Learning Student Networks with Few Data

Shumin Kong,[1] **Tianyu Guo,**[1,2] **Shan You,**[3] **Chang Xu**[1]
[1]School of Computer Science, Faculty of Engineering, The University of Sydney, Australia
[2]Key Laboratory of Machine Percepton (MOE), CMIC, School of EECS, Peking University, China
[3]SenseTime Research, China
{shumin.kong, c.xu}@sydney.edu.au, tianyuguo@pku.edu.cn, youshan@sensetime.com

Abstract

Recently, the teacher-student learning paradigm has drawn much attention in compressing neural networks on low-end edge devices, such as mobile phones and wearable watches. Current algorithms mainly assume the complete dataset for the teacher network is also available for the training of the student network. However, for real-world scenarios, users may only have access to part of training examples due to commercial profits or data privacy, and severe over-fitting issues would happen as a result. In this paper, we tackle the challenge of learning student networks with few data by investigating the ground-truth data-generating distribution underlying these few data. Taking Wasserstein distance as the measurement, we assume this ideal data distribution lies in a neighborhood of the discrete empirical distribution induced by the training examples. Thus we propose to safely optimize the worst-case cost within this neighborhood to boost the generalization. Furthermore, with theoretical analysis, we derive a novel and easy-to-implement loss for training the student network in an end-to-end fashion. Experimental results on benchmark datasets validate the effectiveness of our proposed method.

Introduction

In recent years, computer vision research has rapidly advanced due to the success of deep neural networks. The image classification performance on large-scale datasets (*e.g.*, ImageNet) has been constantly refreshed by various convolutional neural networks (CNNs), such as AlexNet (Krizhevsky, Sutskever, and Hinton 2012), VGGNet (Simonyan and Zisserman 2014), Inception (Szegedy et al. 2015) and ResNet (He et al. 2016). Language models, such as GPT (Radford et al. 2019), BERT(Devlin et al. 2018) and (Wang, Li, and Smola 2019) has achieved superhuman performance on large textual dataset.

However, to achieve the outstanding classification performance, these networks usually have a large volume of parameters and significant resource consumption. For example, to achieve a top-1 error rate of 22.16%, AlexNet requires more than 232 million parameters and more than 700 million multiplications to implement the prediction. This computational demand limits their application on low-end edge devices, such as mobile phones, tablets and wearable watches.

To minimize the resource required by deep neural networks, several techniques to directly compress existing trained networks are investigated, such as vector quantization (Gong et al. 2014), hash encoding (Chen et al. 2015), weight matrices decomposition (Denton et al. 2014) and using unlabeled data (Tang et al. 2019). Others attempt to design efficient architectures to accelerate the inference speed, such as ResNeXt (Xie et al. 2017), Xception network (Chollet 2017) and MobileNets (Howard et al. 2017). Besides these approaches, teacher-student learning paradigm serves as a complementary scheme to obtain light and efficient neural networks. By treating the pre-trained huge networks as teacher networks, the target small network is thus viewed as the student network and can be guided by the teacher network. Ba and Caruana took a straightforward approach by directly minimizing the Euclidean distance between the feature maps generated from the teacher and student networks. The widely-used knowledge distillation method. Hinton, Vinyals, and Dean made a leap forward by introducing a loss to encourage the student network to learn from the softened outputs of the teacher network. Others also attempt to further boost the performance by using multiple teachers (You et al. 2017) or investigating feature layers, such as FitNet (Romero et al. 2015) and activation boundary loss (Heo et al. 2019).

Current teacher-student learning algorithms usually assume that the complete dataset for training teacher network is also available for the learning of the student network. In real-world scenarios, however, users may only have a few data at hand. For instance, due to the consideration of commercial profits or data privacy, many applications do not open their large training dataset completely but only supply a fraction for verification purposes. The limited data usually induce severe over-fitting issues during the learning of the student network. Another example can be an off-line speech recognition network. In this case, a generic speech recognition network can be trained on a data center, which will be used as a teacher. Then, the end-user can provide a few samples of his or her own speech to train the student network. The resulting student network would be small enough to run on a mobile device while maintaining the high-quality of the teacher network. The combination of benefits from

the teacher-student learning paradigms and the generalization ability enables some large deep learning networks to be deployed to low battery and computing power devices, such as mobile phones. Thus, it is important to fit the network to scarce training examples while maintaining its generalization ability.

To boost the generalization ability of the student network, we propose to explore the ground-truth data-generating distribution. Given the training examples, we assume the ground-truth distribution lies in a neighborhood of the discrete empirical distribution, *i.e.*, uniform distribution over i.i.d. training examples. By dint of Wasserstein distance, we thus propose to safely optimize the worst-case or every possible ground-truth distribution's cost within this neighborhood to boost the generalization as a result. However, the worst-case cost does not have a closed-form and is not appropriate for training a student network in an end-to-end fashion. In this way, we furthermore analyze its upper bound in theory and develop a novel loss function accordingly, called WaGe loss. As a result, via this very loss, the student network can maintain its generalization ability while being trained on a fraction of the training examples. Experimental results on benchmark datasets show the effectiveness of our proposed method, and when the training examples are very limited, our method significantly outperforms other comparison methods.

Problem Formulation

Now we formally introduce the teacher-student learning scheme in compressing neural networks, especially the widely-used knowledge distillation method. Then we make a statement that, given a few data, how to boost the generalization ability of the student network. In this paper, we consider the general multi-class classification problem.

Knowledge Distillation

In the teacher-student learning scheme, the teacher network $\mathcal{N}_T$ usually consists of a large amount of parameters and has powerful classification ability accordingly. In contrast, the student network $\mathcal{N}_S$ is light and small, which has much fewer parameters and is appropriate for the low-end computational devices. The goal of the teacher-student learning scheme is to learn the student network with the help of the pre-trained teacher network, instead of solely from the training data.

To transfer the knowledge from the teacher network into the student network, special training guidance or losses are imposed during the learning of student network, *e.g.*, knowledge distillation (KD) loss (Hinton, Vinyals, and Dean 2015). Denote the training data as $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$ where $\boldsymbol{x}_i \in \mathcal{X} \subset \mathbb{R}^d$ and $\boldsymbol{y}_i \in \mathcal{Y} \subset \{0,1\}^k$. KD loss encourages the student network to have similar softened outputs with that of teacher network,

$$\mathcal{L}_{KD}(\mathcal{N}_S) = \frac{1}{N}\sum_{i=1}^{N}[\mathcal{H}(\hat{\boldsymbol{y}}_i, \boldsymbol{y}_i) + C\mathcal{H}(\boldsymbol{p}_i, \boldsymbol{q}_i)], \quad (1)$$

where $\hat{\boldsymbol{y}}_i \in \mathbb{R}^k$ is the prediction output of the student network, C is a balancing constant and $\mathcal{H}(\cdot,\cdot)$ is the cross-entropy loss to measure the discrepancy between the prediction output vector and the ground-truth label vector. $\boldsymbol{p}_i$ and $\boldsymbol{q}_i$ are called the softened outputs of the teacher network and student network, respectively, which are calculated using their raw output logits $\boldsymbol{o}_i^T$ and $\boldsymbol{o}_i^S$ by softmax function, *i.e.*,

$$\boldsymbol{p}_i = \frac{\exp(\boldsymbol{o}_i^T/T)}{\left\|\exp(\boldsymbol{o}_i^T/T)\right\|_1} \quad \text{and} \quad \boldsymbol{q}_i = \frac{\exp(\boldsymbol{o}_i^S/T)}{\left\|\exp(\boldsymbol{o}_i^S/T)\right\|_1}, \quad (2)$$

where T is a temperature parameter to control the *softness* of the probabilistic prediction outputs. The softened outputs contain more information than the one-hot-code ground-truth label vectors and are supposed to better guide the training of the student network.

Rethinking the Generalization of the Student Network

By dint of knowledge distillation loss, the teacher network's softened outputs act as additional (privileged) information during the learning of student network, and is shown to enhance generalization ability of the student network by improving the learning rate (Lopez-Paz et al. 2015). However, when the number of training examples is limited, the generalization error would be still fairly large, and severe overfitting issues would happen consequently. To handle this problem, we suggest rethinking the generalization of the student network.

Denote the ground-truth data-generating distribution of instances $\boldsymbol{x} \in \mathcal{X}$ as $\mathbb{P}$, *i.e.*, $\boldsymbol{x} \sim \mathbb{P}$. Then the aim of the learning student network is to minimize the following population risk,

$$\mathcal{R}(\mathcal{N}_S) = \mathbb{E}^{\mathbb{P}}[\![\ell(\boldsymbol{x}; \mathcal{N}_S, \mathcal{N}_T)]\!], \quad (3)$$

where $\mathbb{E}^{\mathbb{P}}$ is the expectation over the distribution $\mathbb{P}$. $\ell(\boldsymbol{x}; \mathcal{N}_S, \mathcal{N}_T)$ is a loss encouraging the student network to match with the teacher network. For example, for knowledge distillation in Eq. (1) $\ell(\boldsymbol{x}; \mathcal{N}_S, \mathcal{N}_T) = \mathcal{H}(\boldsymbol{p}_i, \boldsymbol{q}_i)$. Given the dataset $\mathcal{D}^{\mathcal{X}} = \{\boldsymbol{x}_i\}_{i=1}^N$, the corresponding empirical risk goes to

$$\hat{\mathcal{R}}(\mathcal{N}_S) = \frac{1}{N}\sum_{i=1}^{N}\ell(\boldsymbol{x}_i; \mathcal{N}_S, \mathcal{N}_T). \quad (4)$$

In fact, Eq. (4) can also be written into an expectation form. Define a discrete distribution $\mathbb{P}_N$ over $\mathcal{X}$ as

$$p(\boldsymbol{x}) = \begin{cases} 1/N, & \text{if } \boldsymbol{x} \in \{\boldsymbol{x}_i\}_{i=1}^N \\ 0, & \text{otherwise,} \end{cases} \quad (5)$$

which is actually a uniform distribution over all N training data and called *discrete empirical distribution*. Then Eq. (4) equals to

$$\hat{\mathcal{R}}(\mathcal{N}_S) = \mathbb{E}^{\mathbb{P}_N}[\![\ell(\boldsymbol{x}; \mathcal{N}_S, \mathcal{N}_T)]\!]. \quad (6)$$

In this way, the gap between the empirical risk and the population risk (*i.e.*, the generalization error) results from the difference between the ground-truth data-generating distribution $\mathbb{P}$ and the discrete empirical distribution $\mathbb{P}_N$. To shrink the gap, we estimate the $\mathbb{P}$ using $\mathbb{P}_N$, and assume $\mathbb{P}$ lies in

a neighborhood of $\mathbb{P}_N$. Since $\mathbb{P}$ is continuous while $\mathbb{P}_N$ is discrete, to measure their distance and to cover every possible (continuous or discrete) distribution "between" them, we adopt the Wasserstein distance (Arjovsky, Chintala, and Bottou 2017) as the measurement. The neighborhood $\mathcal{B}_\epsilon(\mathbb{P}_N)$ can be thus constructed as

$$\mathcal{B}_\epsilon(\mathbb{P}_N) := \{\mathbb{Q} \in \mathcal{M}(\mathcal{X}) : d_W(\mathbb{P}_N, \mathbb{Q}) \leq \epsilon\}, \quad (7)$$

where $\mathcal{M}(\mathcal{X})$ is the set of all possible distributions $\mathbb{Q}$ over $\mathcal{X}$, and $d_W(\cdot,\cdot)$ is the Wasserstein distance between two probabilistic distributions. The rationale behind the Wasserstein distance over other measurements, such as KL-divergence or maximum mean discrepancy, is three-fold: 1) it can lead to a tractable solution which will be presented in the next Section, 2) it can measure the distance between discrete and continuous distributions, and 3) measure concentration results from (Fournier and Guillin 2015) guarantee that $\mathcal{B}_\epsilon(\mathbb{P}_N)$ could contain the unknown ground-truth data distribution with a high confidence. Hence, Eq. (7) can be viewed as a Wasserstein ball with radius of ϵ that surrounds the distribution $\mathbb{P}_N$. Since the distribution $\mathbb{P}$ lies in this ball, to boost the generalization of the student network, we suggest to safely optimize the risk of all possible ground-truth distributions within the $\mathcal{B}_\epsilon(\mathbb{P}_N)$. This goal equals to improving the worst-case risk of all distributions in $\mathcal{B}_\epsilon(\mathbb{P}_N)$, which is equivalent to minimizing the supremum of the risks (Mohajerin Esfahani and Kuhn 2018), *i.e.*,

$$\sup_{\mathbb{Q}\in\mathcal{B}_\epsilon(\mathbb{P}_N)} \mathbb{E}^{\mathbb{Q}}[\![\ell(\boldsymbol{x}; \mathcal{N}_S, \mathcal{N}_T)]\!]. \quad (8)$$

In this way, training with the objective Eq. (8) would alleviate the over-fitting problem caused by few training data, and boost the generalization and the classification performance of the student network accordingly.

Learning with Few Data

In this section, we introduce our detailed solution to boost the generalization ability of the student network, which is related to a novel loss, called Wasserstein generalization (WaGe) loss, developed from theoretical analysis of formulation Eq. (8). We also have discussed some practical tips during training.

Theoretical Analysis of Eq. (8)

As illustrated above, the generalization ability of the student network can be enhanced by optimizing Eq. (8). Nevertheless, Eq. (8) involves a supremum operation over the Wasserstein ball $\mathcal{B}_\epsilon(\mathbb{P}_N)$, which is not computationally tractable for the training networks in an end-to-end fashion. Inspired by the Majorization Minimization method (Hunter and Lange 2004), we choose to analyze the upper bound of Eq. (8) as a surrogate objective, then we can optimize the upper bound instead.

(Mohajerin Esfahani and Kuhn 2018, Theorem 6.3) presents an upper bound related to the empirical risk $\hat{\mathcal{R}}$.

Theorem 1. *If ℓ is proper, convex and lower semicontinous, there exists an upper bound of Eq.* (8) *for any $\epsilon \geq 0$,* i.e.*,*

$$\sup_{\mathbb{Q}\in\mathcal{B}_\epsilon(\mathbb{P}_N)} \mathbb{E}^{\mathbb{Q}}[\![\ell(\boldsymbol{x})]\!] \leq \psi\epsilon + \frac{1}{N}\sum_{i=1}^{N} \ell(\boldsymbol{x}_i), \quad (9)$$

where $\psi := \sup\{\|\theta\|_* : \ell^*(\theta) < \infty\}$, ℓ^* is the convex conjugate function of ℓ and $\|\cdot\|_*$ is the dual norm of $\|\cdot\|$.

Here for ease of notation, throughout this section we suppress the dependence of the network $\mathcal{N}_S$ and $\mathcal{N}_T$ in loss $\ell(\boldsymbol{x}; \mathcal{N}_S, \mathcal{N}_T)$, and denote it as $\ell(\boldsymbol{x})$. And we adopt the ℓ_2 norm throughout this paper. Next, we focus on the estimation of ψ's value, which is dependent on the loss ℓ itself. We have the following result.

Theorem 2. *If ℓ is $\overline{L}$ - lipschitz continuous on $\mathcal{X}$,* i.e.*, for any $\boldsymbol{x}$ and $\boldsymbol{x}' \in \mathcal{X}$,*

$$\|\ell(\boldsymbol{x}) - \ell(\boldsymbol{x}')\| \leq \overline{L}\,\|\boldsymbol{x} - \boldsymbol{x}'\| \quad (10)$$

holds, then the ψ in Eq. (9) *is upper bounded by*

$$\psi \leq \sup_{\boldsymbol{x}\in\mathcal{X}} \|\nabla_{\boldsymbol{x}}\ell(\boldsymbol{x})\|. \quad (11)$$

Proof. (Mohajerin Esfahani and Kuhn 2018, Proposition 6.5) shows that if ℓ is $\overline{L}$ - lipschitz continuous, then

$$\psi \leq \overline{L}. \quad (12)$$

Furthermore, the Lipschitz constant can be larger to the supremum module of the (sub)gradient over $\mathcal{X}$ (Bertsekas 2009), which completes the proof. □

By substituting Eq. (11) in Eq. (9), we can derive the following inequality

$$\sup_{\mathbb{Q}\in\mathcal{B}_\epsilon(\mathbb{P}_N)} \mathbb{E}^{\mathbb{Q}}[\![\ell(\boldsymbol{x})]\!] \leq \frac{1}{N}\sum_{i=1}^{N} \ell(\boldsymbol{x}_i) + \psi\epsilon \quad (13)$$

$$\leq \frac{1}{N}\sum_{i=1}^{N} \ell(\boldsymbol{x}_i) + \epsilon \sup_{\boldsymbol{x}\in\mathcal{X}} \|\nabla_{\boldsymbol{x}}\ell(\boldsymbol{x})\|. \quad (14)$$

However, calculating the supremum can be computationally expensive, so we just approximate it by using the training examples, *i.e.*, $\max_i \|\nabla_{\boldsymbol{x}_i}\ell(\boldsymbol{x}_i)\|$. Moreover, for the stability of the training, we empirically find that a proxy of averaging the gradients would suffice as (Wang et al. 2015; Hein and Andriushchenko 2017), then the upper bound can be relaxed into

$$\frac{1}{N}\sum_{i=1}^{N} \ell(\boldsymbol{x}_i) + \epsilon\frac{1}{N}\sum_{i=1}^{N} \|\nabla_{\boldsymbol{x}_i}\ell(\boldsymbol{x}_i)\|. \quad (15)$$

Recall that the parameter ϵ controls the radius of the Wasserstein ball centered at the discrete empirical distribution $\mathbb{P}_N$.

Method: WaGe loss

To meet the requirements of the loss ℓ in Theorems 1 and 2, in this paper we adopt the following loss,

$$\ell(\boldsymbol{x}; \mathcal{N}_S, \mathcal{N}_T) = \left\|\boldsymbol{o}^S - \boldsymbol{o}^T\right\|_2^2 := \ell(\boldsymbol{o}^S, \boldsymbol{o}^T), \quad (16)$$

where $\boldsymbol{o}^S \in \mathbb{R}^k$ and $\boldsymbol{o}^T \in \mathbb{R}^k$ are the input $\boldsymbol{x}$'s logits of the student and teacher network, respectively. Note that the loss ℓ for Eq. (15) is not limited to Eq. (16). We use Eq. (16) for it serves a straightforward and easy way to measure the discrepancy between the output from student and the teacher network, and works nicely in our experiment.

Combining the adopted loss Eq. (16) and the upper bound Eq. (15), we can obtain a loss promoting the generalization, *i.e.*,

$$\mathcal{L}_W(\mathcal{N}_S) = \frac{1}{N}\sum_{i=1}^{N}\ell(\boldsymbol{o}_i^S, \boldsymbol{o}_i^T) + \epsilon\frac{1}{N}\sum_{i=1}^{N}\left\|\nabla_{\boldsymbol{x}_i}\ell(\boldsymbol{o}_i^S, \boldsymbol{o}_i^T)\right\|. \tag{17}$$

We call this novel loss Wasserstein Generalization (WaGe) loss, which is motivated by focusing on the generalization within the Wasserstein ball of the discrete empirical distribution. Since WaGe loss does not include supervision signal from the ground-truth labels, following the previous works (Romero et al. 2015; Heo et al. 2019), our proposed WaGe loss should be used in conjunction with the KD loss as

$$\mathcal{L}_{train}(\mathcal{N}_S) = \mathcal{L}_{KD}(\mathcal{N}_S) + \alpha\mathcal{L}_W(\mathcal{N}_S), \tag{18}$$

where α is a hyperparameter balancing the two losses. In this way, the student network can be better guided by the teacher network through distilling its knowledge as well as boosting own generalization.

The gradient term included in Equation (17) can be interpreted as a term that improves the generalization ability when being optimized. Intuitively, the gradient term $\nabla_{\boldsymbol{x}_i}\ell(\boldsymbol{o}_i^S, \boldsymbol{o}_i^T)$ can be viewed as the sensitivity of the network with respect to the input (Simonyan, Vedaldi, and Zisserman 2013). For example, if one single image has a large gradient during training, that means the image needs to be paid attention to. By reducing this term, we are effectively reducing the attention of the network to one single image specifically. In other words, optimizing gradient term can smoothen the decision boundaries trained on $\mathbb{P}_N$, and thus reduce the gap between network output for input data in $\mathbb{P}_N$ and $\mathbb{Q}$.

Most existing deep learning works that involve Wasserstein metric often include the Wasserstein metric as a part of their optimization objectives. However, our proposed solution incorporates the Wasserstein metric as a measurement as the constraint of the optimization.

Optimizing Gradient As Data Augmentation

Training deep neural networks requires large amount of data. However, in case of limited data availability, a simple solution is to perform data augmentation. Image data augmentation involves horizontal or vertical flipping of images and adding noise to the images. We next proceed to show that the inclusion of the gradient term in Eq. (15) can be explained as training the network with this data augmentation mechanism.

Theorem 3. *Consider* $\ell(\boldsymbol{x}; \mathcal{N}_S, \mathcal{N}_T) = \left\|\boldsymbol{o}^S - \boldsymbol{o}^T\right\|_2^2 := \ell(\boldsymbol{o}^S, \boldsymbol{o}^T)$ *as the loss function to train the student network. Given a random perturbation* $\boldsymbol{\gamma} \sim (\mathbf{0}, v\mathbf{I})$ *on the input data* $\boldsymbol{x}$*, then the objective on the noisy input to train the student network would be approximated as*

$$\begin{aligned}
&\mathop{\mathbb{E}}_{p(\boldsymbol{x},\boldsymbol{\gamma})}[\![\ell(\boldsymbol{x}+\boldsymbol{\gamma}; \mathcal{N}_S, \mathcal{N}_T)]\!] \\
&= \mathop{\mathbb{E}}_{p(\boldsymbol{x})}[\![\ell(\boldsymbol{x}; \mathcal{N}_S, \mathcal{N}_T)]\!] + v\mathop{\mathbb{E}}_{p(\boldsymbol{x})}[\![\|\nabla_{\boldsymbol{x}}\mathcal{N}_S(\boldsymbol{x})\|_2^2]\!] + O(v^2) \\
&:= \mathop{\mathbb{E}}_{p(\boldsymbol{x})}[\![\tilde{\ell}(\boldsymbol{x}; \mathcal{N}_T, \mathcal{N}_S)]\!] + O(v^2).
\end{aligned} \tag{19}$$

Proof. Given a teacher network $\mathcal{N}_T$, student network $\mathcal{N}_S$ and their respective output $\boldsymbol{o}_T$ and $\boldsymbol{o}_S$, assuming a random perturbation $\gamma \sim (0, v\mathbf{I})$ is added to the training data, the loss function ℓ then becomes

$$\begin{aligned}
&\mathop{\mathbb{E}}_{p(\boldsymbol{x},\boldsymbol{\gamma})}[\![\ell(\boldsymbol{x}+\boldsymbol{\gamma}; \mathcal{N}_S, \mathcal{N}_T)]\!] \\
&= \mathop{\mathbb{E}}_{p(\boldsymbol{x},\gamma)}[\![(\mathcal{N}_S(\boldsymbol{x}+\gamma) - \boldsymbol{o}_T)^2]\!] \\
&= \mathop{\mathbb{E}}_{p(\boldsymbol{x},\gamma)}[\![\mathcal{N}_S^2(\boldsymbol{x}+\gamma) - 2\boldsymbol{o}_T\mathcal{N}_S^2(\boldsymbol{x}+\gamma) + \boldsymbol{o}_T^2]\!],
\end{aligned} \tag{20}$$

where $p(\boldsymbol{x}, \boldsymbol{\gamma})$ denotes the probability distribution of x and $\boldsymbol{\gamma}$. Assuming that the noise γ is small, $\mathcal{N}_S(\boldsymbol{x}+\gamma)$ can be approximated using Taylor series expansion of $\mathcal{N}_S(\boldsymbol{x}+\gamma)$ around $\mathcal{N}_S(\boldsymbol{x})$.

$$\begin{aligned}
\mathcal{N}_S(\boldsymbol{x}+\gamma) =& \mathcal{N}_S(\boldsymbol{x}) + \gamma^\top\nabla_{\boldsymbol{x}}\mathcal{N}_S(\boldsymbol{x}) \\
&+ \frac{1}{2}\gamma^\top\nabla_{\boldsymbol{x}}^2\mathcal{N}_S(\boldsymbol{x})\gamma + O(\gamma^3)
\end{aligned}$$

Substituting the approximation in Eq. (20), $\tilde{\ell}$ can then be approximated as the following

$$\begin{aligned}
&\mathop{\mathbb{E}}_{p(\boldsymbol{x},\boldsymbol{\gamma})}[\![\ell(\boldsymbol{x}+\boldsymbol{\gamma}; \mathcal{N}_S, \mathcal{N}_T)]\!] \\
&\approx \mathop{\mathbb{E}}_{p(\boldsymbol{x},\gamma)}[\![(\mathcal{N}_S(\boldsymbol{x}) + \gamma^\top\nabla_{\boldsymbol{x}}\mathcal{N}_S(\boldsymbol{x}) + \frac{1}{2}\gamma^\top\nabla_{\boldsymbol{x}}^2\mathcal{N}_S(\boldsymbol{x})\gamma)^2]\!] \\
&\quad - 2\mathop{\mathbb{E}}_{p(\boldsymbol{x},\gamma)}[\![\boldsymbol{o}_T\mathcal{N}_S(\boldsymbol{x})\boldsymbol{o}_T\gamma^\top\nabla_{\boldsymbol{x}}\mathcal{N}_S(\boldsymbol{x}) \\
&\qquad + \frac{1}{2}\boldsymbol{o}_T\gamma^\top\nabla_{\boldsymbol{x}}^2\mathcal{N}_S(\boldsymbol{x})\gamma]\!] \\
&= \mathop{\mathbb{E}}_{p(\boldsymbol{x})}[\![(\mathcal{N}_S(\boldsymbol{x}) - \boldsymbol{o}_T)^2]\!] - 2\mathop{\mathbb{E}}_{p(\boldsymbol{x},\gamma)}[\![\frac{1}{2}\boldsymbol{o}_T\gamma^\top\nabla_{\boldsymbol{x}}\mathcal{N}_S(\boldsymbol{x})\gamma]\!] \\
&\quad + \mathop{\mathbb{E}}_{p(\boldsymbol{x},\gamma)}[\![\mathcal{N}_S(\boldsymbol{x})\gamma^\top\nabla_{\boldsymbol{x}}^2\mathcal{N}_S(\boldsymbol{x})\gamma \\
&\qquad + (\gamma^\top\nabla_{\boldsymbol{x}}\mathcal{N}_S(\boldsymbol{x}))^2 + O(\gamma^3)]\!] \\
&= \ell(\boldsymbol{x}; \mathcal{N}_S, \mathcal{N}_T) + v\mathop{\mathbb{E}}_{p(\boldsymbol{x})}[\![(\mathcal{N}_S(\boldsymbol{x}) - \boldsymbol{o}_T)\nabla_{\boldsymbol{x}}^2\mathcal{N}_S(\boldsymbol{x})]\!] \\
&\quad + v\mathop{\mathbb{E}}_{p(\boldsymbol{x})}[\![\|\nabla_{\boldsymbol{x}}\mathcal{N}_S(\boldsymbol{x})\|_2^2]\!].
\end{aligned}$$

If this loss function is minimized by taking the functional gradient of $\mathcal{N}_S(\boldsymbol{x})$ and setting the result to zero, then

$$\mathcal{N}_S(\boldsymbol{x}) = \mathop{\mathbb{E}}_{p(\boldsymbol{o}_T,\boldsymbol{x})}[\![\boldsymbol{o}_T]\!] + O(v)$$

which indicates that

$$\mathop{\mathbb{E}}_{p(\boldsymbol{x})}[\![(\mathcal{N}_S(\boldsymbol{x}) - \boldsymbol{o}_T)\nabla_{\boldsymbol{x}}^2\mathcal{N}_S(\boldsymbol{x})]\!]$$

reduces to $O(v)$. Hence, Eq. (20) becomes

$$\mathop{\mathbb{E}}_{p(\boldsymbol{x},\boldsymbol{\gamma})}[\![\ell(\boldsymbol{x}+\boldsymbol{\gamma};\mathcal{N}_S,\mathcal{N}_T)]\!] = \mathop{\mathbb{E}}_{p(\boldsymbol{x})}[\![\ell(\boldsymbol{x};\mathcal{N}_S,\mathcal{N}_T)]\!] + O(v^2) + v \mathop{\mathbb{E}}_{p(\boldsymbol{x})}[\![\|\nabla_{\boldsymbol{x}}\mathcal{N}_S(\boldsymbol{x})\|_2^2]\!],$$

which we use $\tilde{\ell}$ to denote,

$$\begin{aligned}&\mathop{\mathbb{E}}_{p(\boldsymbol{x},\boldsymbol{\gamma})}[\![\ell(\boldsymbol{x}+\boldsymbol{\gamma};\mathcal{N}_S,\mathcal{N}_T)]\!] \\ &= \mathop{\mathbb{E}}_{p(\boldsymbol{x})}[\![\ell(\boldsymbol{x};\mathcal{N}_S,\mathcal{N}_T) + v\,\|\nabla_{\boldsymbol{x}}\mathcal{N}_S(\boldsymbol{x})\|_2^2]\!] + O(v^2) \\ &:= \mathop{\mathbb{E}}_{p(\boldsymbol{x})}[\![\tilde{\ell}(\boldsymbol{x};\mathcal{N}_T,\mathcal{N}_S)]\!] + O(v^2).\end{aligned}$$

□

This loss function is similar to Eq. (15) in many aspects. Firstly, their first term is identical. Secondly, the second term is the gradient with respect to the input. Optimizing Eq. (15) also optimizes Eq. (19). Since the first terms of Eq. (19) and $\tilde{\ell}$ are equal, the second terms can be seen as a regularization term that penalizes the large gradient value of $\mathcal{N}_S(\boldsymbol{x})$. The two regularization terms can also be treated as ℓ_1 and ℓ_1 regularizations with respect to the gradient. Similar to the feature of L1 regularization, the regularization term of Eq. (15) is robust to large gradients during training. Therefore, optimizing Eq. (15) with $\mathbb{P}_N$ resembles to training with $\mathbb{P}_N$ with random perturbation, which is a popular method to perform data augmentation. The performance of the student network trained with few data can then be guaranteed by emulating the data augmentation.

Experiment

Now we empirically evaluate the proposed algorithm on popular benchmark datasets, including CIFAR-10 dataset, CIFAR-100 dataset and Fashion-MNIST dataset.

Datasets and Network Configuration

CIFAR-10 and CIFAR-100 dataset CIFAR-10 and CIFAR-100 dataset (Krizhevsky 2009) consists of 60,000 tiny RGB images with shape 32×32, where 50,000 of the images are training set and the remaining 10,000 images are intended for testing. The tiny images in CIFAR-10 are split into 10 mutually exclusive categories, which are airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. For the two datasets, we use the same teacher and student network structures, while CIFAR-100 is split into 100 mutually exclusive categories. The architecture of the teacher network for the two datasets consists of three convolutional maxout (Goodfellow et al. 2013) layers followed by a fully connected maxout layer in a 96-192-192-500 plus a softmax layer configuration. The design of the teacher network generally follows the architecture used in FitNet (Romero et al. 2015) and maxout with some minor modifications (See Table 1). The images are rescaled to range $[0, 1]$ and are augmented by random cropping with paddings and random horizontal flipping, before feeding into the network. The mean and standard deviation of the images are rescaled to zero and one across three channels.

Fashion-MNIST dataset Fashion-MNIST dataset (Xiao, Rasul, and Vollgraf 2017) consists of 28×28 greyscale images from ten different categories of fashion items, including T-shirt, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag, and ankle boot. The number of examples for training and testing is 60,000 and 10,000, respectively. The architecture of the teacher network consists of two convolutional layers with kernel size 8×8 followed by a fully connected layer with 4096 units and a softmax layer that predicts the probability distribution over the ten categories. Before images are fed into the network, the images are rescaled to range $[0, 1]$ and are augmented by random horizontal flipping only. Fashion-MNIST is favored over MNIST for its complexity and its more accurate presentation of modern computer vision tasks.

Evaluation of Generalization Ability

To demonstrate the advantage of the proposed algorithm in generalization, we investigate the classification performance when the student network is trained with a different number of examples. The teacher network is pre-trained with complete training examples. As for the student network, we randomly sample M examples for each of categories and use them for training. Then different M values indicate the different size of the training set for the student network. During sampling, balance with respect to the number of examples across different categories is strictly maintained to avoid unnecessary performance impact.

We compare the classification accuracy of the student network when it is trained with different competing losses, including the original knowledge distillation (KD) loss, FitNet (FN) (Romero et al. 2015), Activation Boundary (AB) loss (Heo et al. 2019) as well as our WaGe loss. As mentioned in the previous section, following related works, the WaGe loss is designed to be used on the top of KD loss. For the fair comparison, all methods use the same teacher network, and all student networks have an identical structures. In specific, the pretrained teacher networks have testing accuracy of 89%, 47% and 92% on the CIFAR-10, CIFAR-100 and Fashion-MNIST dataset, respectively. In our experiments, ϵ is set to 1 and α is set to 0.001. Temperature T for KD loss is set to 3. On both datasets, the student networks are trained using back propagation and Stochastic Gradient Descent (SGD) with momentum for 500 epochs. During training, the learning rate and the momentum decay linearly. The experiments are run on a single NVIDIA GeForce 1080 Ti GPU.

Experimental Results

Table 2 describes the performance of each technique using the CIFAR-10 dataset. The simplest model, Student #0, is used for all experiments in this table. As can be shown in Table 2, from the results obtained from the CIFAR-10 dataset, due to the optimization with respect to generalization ability, models trained with WaGe loss outperform in comparison with other techniques. Specifically, when the models are trained with the full training set, the testing accuracy of the model trained with WaGe loss outperforms AB by around 2 percentage points and outperforms other training techniques

Table 1: Details of the network structure used in our experiment. "conv" means the convolutional layers and "FC" stands for the fully-connected layers.

CIFAR Teacher	CIFAR-10 Student #0	CIFAR-10 Student #1	CIFAR-10 Student #2	CIFAR-100˜ Student	Fashion-MNIST Teacher	Fashion-MNIST Student
conv 8x8x96 maxpool 2x2 dropout	conv 3x3x16 conv 3x3x32 conv 3x3x32 maxpool 2x2	conv 3x3x16 conv 3x3x16 conv 3x3x16 maxpool 2x2	conv 3x3x32 conv 3x3x48 conv 3x3x64 conv 3x3x64 maxpool 2x2	conv 3x3x32 conv 3x3x48 conv 3x3x64 conv 3x3x64 maxpool 2x2 dropout	conv 8x8x32 maxpool 2x2	conv 3x3x32 conv 3x3x32 conv 3x3x32 conv 3x3x32 maxpool 2x2
conv 8x8x192 maxpool 2x2 dropout	conv 3x3x48 conv 3x3x64 conv 3x3x80 maxpool 2x2	conv 3x3x32 conv 3x3x32 conv 3x3x32 maxpool 2x2	conv 3x3x80 conv 3x3x80 conv 3x3x80 conv 3x3x80 maxpool 2x2	conv 3x3x80 conv 3x3x80 conv 3x3x80 conv 3x3x80 maxpool 2x2 dropout	conv 8x8x64 maxpool 2x2	conv 3x3x64 conv 3x3x64 conv 3x3x64 conv 3x3x64 maxpool 2x2
conv 5x5x192 maxpool 2x2	conv 3x3x96 conv 3x3x96 conv 3x3x128 maxpool 2x2	conv 3x3x48 conv 3x3x48 conv 3x3x64 maxpool 8x8	conv 3x3x128 conv 3x3x128 conv 3x3x128 conv 3x3x128 maxpool 8x8	conv 3x3x80 conv 3x3x80 conv 3x3x80 conv 3x3x80 maxpool 2x2 dropout	FC 4096	FC 500
maxout 500 FC+softmax	maxout 500 FC+softmax	maxout 500 FC+softmax	maxout 500 FC+softmax	maxout 500 FC+softmax	FC+softmax	FC+softmax

Table 2: Classification accuracy with respect to various methods and different number of training examples on CIFAR-10 dataset.

M	Ours	AB	FN	KD
Full	**84.66%**	82.65%	77.84%	79.72%
1000	**82.01%**	76.19%	63.84%	77.63%
500	**73.08%**	66.69%	54.51%	50.63%
100	**47.24%**	44.7%	29.68%	31.83%
50	**41.16%**	38.23%	31.65%	28.32%

Table 3: Classification accuracy with respect to various methods and different number of training examples on Fashion-MNIST dataset.

M	Ours	AB	FN	KD
Full	**94.72%**	92.1%	92.23%	92.1%
1000	**91.95%**	91.67%	91.53%	90.44%
500	**90.46%**	90.25%	82.6%	89.45%
100	**85.18%**	84.09%	81.92%	83.77%
50	**81.73%**	80.82%	81.40%	78.5%

by a larger margin. When the number of training samples reduces, WaGe loss can still maintain its ability to generalize by outperforming all other baseline techniques. For example, WaGe outperforms the second-best baseline method by 4.25 points on KD, 6.39, 2.54 and 2.93 on AB when M equals to 1000, 500, 100 and 50, respectively.

Similar observations can be found from Table 3, which shows the testing accuracy of the models using the Fashion-MNIST dataset. When being trained on the full training set, the performance of all tested models is superior to the teacher networks, while WaGe loss achieves the highest test set accuracy of 94.72%. As M decreases, the accuracy of WaGe loss drops from 94.72% to 81.73%, while AB loss drops from 92.1% to 80.82%, FitNet drops from 92.23% to 81.4% and vanilla KD drops from 92.1% to 78.5%. The test set accuracy of the model trained with WaGe loss is higher than AB loss, FitNet and KD loss at all tested values of M.

Table 4: Classification accuracy with respect to various methods and different number of training examples on CIFAR-100 dataset.

M	Ours	AB	FN	KD
Full	**49.64%**	48.22%	48.94%	48.74%
250	34.98%	34.28%	**35.38%**	34.7%
100	**21.81%**	20.68%	20.37%	21.76%
50	**20.32%**	19.22%	18.64%	19.27%

Structural Complexity Analysis of Student Networks

To examine the generalization ability benefit of the WaGe loss from the perspective of network structure, we conducted experiments with the same setting as described in Section , but with different network complexities. In this experiment, we investigate the change of performance affected by altering the network complexity. Concretely, we trained Student #1 and #2 using the techniques mentioned in the previous section, and compare the difference in terms of testing accuracy. The number of parameters in Student #1 is smaller than that of Student #0 while the number of parameters in

Table 5: Classification accuracy of student networks on CIFAR-10 trained by various methods using different number of samples.

	Student #1				Student #2			
M	Ours	AB	FN	KD	Ours	AB	FN	KD
1000	78.37%	73.46%	64.83%	62.88%	84.23%	78.64%	62.84%	62.46%
500	67.56%	62.88%	52.71%	48.12%	75.45%	68.82%	49.12%	53.75%
100	43.96%	41.54%	30.2%	28.52%	49.69%	42.95%	42.30%	36.55%
50	41.41%	35.49%	29.0%	35.52%	37.67%	36.45%	29.89%	30.73%

Table 6: Structural complexity statistics of the teacher network and various student networks on CIFAR-10 dataset. The compression and acceleration rate is with respect to the teacher network.

	#Params	#FLOPs	Compression Ratio	Acceleration Ratio
Teacher	11M	59.62M	100%	100%
Student #0	1M	5.27M	8.87%	8.83%
Student #1	0.3M	1.73M	2.92%	2.90%
Student #2	2M	9.95M	16.72%	16.69%

Table 7: Classification accuracy of the student networks on CIFAR-10 trained by WaGe with respect to different hyper-parameters α and ϵ in log scale.

	$log_{10}\alpha$				
$log_{10}\epsilon$	-1	-2	-3	-4	-5
0	63.16%	80.33%	80.65%	77.67%	77.63%
-1	63.55%	80.03%	79.81%	78.14%	78.49%
-2	62.62%	80.50%	79.27%	78.95%	77.77%
-3	62.54%	80.12%	79.48%	78.81%	78.21%
-4	62.64%	80.07%	78.78%	78.58%	74.90%
-5	63.05%	79.92%	78.81%	78.52%	78.10%

Student #2 is larger. Table 6 shows the general information of the architectures and Table 1 contains the detail configuration.

Table 5 shows the testing accuracy achieved by various training techniques on Student #1 and Student #2. As can be seen from Table 5, the performance of WaGe loss remains on the top of other techniques on all listed M values, despite the complexity of the network structure is increased or decreased. On $M = 1000$, Student #2 trained by WaGe loss obtained 84.23%, despite the number of parameters being 16.72% of the teacher network. Meanwhile, the testing accuracy obtained by other techniques on Student #2 is lower than their corresponding results under the same M in Table 5 by a relatively large margin, which demonstrates the superiority of the proposed WaGe loss in terms of generalization.

Hyperparameter Analysis

In this subsection, we perform a hyperparameter analysis of our proposed WaGe loss. To this end, the CIFAR-10 model is trained using $M = 1000$. Other parameters, such as the temperature of the distillation, are heavily discussed in previous work (Hinton, Vinyals, and Dean 2015). Hence, we focus on two important hyperparameters that are included in the proposed WaGe loss, which are the weight of WaGe loss α and ϵ. Table 7 shows the result of the experiments for all combinations of α from 1 to 10^{-5} with increment 0.1, and ϵ from 1 to 10^{-7} with the same increment using 1000 examples. In theory, the optimal radius of the Wasserstein ball ϵ can be computed with given confidence value within the range $(0, 1)$ using Equation 8 in (Mohajerin Esfahani and Kuhn 2018, Theorem 3.4). However, in our experiments, with a fixed value of α, the performance of the models shows a low correlation with the generalization ability of the network. As α decreases, the performance of the network drops dramatically as the importance of WaGe loss diminishes. When α is larger than 0.001, the performance of the network drops dramatically, which is due to the inclusion of the gradient term. This experiment indicates that the effect of WaGe loss is only sensitive to α, which controls the relative weight of WaGe loss itself, in comparison with other losses (*i.e.*, KD loss).

Conclusion

We have proposed an approach to enhance the generalization ability of the student network when the training data is quite limited. Concretely, we assume that the ground-truth data-generating distribution actually lies in a Wasserstein ball centered on the training examples' discrete empirical distribution. Thus we can safely optimize the risks of all possible distributions within this ball to boost the generalization ability. Furthermore, for ease of training networks in end-to-end fashion, we theoretically relax the upper bound of the supremum risk, and develop a novel loss called WaGe loss accordingly. In particular, the proposed WaGe loss is also easy to implement for the networks in real applications. Extensive experimental results on benchmark datasets validate the effectiveness of our proposed method. It has been shown that our WaGe loss is capable of improving the classification performance of the student network, and has significant superiority over other competing methods when the training data is very limited.

Acknowledgement

This work was supported in part by the Australian Research Council under Project DE180101438.

References

Arjovsky, M.; Chintala, S.; and Bottou, L. 2017. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, 214–223.

Ba, J., and Caruana, R. 2014. Do deep nets really need to be deep? In *Advances in neural information processing systems*, 2654–2662.

Bertsekas, D. P. 2009. *Convex optimization theory*. Athena Scientific Belmont.

Chen, W.; Wilson, J.; Tyree, S.; Weinberger, K.; and Chen, Y. 2015. Compressing neural networks with the hashing trick. In *International Conference on Machine Learning*, 2285–2294.

Chollet, F. 2017. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1251–1258.

Denton, E. L.; Zaremba, W.; Bruna, J.; LeCun, Y.; and Fergus, R. 2014. Exploiting linear structure within convolutional networks for efficient evaluation. In *Advances in neural information processing systems*, 1269–1277.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Fournier, N., and Guillin, A. 2015. On the rate of convergence in wasserstein distance of the empirical measure. *Probability Theory and Related Fields* 162(3-4):707–738.

Gong, Y.; Liu, L.; Yang, M.; and Bourdev, L. 2014. Compressing deep convolutional networks using vector quantization. *arXiv preprint arXiv:1412.6115*.

Goodfellow, I. J.; Warde-Farley, D.; Mirza, M.; Courville, A.; and Bengio, Y. 2013. Maxout networks. *arXiv preprint arXiv:1302.4389*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hein, M., and Andriushchenko, M. 2017. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *Advances in Neural Information Processing Systems*, 2266–2276.

Heo, B.; Lee, M.; Yun, S.; and Choi, J. Y. 2019. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3779–3787.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Hunter, D. R., and Lange, K. 2004. A tutorial on mm algorithms. *The American Statistician* 58(1):30–37.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

Krizhevsky, A. 2009. Learning multiple layers of features from tiny images. Technical report, Citeseer.

Lopez-Paz, D.; Bottou, L.; Schölkopf, B.; and Vapnik, V. 2015. Unifying distillation and privileged information. *CoRR* abs/1511.03643.

Mohajerin Esfahani, P., and Kuhn, D. 2018. Data-driven distributionally robust optimization using the wasserstein metric: performance guarantees and tractable reformulations. *Mathematical Programming* 171(1):115–166.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language models are unsupervised multitask learners. *OpenAI Blog* 1(8).

Romero, A.; Ballas, N.; Kahou, S. E.; Chassang, A.; Gatta, C.; and Bengio, Y. 2015. Fitnets: Hints for thin deep nets. *CoRR* abs/1412.6550.

Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *CoRR* abs/1409.1556.

Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1–9.

Tang, Y.; You, S.; Xu, C.; Shi, B.; and Xu, C. 2019. Bringing Giant Neural Networks Down to Earth with Unlabeled Data. *arXiv:1907.06065 [cs, stat]*.

Wang, Z.; Schaul, T.; Hessel, M.; Van Hasselt, H.; Lanctot, M.; and De Freitas, N. 2015. Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581*.

Wang, C.; Li, M.; and Smola, A. J. 2019. Language models with transformers. *CoRR* abs/1904.09408.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; and He, K. 2017. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1492–1500.

You, S.; Xu, C.; Xu, C.; and Tao, D. 2017. Learning from Multiple Teacher Networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, 1285–1294. New York, NY, USA: ACM.