cdot|\boldsymbol{x})$ using SVGD. In Fig. 2, we generate adversarial examples using PGD with random initializations and our SVGD method with different numbers of particles for the MNIST and CIFAR10 datasets. We then calculate sum squared error (SSE) between these particles to evaluate their diversity. At each setting of the number of particles, we run 3 times to calculate the average of the means and standard deviations of SSE.

Results. Recall that in Alg. 1, SVGD is designed to

[1]There are several losses in VADA; thus, we only show our regularization for source and target domain.

Table 1: Performance comparison between our method and VAT using mixup technique for all adversarial particles in mini-batch on Conv-Large architecture.

n particle(s)	1	2	4	8
VAT	0.8601	0.8611	0.858	0.856
Our	0.867	0.876	0.883	0.872
VAT + Mixup	0.870	0.887	0.9013	0.893
Our + Mixup	**0.913**	**0.925**	**0.930**	**0.927**

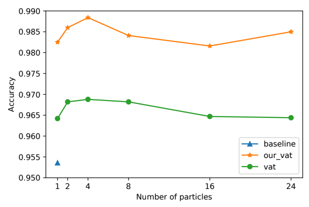
generate diverse adversarial particles from $P_\theta(\cdot|\boldsymbol{x})$ because the first and second terms in ϕ enforce the particles to stay in the high density areas and avoid collapsing into local modes, respectively. Previous methods such as PGD, TRADES and VAT reply on random initialization to generate different adversarial examples in $P_\theta(\cdot|\boldsymbol{x}, y)$. Therefore, these previous methods can lie together and lose diversity [Tashiro et al., 2020]. Thus, adversarial examples generated from the previous methods are not diverse enough to improve the model performance compared to our method. As seen in Fig. 1, our method can have significantly different noise patterns (the middle column) compared to PGD especially in the regions of interests in the first column. In Fig. 2, the figure further shows that our method can generate more diverse samples with bigger SSE compared to PGD with random initializations in both MNIST and CIFAR10 datasets.

5.2 Semi-supervised learning

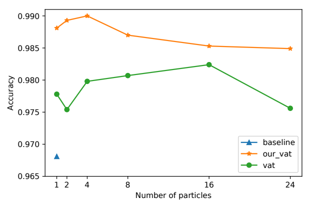
Datasets. In order to conduct the experiment with different numbers of labeled samples, we select 300 and 500 labeled samples from the MNIST training dataset and the rest of training samples are unlabeled samples, denoted by MNIST-300 and MNIST-500 respectively. We also select 1000 and 4000 labeled samples from the CIFAR10 training dataset and the rest of training samples are unlabeled samples, denoted by CIFAR10-1000 and CIFAR10-4000 respectively.

General setup. We set the radius constraint $\epsilon = 0.01$ for MNIST, $\epsilon =$ 5e-4 for CIFAR10 and the number of iterations $N = 1$ for both. We use the LeNet [LeCun et al., 1998] architecture for MNIST and Conv-Large architecture following VAT [Miyato et al., 2018] for CIFAR10. We train both of models 500 epochs using the SGD optimizer[2].

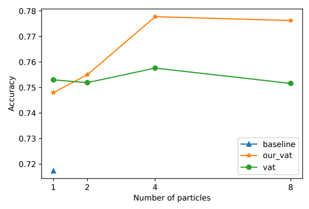
Experimental setup. We setup two experiments for semi-supervised learning using the MNIST and CIFAR10 datasets. The first experiment is the performance comparison between our method and VAT. VAT can generate different adversarial examples using



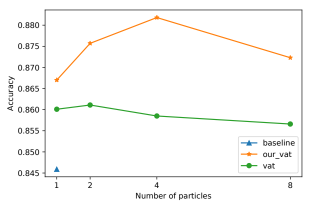
(a) MNIST-300



(b) MNIST-500



(c) CIFAR10-1000



(d) CIFAR10-4000

Figure 3: Performance comparison of semi-supervised learning using MNIST with LeNet (first row) and CIFAR10 with Conv-Large (second row). (a) MNIST-300 and (b) MNIST-500 use 300 and 500 labeled data of MNIST training set respectively and the rest of training set as unlabeled data . (c) CIFAR10-1000 and (b) CIFAR10-4000 use 1000 and 4000 labeled data of CIFAR10 training data respectively and the rest of training set as unlabeled data. Baseline model is trained by using only labeled data.

random initialization; therefore, we compare between VAT and our method at different number of adversarial particles n. Note that we can encourage the regularization strength of both VAT and our method by increasing number of adversarial particles n. The second experiment is to leverage the Mixup [Zhang et al., 2017] technique to encourage global smoothness in case of CIFAR10-4000 with Conv-Large architecture in the work [Miyato et al., 2018]. Mixup [Zhang et al., 2017] is a data augmentation technique which generates new training samples by weighted combinations of random image pairs from the training data. We apply Mixup to all adversarial particles within mini-batch to encourage global smoothness, while the local smoothness is enforced by the adversarial local distribution regularization.

Results. Our method can significantly outperform VAT on both MNIST and CIFAR10, as shown in Fig. 3. When the number of adversarial particles n=4, our method reaches 0.779 and 0.883 accuracy, while VAT with random initializations achieves only 0.757 and 0.858 accuracy in case of CIFAR10-1000 and CIFAR10-4000 respectively. Our method increases 6% and 3.6% accuracy compared to the baseline models as shown in Fig. 3c and 3d. By increasing the number of particles, we accordingly increase the regularization strength of our model. It is as expected that over regularization may hurt the performance. Therefore, we

[2]Based on https://github.com/iBelieveCJM/Tricks-of-Semi-supervisedDeepLeanring-Pytorch

observe the dropping accuracy at n=24 for MNIST and n=8 for CIFAR10. However, in these cases, our method can still outperform VAT.

In Table 1, Mixup with mini-batch to encourage global smoothness can significantly improve accuracy of both VAT and our method. Our method achieves 0.93 accuracy which outperforms VAT because our method can generate more diverse adversarial particles.

5.3 Robust machine learning

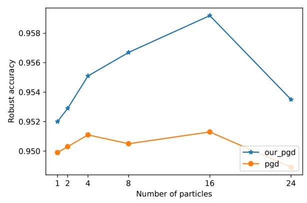
Datasets. The MNIST and CIFAR10 datasets are used in this experiment. For each dataset, all images are scaled from 0 to 1 and we split 1000 samples from the training set as the validation set.

General setup. We set the radius constraint $\epsilon = 0.3$, the number of iterations N=40 for MNIST and $\epsilon = 8/255$, N=10 for CIFAR10. We also use LeNet [LeCun et al., 1998] architecture for MNIST and ResNet18 [He et al., 2016] architecture for CIFAR10[3].

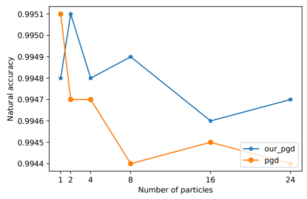
Experimental setup. PGD [Madry et al., 2018] and TRADES [Zhang et al., 2019] are two well-known defense techniques in adversarial training defense. While PGD uses the cross-entropy loss (Eq. 2), TRADES uses the KL divergence loss (Eq. 3). Recall that our method can be applied with any loss function; therefore, we compare our method with the CE loss (denoted by Our_PGD) vs. PGD and our method with the KL divergence loss (denoted by Our_TRADES) vs. TRADES. We also compare our method with adversarial distributional training [Dong et al., 2020a] (ADT) such as ADT-EXP and ADT-EXPAM, which assume that the adversarial distribution explicitly follows normal distribution.

We evaluate natural accuracy and robust accuracy at different number of adversarial particles in Fig. 4 and 5. Natural accuracy is accuracy of a model with natural inputs, while robust accuracy shows the robustness of a model against adversarial examples generated by attacks. PGD [Madry et al., 2018] is widely used to attack models because of its effectiveness and stability. We use PGD with large number of iterations N=200 iteration steps (PGD-200) as a major metric to draw robust accuracy of Fig. 4 and 5. We also use advanced attacks to evaluate the models to evaluate the model robustness against various attacks such as Auto-Attack [Croce and Hein, 2020] and B&W attack [Brendel et al., 2019].

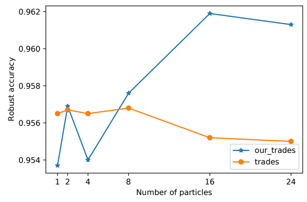
Results. As can be seen in Fig. 4 and 5, our method outperforms PGD and TRADES in term of robust accuracy with the increased number of adversarial particles. All of the four methods, PGD, TRADES,



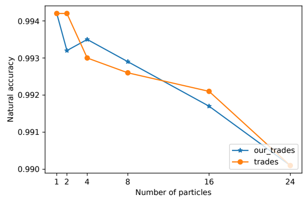
(a) Robust acc - CE loss



(b) Natural acc - CE loss



(c) Robust acc - KL loss



(d) Natural acc - KL loss

Figure 4: Robust accuracy against PGD-200 and natural accuracy comparison using MNIST with LeNet architecture.

Table 2: Robust accuracy comparison using CIFAR10 with ResNet18.

Method	Robust accuracy		
	PGD-200	Auto-Attack	B&B
ADT-EXP	0.458	0.458	0.465
ADT-EXPAM	0.461	0.445	0.458
PGD	0.455	0.419	0.426
Our_PGD	0.471	0.436	0.44
TRADES	0.525	0.483	0.487
Our_TRADES	**0.539**	**0.501**	**0.506**

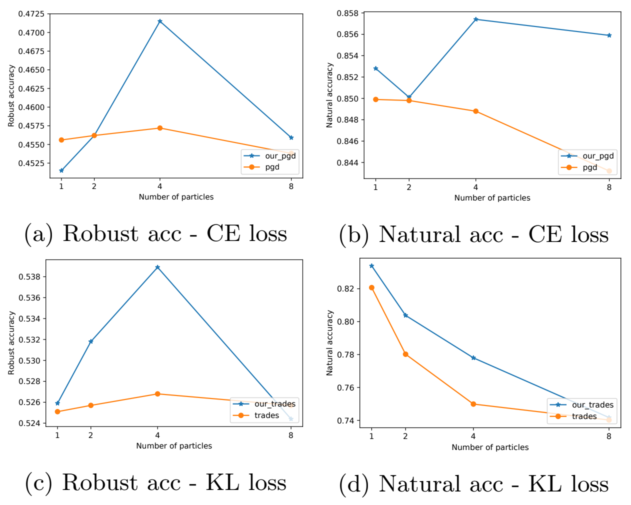
Our_PGD and Our_TRADES decrease natural accuracy with the overly enforced regularization strength when the number of adversarial particles is set to a too large number. This trade-off between natural and robust accuracy is inline with the study in Zhang et al. [2019]. However, our method can achieve higher natural accuracy than others at the same number of particles.

In Table 3, Our_PGD and Our_TRADES can consistently outperform standard PGD and TRADES against various attacks respectively. ADT has better robust accuracy than Our_PGD in Auto-Attack and B&W attack but Our_TRADES achieves the best performance. Here we emphasize that our method does not assume a particular parameterization of the adversarial local distribution, which is more flexible than ADT.

5.4 Domain adaptation

General setup. We use a pair of the SVHN and MNIST datasets to evaluate performance of our

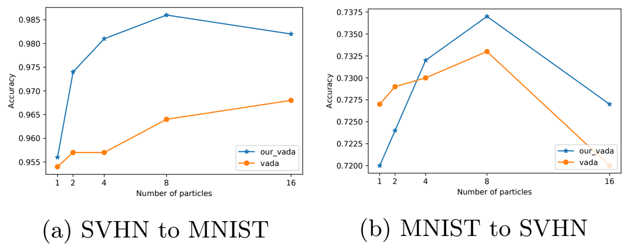
[3]Based on https://github.com/tuananhbui89/Adversarial-Divergence-Reduction



(a) Robust acc - CE loss (b) Natural acc - CE loss

(c) Robust acc - KL loss (d) Natural acc - KL loss

Figure 5: Robust accuracy against PGD-200 and natural accuracy comparison using CIFAR10 with ResNet18 architecture.



(a) SVHN to MNIST (b) MNIST to SVHN

Figure 6: Domain adaptation performance comparison between VADA and our method.

method compared to VADA [Shu et al., 2018] [4]. For transferring from the SVHN source to the MNIST target, we set the radius constraint $\epsilon = 3.5$ and the number of iterations N=1. For transferring from the MNIST source to the SVHN target, we set the $\epsilon = 1.5$ and N=1.

Results. In Fig. 6, we can achieve higher performance when increasing the number of adversarial particles. Moreover, our method can significantly outperform VADA in case of SVHN to MNIST when n=8. When the number of adversarial particles n=14, as discussed in the other tasks, over regularization can hurt the accuracy the target domain. However, our method remains better accuracy than VADA.

5.5 Running time

General setup. PGD, TRADES and VAT have different Pytorch implementations. Therefore, we adapt our method to these individual code base. We observe the running time on our workstation machine with a TITAN V GPU, 16 cores CPU and 64GB of RAM. In addition, PGD, TRADES and VAT implementations

[4]Based on https://github.com/ozanciga/dirt-t

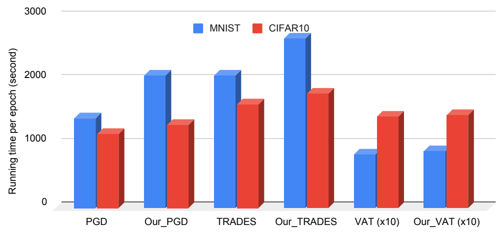
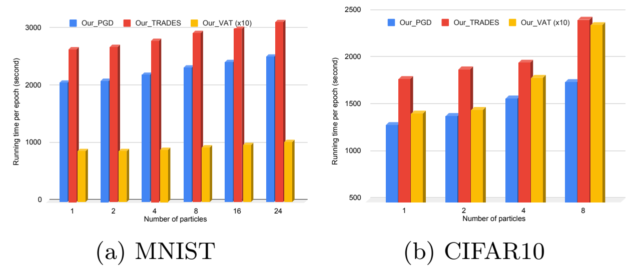


Figure 7: Running time per epoch of compared methods on MNIST and CIFAR10.



(a) MNIST (b) CIFAR10

Figure 8: Running time per epoch of our method at different number of adversarial particles on MNIST and CIFAR10.

do not optimized for multiple adversarial examples. Thus, we only compare running time per epoch when number of particles n=1 in Fig. 7. We also illustrate the running of our method at different number of adversarial particles in Fig. 8.

Results. Due to overhead of kernel computation, the running time of our method is slightly bigger than PGD, TRADES and VAT, as shown in Fig. 7. However, with the efficient implementation on GPUs, the running time per epoch of our method scales linearly with the number of adversarial particles, as shown in Fig. 8.

6 Conclusion

In this paper, we have introduced a novel adversarial local distribution regularization technique that extends and improves on previous methods (e.g., FGSM, PGD, TRADES, VAT and VADA).In our method, SVGD is used to approximate the adversarial local distribution by using more diverse adversarial particles. We adapt our method to a wide range of applications where better generalization is needed, such as semi-supervised learning, robust machine learning and domain adaptation. Comprehensive experiments show that our method can significantly outperform many widely-used regularization approaches used in the above applications, such as PGD, TRADES, ADT in robust machine learning, VAT in semi-supvervised learning and VADA in domain adaptation.

Acknowledgements

This work was partially supported by the Australian Defence Science and Technology (DST) Group under the Next Generation Technology Fund (NGTF) scheme.

References

DIRT-T. `https://github.com/ozanciga/dirt-t`. Accessed: 2021-10-15.

A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of ICML*, pages 274–283. PMLR, 2018.

D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. Raffel. Mixmatch: A holistic approach to semi-supervised learning. *arXiv preprint arXiv:1905.02249*, 2019.

K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of CVPR*, pages 3722–3731, 2017.

W. Brendel, J. Rauber, M. Kümmerer, I. Ustyuzhaninov, and M. Bethge. Accurate, reliable and fast robustness evaluation. *arXiv preprint arXiv:1907.01003*, 2019.

A. Bui, T. Le, H. Zhao, P. Montague, O. deVel, T. Abraham, and D. Phung. Improving adversarial robustness by enforcing local and global compactness. In *Proceedings of ECCV*, pages 209–223. Springer, 2020.

A. Bui, T. Le, H. Zhao, P. Montague, S. Camtepe, and D. Phung. Understanding and achieving efficient robustness with adversarial supervised contrastive learning. *arXiv preprint arXiv:2101.10027*, 2021a.

A. T. Bui, T. Le, H. Zhao, P. Montague, O. deVel, T. Abraham, and D. Phung. Improving ensemble robustness by collaboratively promoting and demoting adversarial robustness. *Proceedings of AAAI*, 35 (8):6831–6839, May 2021b.

A. T. Bui, T. Le, Q. H. Tran, H. Zhao, and D. Phung. A unified wasserstein distributional robustness framework for adversarial training. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=Dzpe9C1mpiv`.

F. Croce and M. Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *Proceedings of ICML*, pages 2206–2216. PMLR, 2020.

Y. Dong, Z. Deng, T. Pang, J. Zhu, and H. Su. Adversarial distributional training for robust deep learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Proceedings of NeurIPS*, volume 33, pages 8270–8283, 2020a.

Y. Dong, Q.-A. Fu, X. Yang, T. Pang, H. Su, Z. Xiao, and J. Zhu. Benchmarking adversarial robustness on image classification. In *Proceedings of CVPR*, pages 321–331, 2020b.

Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proceedings of ICML*, pages 1180–1189. PMLR, 2015.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Y. Grandvalet, Y. Bengio, et al. Semi-supervised learning by entropy minimization. *CAP*, 367:281–296, 2005.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of CVPR*, pages 770–778, 2016.

Q. Hoang, T. Le, and D. Phung. Parameterized rate-distortion stochastic encoder. In H. D. III and A. Singh, editors, *Proceedings of ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 4293–4303. PMLR, 13–18 Jul 2020.

H. Kannan, A. Kurakin, and I. Goodfellow. Adversarial logit pairing. *arXiv preprint arXiv:1803.06373*, 2018.

A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

T. Le, T. Nguyen, N. Ho, H. Bui, and D. Phung. Lamda: Label matching deep domain adaptation. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 6043–6054. PMLR, 18–24 Jul 2021.

T. Le, A. Bui, T. Le, H. Zhao, P. Montague, Q. Tran, and P. Dinh. On global-view based defense via adversarial attack and defense risk guaranteed bounds. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2022.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

D.-H. Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Proceedings of ICML workshop*, volume 3, page 896, 2013.

Q. Liu and D. Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In D. Lee, M. Sugiyama, U. Luxburg,

I. Guyon, and R. Garnett, editors, *Proceedings of NeurIPS*, volume 29, 2016.

A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *Proceedings of ICLR*, 2018.

T. Miyato, S.-i. Maeda, M. Koyama, K. Nakae, and S. Ishii. Distributional smoothing with virtual adversarial training. *arXiv preprint arXiv:1507.00677*, 2015.

T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE TPAMI*, 41(8):1979–1993, 2018.

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

T. Nguyen, T. Le, N. Dam, Q. H. Tran, T. Nguyen, and D. Phung. Tidot: A teacher imitation learning approach for domain adaptation with optimal transport. In Z.-H. Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 2862–2868. International Joint Conferences on Artificial Intelligence Organization, 8 2021a. doi: 10.24963/ijcai.2021/394. URL `https://doi.org/10.24963/ijcai.2021/394`. Main Track.

T. Nguyen, T. Le, H. Zhao, Q. H. Tran, T. Nguyen, and D. Phung. Most: Multi-source domain adaptation via optimal transport for student-teacher learning. In *Uncertainty in Artificial Intelligence*, pages 225–235. PMLR, 2021b.

V.-A. Nguyen, T. Nguyen, T. Le, Q. H. Tran, and D. Phung. Stem: An approach to multi-source domain adaptation with guarantees. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9352–9363, 2021c.

C. Qin, J. Martens, S. Gowal, D. Krishnan, K. Dvijotham, A. Fawzi, S. De, R. Stanforth, and P. Kohli. Adversarial robustness through local linearization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Proceedings of NeurIPS*, volume 32, 2019.

K. Saito, Y. Ushiku, and T. Harada. Asymmetric tri-training for unsupervised domain adaptation. In *Proceedings of ICML*, pages 2988–2997. PMLR, 2017.

O. Sener, H. O. Song, A. Saxena, and S. Savarese. Learning transferrable representations for unsupervised domain adaptation. In *Proceedings of NeurIPS*, pages 2110–2118, 2016.

R. Shu, H. H. Bui, H. Narui, and S. Ermon. A dirt-t approach to unsupervised domain adaptation. *arXiv preprint arXiv:1802.08735*, 2018.

C. Song, K. He, L. Wang, and J. E. Hopcroft. Improving the generalization of adversarial training with domain adaptation. *arXiv preprint arXiv:1810.00740*, 2018.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Machine Learning Research*, 15(1):1929–1958, 2014.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Y. Tashiro, Y. Song, and S. Ermon. Diversity can be transferred: Output diversification for white-and black-box attacks. *Proceedings of NeurIPS*, 33, 2020.

C. Xie, Y. Wu, L. v. d. Maaten, A. L. Yuille, and K. He. Feature denoising for improving adversarial robustness. In *Proceedings of CVPR*, pages 501–509, 2019.

H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan. Theoretically principled trade-off between robustness and accuracy. In *Proceedings of ICML*, pages 7472–7482. PMLR, 2019.

Appendix

Natural accuracy of Table 2

We illustrate additional natural accuracy in the Table 2 of main paper. As can be seen, Our_PGD can achieve the highest natural accuracy. Our_TRADES is more robust against various attacks but we trade off natural accuracy. This trade-off between natural and robust accuracy is inline with the study in [Zhang et al., 2019].

Table 3: Robust and natural accuracy comparison using CIFAR10 with ResNet18.

Method	Natural accuracy
ADT-EXP	0.83
ADT-EXPAM	0.84
PGD	0.852
Our_PGD	**0.857**
TRADES	0.834
Our_TRADES	0.778

Experimental setting details

Semi-supervised learning

For MNIST,

- We select 300 and 500 labeled samples from the MNIST training dataset and the rest of training samples (59700 and 59500) are unlabeled samples , denoted by MNIST-300 and MNIST-500 respectively. Test set consists of 10000 images. All images are scaled from 0 to 1.
- We set ϵ = 0.01, τ=0.01, r = l_2 normalization, η = 10, $\lambda_1 = \lambda_2$=30 and N = 1.
- We use LeNet architecture [LeCun et al., 1998] trained by 400 epochs using SGD optimizer with initial learning rate = 0.1, momentum = 0.9, batch size = 128 and cosine annealing learning rate scheduling between 1e-4 and 0.1.

For CIFAR10,

- We select 1000 and 4000 labeled samples from the CIFAR10 training dataset and the rest of training samples (49000 and 46000) are unlabeled samples , denoted by CIFAR-1000 and CIFAR10-4000 respectively. Test set consists of 10000 images. All images are scaled using mean = [0.4914, 0.4822, 0.4465], std = [0.2023, 0.1994, 0.2010]
- We set ϵ = 5e-4 , τ=0.01, r = l_2 normalization, η = 10, $\lambda_1 = \lambda_2$=30 and N = 1.
- We use Conv-Large architecture [Miyato et al., 2018] trained by 600 epochs using SGD optimizer with initial learning rate = 0.1, momentum = 0.9, batch size = 128 and cosine annealing learning rate scheduling between 1e-4 and 0.1.

Robust machine learning

For MNIST,

- We select 59000 images for training, 1000 images for validation and 10000 images for testing. All images are scaled from 0 to 1.
- We set ϵ = 0.3, τ=0.01, r = l_{inf} normalization, η = 1e-3, λ=1 and N = 40.
- We use LeNet architecture [LeCun et al., 1998] trained by 100 epochs using SGD optimizer with initial learning rate = 0.01, momentum = 0.9, batch size = 100 and learning rate decay (0.1) scheduling at [75, 90] epoch.

For CIFAR10,

- We select 59000 images for training, 1000 images for validation and 10000 images for testing. All images are scaled from 0 to 1.
- We set ϵ = 0.031, τ=0.007, r = l_{inf} normalization, η = 1e-3 and N = 10. PGD and Our_PGD use λ=1, while TRADES and Our_TRADES use λ=6.
- We use ResNet18 architecture [He et al., 2016] trained by 100 epochs using SGD optimizer with learning rate = 0.1, momentum = 0.9, weight decay = 5e-4, batch size = 100 and learning rate decay (0.1) scheduling at [75, 90] epoch.

Domain adaptation

For SVHN source to MNIST target,

- We scale all images in SVHN and MNIST to [0, 1].
- We set ϵ = 0.35, r = l_2 normalization, η = 10, λ_1 = 0.1, λ_2 = 1.0 and N = 1.
- We use Small-CNN-Net architecture in [DIR] trained by 100 epochs using Adam optimizer with learning rate = 2e-3.

For MNIST source to SVHN target,

- We scale all images in MNIST and SVHN using mean = 0.5 and std = 0.5.
- We set $\epsilon = 0.15$, $r = l_2$ normalization, $\eta = 10$, $\lambda_1 = $ 1e-2, $\lambda_2 = $ 1e-2 and $N = 1$.
- We use Large-CNN-Net architecture in [DIR] trained by 100 epochs using Adam optimizer with learning rate = 2e-3 and batch size =32.