TANGENT-NORMAL ADVERSARIAL REGULARIZATION FOR SEMI-SUPERVISED LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

The ever-increasing size of modern datasets combined with the difficulty of obtaining label information has made semi-supervised learning of significant practical importance in modern machine learning applications. In comparison to supervised learning, the key difficulty in semi-supervised learning is how to make full use of the unlabeled data. In order to utilize manifold information provided by unlabeled data, we propose a novel regularization called the tangent-normal adversarial regularization, which is composed by two parts. The two parts complement with each other and jointly enforce the smoothness along two different directions that are crucial for semi-supervised learning. One is applied along the tangent space of the data manifold, aiming to enforce local invariance of the classifier on the manifold, while the other is performed on the normal space orthogonal to the tangent space, intending to impose robustness on the classifier against the noise causing the observed data deviating from the underlying data manifold. Both of the two regularizers are achieved by the strategy of virtual adversarial training. Our method has achieved state-of-the-art performance on semi-supervised learning tasks on both artificial dataset and practical datasets.

1 INTRODUCTION

The recent success of supervised learning (SL) models, like deep convolutional neural networks, highly relies on the huge amount of labeled data. However, though obtaining data itself might be relatively effortless in various circumstances, to acquire the annotated labels is still costly, limiting the further applications of SL methods in practical problems. Semi-supervised learning (SSL) models, which requires only a small part of data to be labeled, does not suffer from such restrictions. The advantage that SSL depends less on well-annotated datasets makes it of crucial practical importance and draws lots of research interests. The common setting in SSL is that we have access to a relatively small amount of labeled data and much larger amount of unlabeled data. And we need to train a classifier utilizing those data. Comparing to SL, the main challenge of SSL is how to make full use of the huge amount of unlabeled data, i.e., how to utilize the marginalized input distribution $p(x)$ to improve the prediction model i.e., the conditional distribution of supervised target $p(y|x)$. To solve this problem, there are mainly three streams of research.

The first approach, based on probabilistic models, recognizes the SSL problem as a specialized missing data imputation task for classification problem. The common scheme of this method is to establish a hidden variable model capturing the relationship between the input and label, and then applies Bayesian inference techniques to optimize the model (Kingma et al., 2014; Zhu et al., 2003; Rasmus et al., 2015). Suffering from the estimation of posterior being either inaccurate or computationally inefficient, this approach performs less well especially in high-dimensional dataset (Kingma et al., 2014).

The second line tries to construct proper regularization using the unlabeled data, to impose the desired smoothness on the classifier. One kind of useful regularization is achieved by adversarial training (Goodfellow et al., 2014b), or virtual adversarial training (VAT) when applied to unlabeled data (Miyato et al., 2016; 2017). Such regularization leads to robustness of classifier to adversarial examples, thus inducing smoothness of classifier in input space where the observed data is presented. The input space being high dimensional, though, the data itself is concentrated on a underlying manifold of much lower dimensionality (Cayton, 2005; Narayanan & Mitter, 2010; Chapelle et al., 2009;

Rifai et al., 2011). Thus directly performing VAT in input space might overly regularize and does potential harm to the classifier. Another kind of regularization called manifold regularization aims to encourage invariance of classifier on manifold (Simard et al., 1998; Belkin et al., 2006; Niyogi, 2013; Kumar et al., 2017; Rifai et al., 2011), rather than in input space as VAT has done. Such manifold regularization is implemented by *tangent propagation* (Simard et al., 1998; Kumar et al., 2017) or *manifold Laplacian norm* (Belkin et al., 2006; Lecouat et al., 2018), requiring evaluating the Jacobian of classifier (with respect to manifold representation of data) and thus being highly computationally inefficient.

The third way is related to generative adversarial network (GAN) (Goodfellow et al., 2014a). Most GAN based approaches modify the discriminator to include a classifier, by splitting the real class of original discriminator into K subclasses, where K denotes the number of classes of labeled data (Salimans et al., 2016; Odena, 2016; Dai et al., 2017; Qi et al., 2018). The features extracted for distinguishing the example being real or fake, which can be viewed as a kind of coarse label, have implicit benefits for supervised classification task. Besides that, there are also works jointly training a classifier, a discriminator and a generator (Li et al., 2017).

Our work mainly follows the second line. We firstly sort out three important assumptions that motivate our idea:

The manifold assumption The observed data presented in high dimensional space is with high probability concentrated in the vicinity of some underlying manifold of much lower dimensionality (Cayton, 2005; Narayanan & Mitter, 2010; Chapelle et al., 2009; Rifai et al., 2011). We denote the underlying manifold as $\mathcal{M}$. We further assume that the classification task concerned relies and only relies on $\mathcal{M}$ (Rifai et al., 2011).

The noisy observation assumption The observed data x can be decomposed into two parts as $x = x_0 + n$, where x_0 is exactly supported on the underlying manifold $\mathcal{M}$ and n is some noise independent of x_0 (Bengio et al., 2013; Rasmus et al., 2015). With the assumption that the classifier only depends on the underlying manifold $\mathcal{M}$, the noise part might have undesired influences on the learning of the classifier.

The semi-supervised learning assumption If two points $x_1, x_2 \in \mathcal{M}$ are close in manifold distance, then the conditional probability $p(y|x_1)$ and $p(y|x_2)$ are similar (Belkin et al., 2006; Rifai et al., 2011; Niyogi, 2013). In other words, the true classifier, or the true condition distribution $p(y|X)$ varies smoothly along the underlying manifold $\mathcal{M}$.

Inspired by the three assumptions, we introduce a novel regularization called the *tangent-normal adversarial regularization* (TNAR), which is composed by two parts. The *tangent adversarial regularization* (TAR) induces the smoothness of the classifier along the tangent space of the underlying manifold, to enforce the invariance of the classifier along manifold. And the *normal adversarial regularization* (NAR) penalizes the deviation of the classifier along directions orthogonal to the tangent space, to impose robustness on the classifier against the noise carried in observed data. The two regularization terms enforce different aspects of the classifier's smoothness and jointly improve the generalization performance, as demonstrated in Section 4.

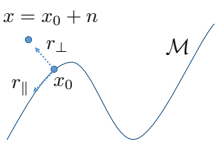

Figure 1: Illustration for tangent-normal adversarial regularization. $x = x_0 + n$ is the observed data, where x_0 is exactly supported on the underlying manifold $\mathcal{M}$ and n is the noise independent of x_0. $r_\parallel$ is the adversarial perturbation along the tangent space to induce invariance of the classifier on manifold; $r_\perp$ is the adversarial perturbation along the normal space to impose robustness on the classifier against noise n.

To realize our idea, we have two challenges to conquer: how to estimate the underlying manifold and how to efficiently perform TNAR.

For the first issue, we take advantage of the generative models equipped with an extra encoder, to characterize coordinate chart of manifold (Kumar et al., 2017; Lecouat et al., 2018; Qi et al., 2018). More specifically, in this work we choose variational autoendoer (VAE) (Kingma & Welling, 2013) and localized GAN (Qi et al., 2018) to estimate the underlying manifold from data.

For the second problem, we develop an adversarial regularization approach based on virtual adversarial training (VAT) (Miyato et al., 2017). Different from VAT, we perform virtual adversarial training in tangent space and normal space separately as illustrated in Figure 1, which leads to a number of new technical difficulties and we will

elaborate the corresponding solutions later. Compared with the traditional manifold regularization methods based on tangent propagation (Simard et al., 1998; Kumar et al., 2017) or manifold Laplacian norm (Belkin et al., 2006; Lecouat et al., 2018), our realization does not require explicitly evaluating the Jacobian of classifier. All we need is to calculate the derivative of matrix vector product, which only costs a few times of back or forward propagation of network.

2 BACKGROUND

2.1 NOTATIONS

We denote the labeled and unlabeled dataset as $\mathcal{D}_l = \{(x_l, y_l)\}$ and $\mathcal{D}_{ul} = \{x_{ul}\}$ respectively, thus $\mathcal{D} := \mathcal{D}_l \cup \mathcal{D}_{ul}$ is the full dataset. The output of classification model is written as $p(y|x, \theta)$, where θ is the model parameters to be trained. We use $\ell(\cdot, \cdot)$ to represent supervised loss function. And the regularization term is denoted as $\mathcal{R}$ with specific subscript for distinction. The observed space of x is written as $\mathbb{R}^D$. And the underlying manifold of the observed data x is written as $\mathcal{M} \cong \mathbb{R}^d, d \ll D$. We use z for the manifold representation of data x. We denote the decoder, or the generator, as $x = g(z)$ and the encoder as $z = h(x)$, which form the coordinate chart of manifold together. If not stated otherwise, we always assume x and z correspond to the coordinate of the same data point in observed space $\mathbb{R}^D$ and on manifold $\mathcal{M}$, i.e., $g(z) = x$ and $h(x) = z$. The tangent space of $\mathcal{M}$ at point x is $T_x\mathcal{M} = J_z g(\mathbb{R}^d) \cong \mathbb{R}^d$, where $J_z g$ is the Jacobian of g at point z. The tangent space $T_x\mathcal{M}$ is also the span of the columns of $J_z g$. For convenience, we define $J := J_z g$.

The perturbation in the observed space $\mathbb{R}^D$ is denoted as $r \in \mathbb{R}^D$, while the perturbation on the manifold representation is denoted as $\eta \in \mathbb{R}^d$. Hence the perturbation on manifold is $g(z + \eta) - g(z) \in \mathbb{R}^D$. When the perturbation η is small enough for the holding of the first order Taylor's expansion, the perturbation on manifold is approximately equal to the perturbation on its tangent space, $g(z + \eta) - g(z) \approx J \cdot \eta \in T_x\mathcal{M}$. Therefore we say a perturbation $r \in \mathbb{R}^D$ is actually on manifold, if there is a perturbation $\eta \in \mathbb{R}^d$, such that $r = J \cdot \eta$.

2.2 VIRTUAL ADVERSARIAL TRAINING

VAT (Miyato et al., 2017) is an effective regularization method for SSL. The virtual adversarial loss introduced in VAT is defined by the robustness of the classifier against local perturbation in the input space $\mathbb{R}^D$. Hence VAT imposes a kind of smoothness condition on the classifier. Mathematically, the virtual adversarial loss in VAT for SSL is $L(\mathcal{D}_l, \mathcal{D}_{ul}, \theta) := \mathbb{E}_{(x_l, y_l) \in \mathcal{D}_l} \ell(y_l, p(y|x_l, \theta)) + \alpha \mathbb{E}_{x \in \mathcal{D}} \mathcal{R}_{\text{vat}}(x, \theta)$, where the VAT regularization $\mathcal{R}_{\text{vat}}$ is defined as $\mathcal{R}_{\text{vat}}(x; \theta) := \max_{\|r\|_2 \le \epsilon} dist(p(y|x, \theta), p(y|x + r, \theta))$, where $dist(\cdot, \cdot)$ is some distribution distance measure and ϵ controls the magnitude of the adversarial example. For simplicity, define

$$F(x, r, \theta) := dist(p(y|x, \theta), p(y + r, \theta)). \tag{1}$$

Then $\mathcal{R}_{\text{vat}} = \max_{\|r\|_2 \le \epsilon} F(x, r, \theta)$. The so called virtual adversarial example is $r^* := \arg\max_{\|r\| \le \epsilon} F(x, r, \theta)$. Once we have r^*, the VAT loss can be optimized with the objective as $L(\mathcal{D}_l, \mathcal{D}_{ul}, \theta) = \mathbb{E}_{(x_l, y_l) \in \mathcal{D}_l} \ell(y_l, p(y|x_l, \theta)) + \alpha \mathbb{E}_{x \in \mathcal{D}} F(x, r^*, \theta)$.

To obtain the virtual adversarial example r^*, Miyato et al. (2017) suggested to apply second order Taylor's expansion to $F(x, r, \theta)$ around $r = 0$ as

$$F(x, r, \theta) \approx \frac{1}{2} r^T H r, \tag{2}$$

where $H := \nabla_r^2 F(x, r, \theta)|_{r=0}$ denotes the Hessian of F with respect to r. The vanishing of the first two terms in Taylor's expansion occurs because that $dist(\cdot, \cdot)$ is a distance measure with minimum zero and $r = 0$ is the corresponding optimal value, indicating that at $r = 0$, both the value and the gradient of $F(x, r, \theta)$ are zero. Therefore for small enough ϵ, $r^* \approx \arg\max_{\|r\|_2 \le \epsilon} \frac{1}{2} r^T H r$, which is an eigenvalue problem and the direction of r^* can be solved by power iteration.

2.3 GENERATIVE MODELS FOR DATA MANIFOLD

We take advantage of generative model with both encoder h and decoder g to estimate the underlying data manifold $\mathcal{M}$ and its tangent space $T_x\mathcal{M}$. As assumed by previous works (Kumar et al., 2017;

Lecouat et al., 2018), perfect generative models with both decoder and encoder can describe the data manifold, where the decoder $g(z)$ and the encoder $h(x)$ together serve as the coordinate chart of manifold $\mathcal{M}$. Note that the encoder is indispensable for it helps to identify the manifold coordinate $z = h(x)$ for point $x \in \mathcal{M}$. With the trained generative model, the tangent space is given by $T_x\mathcal{M} = J_z g(\mathbb{R}^d)$, or the span of the columns of $J = J_z g$.

In this work, we adopt VAE (Kingma & Welling, 2013) and localized GAN (Qi et al., 2018) to learn the targeted underlying data manifold $\mathcal{M}$ as summarized below.

VAE VAE (Kingma & Welling, 2013) is a well known generative model consisting of both encoder and decoder. The training of VAE is by optimizing the variational lower bound of log likelihood,

$$\log p(x, \theta) \geq \mathbb{E}_{z\sim q(z|x,\theta)} \left[\log p(x|z, \theta)\right] - KL(q(z|x, \theta)\|p(z)). \tag{3}$$

Here $p(z)$ is the prior of hidden variable z, and $q(z|x, \theta)$, $p(x|z, \theta)$ models the encoder and decoder in VAE, respectively. The derivation of the lower bound with respect to θ is well defined thanks to the reparameterization trick, thus it could be optimized by gradient based method. The lower bound could also be interpreted as a reconstruction term plus a regularization term (Kingma & Welling, 2013). With a trained VAE, the encoder and decoder are given as $h(x) = \arg\max_z q(z|x)$ and $g(z) = \arg\max_x q(x|z)$ accordingly.

Localized GAN Localized GAN (Qi et al., 2018) suggests to use a localized generator $G(x, z)$ to replace the global generator $g(z)$ in vanilla GAN Goodfellow et al. (2014a). The key difference between localized GAN and previous generative model for manifold is that, localized GAN learns a distinguishing local coordinate chart for each point $x \in \mathcal{M}$, which is given by $G(x, z)$, rather than one global coordinate chart. To model the local coordinate chart in data manifold, localized GAN requires the localized generator to satisfy two more regularity conditions: 1) *locality:* $G(x, 0) = x$, so that $G(x, z)$ is localized around x; 2) *orthogonmality:* $\left(\frac{\partial G(x,z)}{\partial z}\right)^T \frac{\partial G(x,z)}{\partial z} = I$, to ensure $G(x, z)$ is non-degenerated. The two conditions are achieved by the following penalty during training of localized GAN:

$$\mathcal{R}_{\text{localized GAN}} := \mu_1 \big\|G(x, 0) - x\big\|^2 + \mu_2 \left\| \left(\frac{\partial G(x, z)}{\partial z}\right)^T \frac{\partial G(x, z)}{\partial z} - I \right\|^2.$$

Since $G(x, z)$ defines a local coordinate chart for each x separately, in which the latent encode of x is $z = 0$, there is no need for the extra encoder to provide the manifold representation of x.

3 METHOD

In this section we elaborate our proposed *tangent-normal adversarial regularization* (TNAR) strategy. The TNAR loss to be minimized for SSL is

$$L(D_l, D_{ul}, \theta) := \mathbb{E}_{(x_l,y_l)\in\mathcal{D}_l} \ell\left(y_l, p(y|x_l, \theta)\right) + \alpha_1 \mathbb{E}_{x\in\mathcal{D}} \mathcal{R}_{\text{tangent}}(x, \theta) + \alpha_2 \mathbb{E}_{x\in\mathcal{D}} \mathcal{R}_{\text{normal}}(x, \theta). \tag{4}$$

The first term in Eq. (4) is a common used supervised loss. $\mathcal{R}_{\text{tangent}}$ and $\mathcal{R}_{\text{normal}}$ is the so called *tangent adversarial regularization* (TAR) and *normal adversarial regularization* (NAR) accordingly, jointly forming the proposed TNAR. We assume that we already have a well trained generative model for the underlying data manifold $\mathcal{M}$, with encoder h and decoder g, which can be obtained as described in Section 2.3.

3.1 TANGENT ADVERSARIAL REGULARIZATION

Vanilla VAT penalizes the variety of the classifier against local perturbation in the input space $\mathbb{R}^D$ (Miyato et al., 2017), which might overly regularize the classifier, since the semi-supervised learning assumption only indicates that the true conditional distribution varies smoothly along the underlying manifold $\mathcal{M}$, but not the whole input space $\mathbb{R}^D$ (Belkin et al., 2006; Rifai et al., 2011; Niyogi, 2013). To avoid this shortcoming of vanilla VAT, we propose the tangent adversarial regularization (TAR), which restricts virtual adversarial training to the tangent space of the underlying manifold $T_x\mathcal{M}$, to enforce manifold invariance property of the classifier.

$$\mathcal{R}_{\text{tangent}}(x; \theta) := \max_{\|r\|_2\leq\epsilon, r\in T_x\mathcal{M}=J_z g(\mathbb{R}^d)} F(x, r, \theta), \tag{5}$$

where $F(x, r, \theta)$ is defined as in Eq. (1). To optimize Eq. (5), we first apply Taylor's expansion to $F(x, r, \theta)$ so that $\mathcal{R}_{\text{tangent}}(x; \theta) \approx \max_{\|r\|_2 \leq \epsilon, r \in T_x\mathcal{M} = J_z g(\mathbb{R}^d)} \frac{1}{2} r^T H r$, where the notations and the derivation are as in Eq. (2). We further reformulate $\mathcal{R}_{\text{tangent}}$ as

$$\underset{r \in \mathbb{R}^D}{\text{maximize}} \quad \frac{1}{2} r^T H r, \qquad \text{s.t.} \quad \|r\|_2 \leq \epsilon, \quad r = J\eta. \quad (\eta \in \mathbb{R}^d, J := J_z g \in \mathbb{R}^{D \times d}, H \in \mathbb{R}^{D \times D}) \tag{6}$$

Or equivalently,

$$\underset{\eta \in \mathbb{R}^d}{\text{maximize}} \quad \frac{1}{2} \eta^T J^T H J \eta, \qquad \text{s.t.} \quad \eta^T J^T J \eta \leq \epsilon^2. \tag{7}$$

This is a classic generalized eigenvalue problem, the optimal solution η^* of which could be obtained by power iteration and conjugate gradient (and scaling). The iteration framework is as

$$v \leftarrow J^T H J \eta; \qquad \mu \leftarrow (J^T J)^{-1} v; \qquad \eta \leftarrow \mu / \|\mu\|_2 . \tag{8}$$

Now we elaborate the detailed implementation of each step in Eq. (8).

Computing $J^T H J \eta$. Note that $z = h(x), x = g(z)$. Define $r(\eta) := g(z + \eta) - g(z)$. For $F\big(x, r(\eta), \theta\big) = dist(p(y|x, \theta) \| p(y|x + r(\eta), \theta))$, we have $\nabla^2_\eta F(x, r(\eta), \theta) = (J_{z+\eta} g)^T \nabla^2_r F(x, r(\eta), \theta)(J_{z+\eta} g) + \nabla^2_\eta g(z + \eta) \cdot \nabla_r F(x, r(\eta), \theta)$. While on the other hand, since $dist(\cdot, \cdot)$ is some distance measure with minimum zero and $r(0) = 0$ is the corresponding optimal value, we have $F(x, r(0), \theta) = 0, \nabla_r F(x, r(0), \theta) = 0$. Therefore, $\nabla^2_\eta F(x, r(0), \theta) = (J_z g)^T \nabla^2_r F(x, r(0), \theta) J_z g = J^T H J$. Thus the targeted matrix vector product could be efficiently computed as $J^T H J \eta = \nabla^2_\eta F(x, r(0), \theta) \cdot \eta = \nabla_\eta \big(\nabla_\eta F(x, r(0), \theta) \cdot \eta\big)$. Note that $\nabla_\eta F(x, r(0), \theta) \cdot \eta$ is a scalar, hence the gradient of which could be obtained by back propagating the network for once. And it only costs twice back propagating for the computation of $J^T H J \eta$.

Solving $J^T J \mu = v$. Similarly, define $K(\eta) := \big(g(z + \eta) - g(z)\big)^T \big(g(z + \eta) - g(z)\big)$. We have $\nabla^2_\eta K(\eta) = (J_{z+\eta} g)^T J_{z+\eta} g + \nabla^2_\eta g(z + \eta) \cdot K(\eta)$. Since $K(0) = 0$, we have $\nabla^2_\eta K(0) = (J_z g)^T J_z g = J^T J$. Thus the matrix vector product $J^T J \mu$ could be evaluated similarly as $J^T J \mu = \nabla_\eta \big(\nabla_\eta K(0) \cdot \mu\big)$. The extra cost for evaluating $J^T J \mu$ is still back propagating the network for twice. Due to $J^T J$ being positive definite (g is non-degenerated), we can apply several steps of conjugate gradient to solve $J^T J \mu = v$ efficiently.

By iterating Eq. (8), we obtain the optimal solution $\eta_\|$ of Eq. (7). The desired optimal solution is then $r_\| = \epsilon J \eta_\| / \|J \eta_\|\|$, using which we obtain $\mathcal{R}_{\text{tangent}}(x; \theta) = F(x, r_\|, \theta)$.

Compared with manifold regularization based on tangent propagation (Simard et al., 1998; Kumar et al., 2017) or manifold Laplacian norm (Belkin et al., 2006; Lecouat et al., 2018), which is computationally inefficient due to the evaluation of Jacobian, our proposed TAR could be efficiently implemented, thanks to the low computational cost of virtual adversarial training.

3.2 Normal adversarial regularization

Motivated by the noisy observation assumption indicating that the observed data contains noise driving them off the underlying manifold, we come up with the normal adversarial regularization (NAR) to enforce the robustness of the classifier against such noise, by performing virtual adversarial training in the normal space. The mathematical description is

$$\mathcal{R}_{\text{normal}}(x; \theta) := \max_{\|r\|_2 \leq \epsilon, r \perp T_x\mathcal{M}} F(x, r, \theta) \approx \max_{\|r\|_2 \leq \epsilon, r \perp T_x\mathcal{M}} \frac{1}{2} r^T H r. \tag{9}$$

Note that $T_x\mathcal{M}$ is spanned by the coloums of $J = J_z g$, thus $r \perp T_x\mathcal{M} \Leftrightarrow J^T \cdot r = 0$. Therefore we could reformulate Eq. (9) as

$$\underset{r \in \mathbb{R}^D}{\text{maximize}} \quad \frac{1}{2} r^T H r, \qquad \text{s.t.} \quad \|r\|_2 \leq \epsilon, \quad J^T \cdot r = 0. \tag{10}$$

However, Eq. (10) is not easy to optimize since $J^T \cdot r$ cannot be efficiently computed. To overcome this, instead of requiring r being orthogonal to the whole tangent space $T_x\mathcal{M}$, we take a step back to

demand r being orthogonal to only one specific tangent direction, i.e., the tangent space adversarial perturbation $r_{\|}$. Thus the constraint $J^T \cdot r = 0$ is relaxed to $(r_{\|})^T \cdot r = 0$. And we further replace the constraint by a regularization term,

$$\underset{r \in \mathbb{R}^D}{\text{maximize}} \quad \frac{1}{2} r^T H r - \lambda r^T (r_{\|} r_{\|}^T) r, \qquad \text{s.t.} \quad \|r\|_2 \leq \epsilon, \tag{11}$$

where λ is a hyperparameter introduced to control the orthogonality of r.

Since Eq. (11) is again an eigenvalue problem, and we can apply power iteration to solve it. Note that a small identity matrix $\lambda \|r_{\|}\| I$ is needed to be added to keep $\frac{1}{2} H - \lambda r_{\|} r_{\|}^T + \lambda \|r_{\|}\| I$ semi-positive definite, which does not change the optimal solution of the eigenvalue problem. The power iteration is as

$$r \leftarrow \frac{1}{2} H r - \lambda (r_{\|})^T r_{\|} r + \lambda \|r_{\|}\| r. \tag{12}$$

And the evaluation of Hr is by $Hr = \nabla_r \left(\nabla_r F(x, 0, \theta) \cdot r \right)$, which could be computed efficiently. After finding the optimal solution of Eq. (11) as $r_{\perp}$, the NAR becomes $\mathcal{R}_{\text{normal}}(x, \theta) = F(x, r_{\perp}, \theta)$.

Finally, as in (Miyato et al., 2017), we add *entropy regularization* to our loss function. It ensures neural networks to output a more determinate prediction and has implicit benefits for performing virtual adversarial training, $\mathcal{R}_{\text{entropy}}(x, \theta) := -\sum_y p(y|x, \theta) \log p(y|x, \theta)$. Our final loss for SSL is

$$\begin{aligned} L(D_l, D_{ul}, \theta) :=& \mathbb{E}_{(x_l, y_l) \in \mathcal{D}_l} \ell \left(y_l, p(y|x_l, \theta) \right) + \alpha_1 \mathbb{E}_{x \in \mathcal{D}} \mathcal{R}_{\text{tangent}}(x, \theta) \\ &+ \alpha_2 \mathbb{E}_{x \in \mathcal{D}} \mathcal{R}_{\text{normal}}(x, \theta) + \alpha_3 \mathbb{E}_{x \in \mathcal{D}} \mathcal{R}_{\text{entropy}}(x, \theta). \end{aligned} \tag{13}$$

The TAR inherits the computational efficiency from VAT and the manifold invariance property from traditional manifold regularization. The NAR causes the classifier for SSL being robust against the off manifold noise contained in the observed data. These advantages make our proposed TNAR, the combination of TAR and NAR, a reasonable regularization method for SSL, the superiority of which will be shown in the experiment part in Section 4.

4 EXPERIMENTS

To demonstrate the advantages of our proposed TNAR for SSL, we conduct a series of experiments on both artificial and real dataset. The compared methods for SSL include: 1) SL: supervised learning using only the labeled data; 2) VAT: vanilla VAT (Miyato et al., 2017); 3) TNAR-VAE: the proposed TNAR method, with the underlying manifold estimated by VAE; 4) TNAR-LGAN: the proposed TNAR method, with the underlying manifold estimated by localized GAN; 5) TNAR-Manifold: the proposed TNAR method with oracle underlying manifold for the observed data, only used for artificial dataset; 6) TNAR-AE: the proposed TNAR method, with the underlying manifold estimated roughly by autoendoer, only used for artificial dataset. 7) TAR: the tangent adversarial regularization, used in ablation study. 8) NAR: the normal adversarial regularization, used in ablation study. If not stated otherwise, all the above methods contain entropy regularization term.

4.1 TWO-RINGS ARTIFICIAL DATASET

We introduce experiments on a two-rings artificial dataset to show the effectiveness of our proposed methods intuitively. In this experiments, there is $3,000$ unlabeled data (gray dots) and 6 labeled data (blue dots), 3 for each class. The detailed construction could be found in Appendix.

The performance of each compared methods is shown in Table 1, and the corresponding classification boundary is demonstrated in Figure 2. The TNAR under true underlying manifold (TNAR-Manifold) perfectly classifies the two-rings dataset with merely 6 labeled data, while the other methods fail to predict the correct decision boundary. Even with the underlying manifold roughly approximated by an autoendoer, our approach (TNAR-AE) outperforms VAT in this artificial dataset. However, the performance of TNAR-AE is worse than TNAR-Manifold, indicating that the effectiveness of TNAR relies on the quality of estimating the underlying manifold.

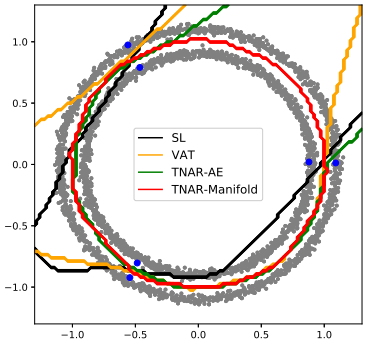

Figure 2: The decision boundaries of compared methods on two-rings artificial dataset. Gray dots distributed on two rings: the unlabeled data. Blue dots (3 in each ring): the labeled data. Colored curves: the decision boundaries found by compared methods.

Table 1: Classification errors (%) of compared methods on two-ring artificial dataset. We test with and without entropy regularization in each method and report the best one. In VAT and TNAR-AE, without entropy regularization is better; For TNAR-Manifold, adding entropy regularization is better.

Model	Error (%)
SL	32.95
VAT	23.80
TNAR-AE	12.45
TNAR-Manifold	9.90
TNAR-Manifold (ent)	0

4.2 FASHIONMNIST

We also conduct experiments on FashionMNIST dataset[1]. There are three sets of experiments with the number of labeled data being 100, 200 and $1,000$, respectively. The details about the networks are in Appendix.

The corresponding results are shown in Table 2, from which we observe at least two phenomena. The first is that our proposed TANR methods (TNAR-VAE, TNAR-LGAN) achieve lower classification errors than VAT in all circumstances with different number of labeled data. The second is that the performance of our method depends on the estimation of the underlying manifold of the observed data. In this case, TNAR-VAE brings larger improvement than TNAR-LGAN, since VAE produces better diverse examples according to our observation. As the development of generative model capturing more accurate underlying manifold, it is expected that our proposed regularization strategy benefits more for SSL.

Table 2: Classification errors (%) of compared methods on FashionMNIST dataset.

Method	100 labels	200 labels	1000 labels
VAT	27.69	20.85	14.51
TNAR/TAR/NAR-LGAN	23.65/24.87/28.73	18.32/19.16/24.49	13.52/14.09/15.94
TNAR/TAR/NAR-VAE	**23.35**/26.45/27.83	**17.23**/20.53/24.81	**12.86**/14.02/15.44

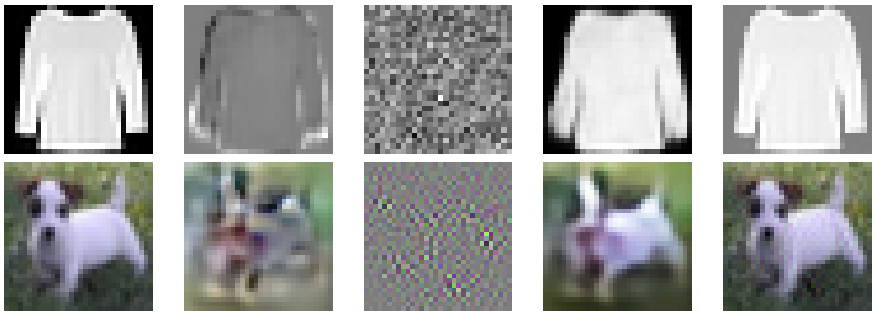

Figure 3: The perturbations and adversarial examples in the tangent space and the normal space. Note that the perturbations is actually too small to distinguish easily, thus we show the scaled perturbations. First row: FashionMNIST dataset; Second row: CIFAR-10 dataset. From left to right: original example, tangent adversarial perturbation, normal adversarial perturbation, tangent adversarial example, normal adversarial example.

[1] https://github.com/zalandoresearch/fashion-mnist

4.3 ABLATION STUDY

We conduct ablation study on FashionMNIST datasets to demonstrate that both of the two regularization terms in TNAR are crucial for SSL. The results are reported in Table 2. Removing either tangent adversarial regularization (NAR) or normal adversarial regularization (TAR) will harm the final performance, since they fail to enforce the manifold invariance or the robustness against the off-manifold noise. Furthermore, the adversarial perturbations and adversarial examples are shown in Figure 3. We can easily observe that the tangent adversarial perturbation focuses on the edges of foreground objects, while the normal space perturbation mostly appears as certain noise over the whole image. This is consistent with our understanding on the role of perturbation along the two directions that capture the different aspects of smoothness.

4.4 CIFAR-10 AND SVHN

There are two classes of experiments for demonstrating the effectiveness of TNAR in SSL, SVHN with $1,000$ labeled data, and CIFAR-10 with $4,000$ labeled data. The experiment setups are identical with Miyato et al. (2017). We test two kinds of convolutional neural networks as classifier (denoted as "small" and "large") as in Miyato et al. (2017). Since it is difficult to obtain satisfying VAE on CIFAR-10, we only conduct the proposed TNAR with the underlying manifold identified by Localized GAN (TNAR-LGAN) for CIFAR-10. Note that in Miyato et al. (2017), the authors applied ZCA as pre-processing procedure, while other compared methods do not use this trick. For fair comparison, we only report the performance of VAT without ZCA. More detailed experimental settings are included in Appendix.

Table 3: Classification errors (%) of compared methods on SVHN / CIFAR-10 dataset.

Method	SVHN 1,000 labels	CIFAR-10 4,000 labels
VAT (small)	4.37	15.67
VAT (large)	4.23	15.29
ALI (Dumoulin et al., 2016)	7.41	17.99
Improved GAN (Salimans et al., 2016)	8.11	18.63
Tripple GAN (Li et al., 2017)	5.77	16.99
FM GAN (Kumar et al., 2017)	4.39	16.20
LGAN (Qi et al., 2018)	4.73	14.23
TNAR-VAE (small)	3.93	-
TNAR-VAE (large)	**3.84**	-
TNAR-LGAN (small)	4.10	13.63
TNAR-LGAN (large)	3.93	**13.53**

In Table 3 we report the experiments results on CIFAR-10 and SVHN, showing that our proposed TNAR outperforms other state-of-the-art SSL methods on both SVHN and CIFAR-10, demonstrating the superiority of our proposed TNAR.

5 CONCLUSION

We present the tangent-normal adversarial regularization, a novel regularization strategy for semi-supervised learning, composing of regularization on the tangent and normal space separately. The tangent adversarial regularization enforces manifold invariance of the classifier, while the normal adversarial regularization imposes robustness of the classifier against the noise contained in the observed data. Experiments on artificial dataset and multiple practical datasets demonstrate that our approach outperforms other state-of-the-art methods for semi-supervised learning. The performance of our method relies on the quality of the estimation of the underlying manifold, hence the breakthroughs on modeling data manifold could also benefit our strategy for semi-supervised learning, which we leave as future work.

REFERENCES

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, 7 (Nov):2399–2434, 2006.

Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising auto-encoders as generative models. In *Advances in Neural Information Processing Systems*, pp. 899–907, 2013.

Lawrence Cayton. Algorithms for manifold learning. *Univ. of California at San Diego Tech. Rep*, 12(1-17):1, 2005.

Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

Zihang Dai, Zhilin Yang, Fan Yang, William W Cohen, and Ruslan R Salakhutdinov. Good semi-supervised learning that requires a bad gan. In *Advances in Neural Information Processing Systems*, pp. 6510–6520, 2017.

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014a.

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

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pp. 3581–3589, 2014.

Abhishek Kumar, Prasanna Sattigeri, and Tom Fletcher. Semi-supervised learning with gans: Manifold invariance with improved inference. In *Advances in Neural Information Processing Systems*, pp. 5540–5550, 2017.

Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016.

Bruno Lecouat, Chuan-Sheng Foo, Houssam Zenati, and Vijay R Chandrasekhar. Semi-supervised learning with gans: Revisiting manifold regularization. *arXiv preprint arXiv:1805.08957*, 2018.

Chongxuan Li, Kun Xu, Jun Zhu, and Bo Zhang. Triple generative adversarial nets. *arXiv preprint arXiv:1703.02291*, 2017.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*, 2016.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *arXiv preprint arXiv:1704.03976*, 2017.

Hariharan Narayanan and Sanjoy Mitter. Sample complexity of testing the manifold hypothesis. In *Advances in Neural Information Processing Systems*, pp. 1786–1794, 2010.

Partha Niyogi. Manifold regularization and semi-supervised learning: Some theoretical analyses. *The Journal of Machine Learning Research*, 14(1):1229–1250, 2013.

Augustus Odena. Semi-supervised learning with generative adversarial networks. *arXiv preprint arXiv:1606.01583*, 2016.

Guo-Jun Qi, Liheng Zhang, Hao Hu, Marzieh Edraki, Jingdong Wang, and Xian-Sheng Hua. Global versus localized generative adversarial nets. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *Advances in Neural Information Processing Systems*, pp. 3546–3554, 2015.

Salah Rifai, Yann N Dauphin, Pascal Vincent, Yoshua Bengio, and Xavier Muller. The manifold tangent classifier. In *Advances in Neural Information Processing Systems*, pp. 2294–2302, 2011.

Alan E Robinson, Paul S Hammon, and Virginia R de Sa. Explaining brightness illusions using spatial filtering and local response normalization. *Vision research*, 47(12):1631–1644, 2007.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pp. 2234–2242, 2016.

Patrice Y Simard, Yann A LeCun, John S Denker, and Bernard Victorri. Transformation invariance in pattern recognitiontangent distance and tangent propagation. In *Neural networks: tricks of the trade*, pp. 239–274. Springer, 1998.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pp. 912–919, 2003.

A TWO-RINGS DATASET

The underlying manifold for two-rings data is given by $\mathcal{M} = \mathcal{M}_+ \cup \mathcal{M}_-$, where $\mathcal{M}_+ = \left\{(x_1, x_2) \,|x_1^2 + x_2^2 = 0.9^2\right\}$ and $\mathcal{M}_- = \left\{(x_1, x_2) \,|x_1^2 + x_2^2 = 1.1^2\right\}$ represent two different classes. The observed data is sampled as $x = x_0 + n$, where x_0 is uniformly sampled from $\mathcal{M}$ and $n \sim \mathcal{N}(0, 2^{-2})$. We sample 6 labeled training data, 3 for each class, and $3,000$ unlabeled training data, as shown in Figure 2.

B EXPERIMENTS DETAILS ON FASHIONMNIST

In FashionMNIST[2] experiments, we preserver $1,00$ data for validation from the original training dataset. That is, we use $100/200/1,000$ labeled data for training and the other 100 labeled data for validation. For pre-processing, we scale images into $0 \sim 1$. The classification neural network is as following. (a, b) means the convolution filter is with $a \times a$ shape and b channels. The max pooling layer is with stride 2. And we apply local response normalization (LRN) (Robinson et al., 2007). The number of hidden nodes in the first fully connected layer is 512.

$$\begin{aligned}&\text{Conv}(3, 32) \to \text{ReLU} \to \text{Conv}(3, 32) \to \text{ReLU} \to \text{MaxPooling} \to \text{LRN}\\ &\to \text{Conv}(3, 64) \to \text{ReLU} \to \text{Conv}(3, 64) \to \text{ReLU} \to \text{MaxPooling} \to \text{LRN}\\ &\to \text{FC1} \to \text{ReLU} \to \text{FC2}\end{aligned}$$

For the labeled data, the batch size is 32, and for the unlabeled data, the batch size is 128. All networks are trained for $12,000$ updates. The optimizer is ADAM with initial learning rate 0.001, and linearly decay over the last $4,000$ updates. The hyperparameters tuned is the magnitude of the tangent adversarial perturbation (ϵ_1), the magnitude of the normal adversarial perturbation (ϵ_2) and the hyperparameter λ in Eq. (11). Other hyperparameters are all set to 1. We tune λ from $\{1, 0.1, 0.01, 0.001\}$, and ϵ_1, ϵ_2 randomly from $[0.05, 20]$.

[2] https://github.com/zalandoresearch/fashion-mnist

Table 4: The structure of convolutional neural networks for experiments on CIFAR-10 and SVHN, based on Springenberg et al. (2014); Salimans et al. (2016); Laine & Aila (2016). All the convolutional layers and fully connected layers are followed by batch normalization except the fully connected layer on CIFAR-10. The slopes of all lReLU functions in the networks are 0.1.

Conv-Small on SVHN	Conv-Small on CIFAR-10	Conv-Large
32×32 RGB image		
3×3 conv. 64 lReLU	3×3 conv. 96 lReLU	3×3 conv. 128 lReLU
3×3 conv. 64 lReLU	3×3 conv. 96 lReLU	3×3 conv. 128 lReLU
3×3 conv. 64 lReLU	3×3 conv. 96 lReLU	3×3 conv. 128 lReLU
2×2 max-pool, stride 2 dropout, $p = 0.5$		
3×3 conv. 128 lReLU	3×3 conv. 192 lReLU	3×3 conv. 256 lReLU
3×3 conv. 128 lReLU	3×3 conv. 192 lReLU	3×3 conv. 256 lReLU
3×3 conv. 128 lReLU	3×3 conv. 192 lReLU	3×3 conv. 256 lReLU
2×2 max-pool, stride 2 dropout, $p = 0.5$		
3×3 conv. 128 lReLU	3×3 conv. 192 lReLU	3×3 conv. 512 lReLU
1×1 conv. 128 lReLU	1×1 conv. 192 lReLU	1×1 conv. 256 lReLU
1×1 conv. 128 lReLU	1×1 conv. 192 lReLU	1×1 conv. 128 lReLU
global average pool, 6×6 → 1×1		
dense 128 → 10	dense 192→ 10	dense 128→ 10
10-way softmax		

The encoder of the VAE for identify the underlying manifold is a LeNet-like one, with two convolutional layers and one fully connected layer. And the decoder is symmetric with the encoder, except using deconvolutional layers to replace convolutional layer. The latent dimensionality is 128. The localized GAN for identify the underlying manifold is similar as stated in Qi et al. (2018). And the implementation is modified from `https://github.com/z331565360/Localized-GAN`. We change the latent dimensionality into 128.

We tried both joint training the LGAN with the classifier, and training them separately, observing no difference.

C Experiments details on SVHN and CIFAR-10

In SVHN[3] and CIFAR-10[4] experiments, we preserve $1,000$ data for validation from the original training set. That is, we use $1,000/4,000$ labeled data for training and the other $1,000$ labeled data for validation. The only pre-processing on data is to scale the pixels value into $0 \sim 1$. We do not use data augmentation. The structure of classification neural network is shown in Table 4, which is identical as in Miyato et al. (2017).

For the labeled data, the batch size is 32, and for the unlabeled data, the batch size is 128. For SVHN, all networks are trained for $48,000$ updates. And for CIFAR-10, all networks are trained for $200,000$ updates. The optimizer is ADAM with initial learning rate 0.001, and linearly decay over the last $16,000$ updates. The hyperparameters tuned is the magnitude of the tangent adversarial perturbation (ϵ_1), the magnitude of the normal adversarial perturbation (ϵ_2) and the hyperparameter λ in Eq. (11). Other hyperparameters are all set to 1. We tune λ from $\{1, 0.1, 0.01, 0.001\}$, and ϵ_1, ϵ_2 randomly from $[0.05, 20]$.

[3] `http://ufldl.stanford.edu/housenumbers/`

[4] `https://www.cs.toronto.edu/~kriz/cifar.html`

The VAE for identify the underlying manifold for SVHN is implemented as in `https://github.com/axium/VAE-SVHN`. The only modification is we change the coefficient of the regularization term from 0.01 to 1. The localized GAN for identify the underlying manifold for SVHN and CIFAR-10 is similar as stated in Qi et al. (2018). And the implementation is modified from `https://github.com/z331565360/Localized-GAN`. We change the latent dimensionality into 512 for both SVHN and CIFAR-10.

D MORE ADVERSARIAL EXAMPLES

More adversarial perturbations and adversarial examples in tangent space and normal space are shown in Figure 4 and Figure 5.

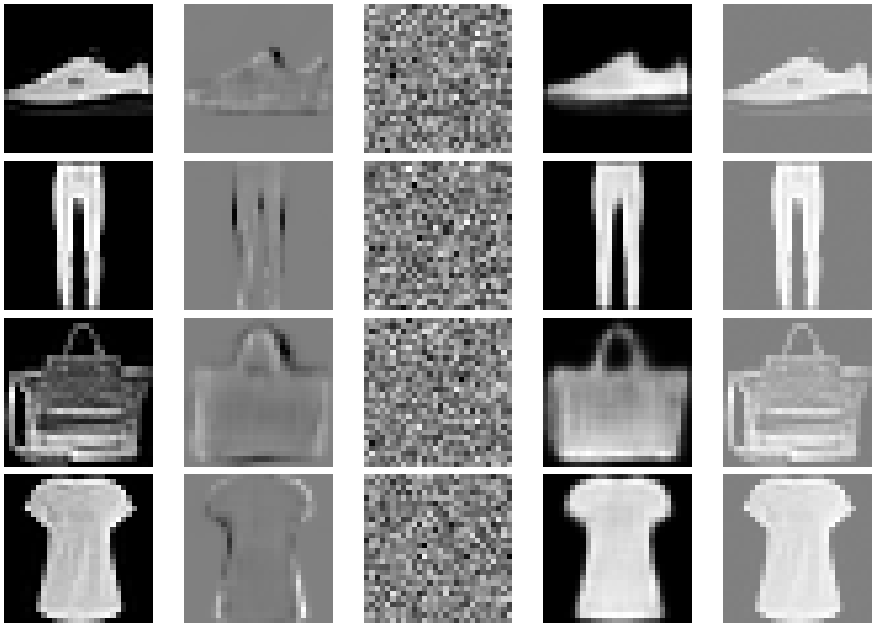

Figure 4: The perturbations and adversarial examples in tangent space and normal space for FashionMNIST dataset. Note that the perturbations is actually too small to distinguish easily, thus we show the scaled perturbations. From left to right: original example, tangent adversarial perturbation, normal adversarial perturbation, tangent adversarial example, normal adversarial example.

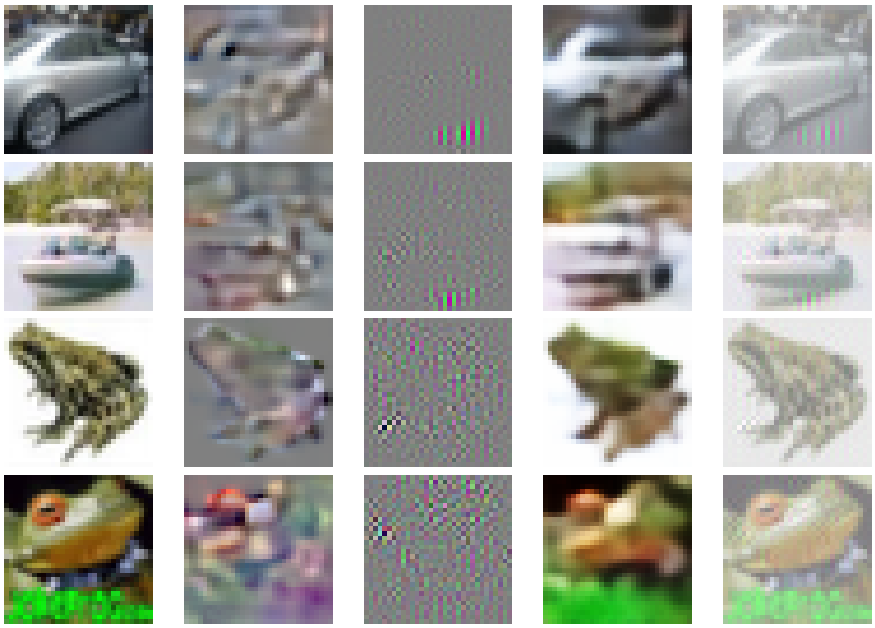

Figure 5: The perturbations and adversarial examples in tangent space and normal space for CIFAR-10 dataset. Note that the perturbations is actually too small to distinguish easily, thus we show the scaled perturbations. From left to right: original example, tangent adversarial perturbation, normal adversarial perturbation, tangent adversarial example, normal adversarial example.