Robust Student Network Learning

Tianyu Guo, Chang Xu✉, Shiyi He, Boxin Shi, Member, IEEE, Chao Xu, and Dacheng Tao✉, Fellow, IEEE

Abstract—Deep neural networks bring in impressive accuracy in various applications, but the success often relies on heavy network architectures. Taking well-trained heavy networks as teachers, classical teacher–student learning paradigm aims to learn a student network that is lightweight yet accurate. In this way, a portable student network with significantly fewer parameters can achieve considerable accuracy, which is comparable to that of a teacher network. However, beyond accuracy, the robustness of the learned student network against perturbation is also essential for practical uses. Existing teacher-student learning frameworks mainly focus on accuracy and compression ratios, but ignore the robustness. In this paper, we make the student network produce more confident predictions with the help of the teacher network, and analyze the lower bound of the perturbation that will destroy the confidence of the student network. Two important objectives regarding prediction scores and gradients of examples are developed to maximize this lower bound, to enhance the robustness of the student network without sacrificing the performance. Experiments on benchmark data sets demonstrate the efficiency of the proposed approach to learning robust student networks that have satisfying accuracy and compact sizes.

Index Terms—Deep learning, knowledge distillation (KD), teacher–student learning.

I. INTRODUCTION

RECENT years have witnessed the marked progress of deep learning. Since the breakthrough in 2012 ImageNet competition [1] achieved by AlexNet [2] using five convolutional layers and three fully connected layers, a series of more advanced deep neural networks have been developed to keep rewriting the record, e.g., VGGNet [3], GoogLeNet [4], and ResNet [5]. However, their excellent performances require the support of a huge amount of computation. For instance, AlexNet [3] contains about 232 million parameters and needs $7.24 \times 10^8$ multiplications to process an image with resolution of $227 \times 227$. Hence, the potential power of deep neural networks can only be fully unlocked on high-performance GPU servers or clusters. In contrast, the majority of the mobile devices used in our daily life usually have rigorous constraints on the storage and computational resource, which prevents them from fully taking advantages of the deep neural network. As a result, networks with smaller hardware demanding while still maintaining similar accuracies are of great interests to machine learning and computer vision community.

Compressing convolutional neural networks can be achieved by vector quantization [6], decomposing weight matrices [7], and encoding with hashing tricks [8]. Unimportant weights can be pruned to achieve the same goal by removing the subtle weights [9], [10], reducing the redundancy between weights in the frequency domain [11], and using the binary networks [12], [13]. Another straightforward approach is to design a compact network directly, e.g., ResNeXt [14], Xception network [15], and MobileNets [16]. These networks are often deep and thin with fewer parameters in each layer, and the nonlinearity of these networks are strengthened by increasing the number of layers, which guarantees the performance of the network.

Teacher–student learning framework, introduced in knowledge distillation (KD) [17], is one of the most popular approaches to realize model compression and acceleration [11], [12]. Taking a heavy neural network, such as GoogleNet [4] or ResNet [5], that has already been well trained with massive data and computing resources as the teacher network, a student network of light architecture can be better learned under the teacher’s guidance. To inherit the advantages of the teacher network, different methods [18]–[21] have been proposed to encourage the consistency between teacher and student networks.

These aforementioned algorithms have achieved impressive experimental results. However, they were mainly developed in ideal scenarios, where all data are implicitly assumed to be clean. In practice, given examples with perturbation, the training process of the network can be seriously influenced, and the resulting network would not be confident as before to make predictions of examples. The teacher network might make some mistakes since it is difficult for the teacher network to be familiar with all examples fed into the student network. This is consistent with student–teacher learning in the real world. An excellent student is expected to solve practical problems in changeable circumstances, where there might be questions even not known by teachers.

To solve this problem, in this paper, we introduce a robust teacher–student learning algorithm. The framework of the proposed method is illustrated in Fig. 1. We enable the student
network to be more confident in its prediction with the help of a teacher network. Perturbations on examples might seriously influence the learning of the student network. We derive the lower bound of the perturbations that can make the student more vulnerable than a teacher through rigorous theoretical analysis. New objectives in terms of prediction scores and gradients of examples are further developed to maximize the lower bound of the required perturbation. Hence, the overall robustness of the student network to resist perturbations on examples can be improved. Experimental results on benchmark data sets demonstrate the superiority of the proposed method for learning compact and robust deep neural networks.

We organized the rest of the paper as follows. In Section II, we summarize related works on learning convolutional neural networks with fewer parameters by different methods. Section III introduces the previous work we based on. In Section IV, we formally introduce our robust student network learning method in detail, including mathematical proof to the proposed theorem, the calculation method of the loss function, and the training strategy. Section V provides the results of our algorithm obtained on various benchmark data sets to prove the effectiveness of the proposed method. Section VI concludes this paper.

II. RELATED WORKS

In this section, we briefly introduce related works on learning an efficient convolutional neural network with fewer parameters. There are two different categories of methods according to their techniques and motivations.

A. Network Trimming

Network trimming aims to remove redundancy in heavy networks to obtain a compact network with fewer parameters and less computational complexity, whereas the accuracy of this portable network is close to that of the original large model. Gong et al. [6] utilized the benefits of vector quantization to compress neural networks, and a cluster center of weights was introduced as the representation of similar weights. Denton et al. [7] implemented singular value decomposition to the weight matrix of a fully connected layer to reduce the number of parameters. Chen et al. [8] attempted to explore hash encoding to improve the compression ratio. Courbariaux et al. [12] and Rastegari et al. [13] implemented binary networks. All weights previously stored as 32-bit floating, are converted to binary \((-1, 0, 1)\) or \((-1, 1)\). Moreover, Wang et al. [11] and Han et al. [9] exploited weight pruning to achieve the same goal. In particular, Han et al. [9] focused on removing subtle weights to reduce the parameters while minimizing the impact of removing them. Over 80% of subtle weights were dropped without the accuracy drop. Furthermore, Han et al. [10] integrated several neural network compression techniques, i.e., pruning, quantization, and Huffman coding, to further compress the network. Wang et al. [11] showed that redundancy exists not only subtle weights but also large weights. It converted convolutional kernels into the frequency domain to reduce the redundancy contained in larger weights and thereby compress networks with a higher compression ratio. Also, Wang et al. [22] focused on the redundancy in feature maps instead of network weights, which can also be considered as a modification of network architecture. Although the network trimming method brings a considerable compression and speedup ratio, due to the highly sparse parameters and the irregular network architectures, the actual acceleration effect is often heavily dependent on the customized hardware.

B. Design Small Networks

Directly designing a new deep neural network of light size is a straightforward approach to realize efficient deep learning. Most of these methods increase the depth of networks with much lower complexity compared to simply stacking

C. Teacher–Student Learning

As one way of training a portable network, teacher–student learning aims to learn a student network with much fewer parameters compared with the teacher network. There are many heavy networks with a huge number of parameters. With the help of large amounts of data, these heavy networks are well trained and will achieve state-of-the-art performance on many challenging data sets. However, the high demands placed on the storage space and computing power caused by the huge number of parameters limit the application scenarios of these giant networks. This allows them to be deployed only on high-performance computers with GPUs. Learning a small network directly has proven to be hard to train and difficult to achieve satisfactory accuracy. Teacher–student learning introduces a method of learning a small network by building a bridge between the heavy network and the small network. By regarding the well-trained heavy network as a teacher and the portable network as a student, this method is successful in utilizing the intrinsic information captured by the teacher to conduct the learning progress of the student. There are lots of methods beneficial to learning deep networks and improving proposed to distillate knowledge from the teacher and learning a portable student network. Ba and Caruana [18] introduced a method that the student network mimics the features extracted from the last layer of the teacher network to assist the training progress of student networks, thereby increasing the depth of the student network. KD [17] pointed out that, for two networks with huge structural differences, it is difficult to directly mimic features. Therefore, KD [17] proposed to minimize the relaxed output of softmax layers of the two networks. This strategy can further deepen the student network. FitNet [33], based on KD, minimized the difference between the features extracted from the middle layers of the student and teacher networks. They added several layers of multilayer perceptrons at the middle layer of the teacher network to match the dimensions of the features of the student network. By establishing a connection between the middle layers of two networks, the student network can be further deepened with fewer parameters. McClure and Kriegeskorte [19] attempted to minimize the distance between pairs of samples to reduce the difficulty of training the student network. You et al. [20] proposed utilizing multiple teacher networks to provide more guidance for the training of student networks. They leverage a voting strategy to balance the multiple guidance from each teacher network. Wang et al. [21] regarded the student network as a generator which is a part of generative adversarial network [34], as well as utilized a discriminator as an assistant of teacher for forcing the student to generate features which are difficult to distinguish from features of the teacher.

Compared to the network trimming algorithm, the student–teacher learning framework has more flexibility, no special requirements on hardware, and a more structured network structure. Compared to the direct design of a deeper network, guidance from the teacher is beneficial to learning deep networks and improving the performance of the student. However, existing student–teacher algorithms pay more attention to improving the performance of the student network on pure data sets. The instability caused by the large reduction in parameters makes the performance degradation under the perturbation settings not yet studied. Therefore, a more robust learning algorithm for improving student network performance under perturbed conditions is desired. This paper proposed a method under the teacher–student learning and KD framework to improve the robustness of the student network.

III. Preliminary of Teacher–Student Learning

To make this paper self-contained, we briefly introduce some preliminary knowledge of teacher-student learning here. The teacher network \( \mathcal{N}_T \) has a complicated architecture, and it has already been well trained to achieve sufficiently high performance. We aim to learn a student network \( \mathcal{N}_S \), which is deeper yet thinner than the teacher network \( \mathcal{N}_T \) but still has a satisfying accuracy. Let \( \mathcal{X} \) be the example space and \( \mathcal{Y} \) be its corresponding \( k \)-label space. Outputs of these two networks are defined as

\[
o_T = \text{softmax}(o_T), \quad o_S = \text{softmax}(a_S)
\]

where \( o_T \) and \( a_S \) are the features produced by presoftmax layers of teacher and student networks, respectively.

The teacher network \( \mathcal{N}_T \) is usually trained on a relatively large data set and consists of a large number of parameters so that the teacher network usually achieves high accuracy in the classification task. Given significantly fewer parameters and numbers of multiplication operations, if adopting the same training strategies as the teacher network, the student network \( \mathcal{N}_S \) is difficult to achieve high performance. It is,
therefore, necessary to improve the student network performance by investigating the assistance of the teacher network. A straightforward method is to encourage the features of an input to be similar [18].

The objection function can be written as

\[
\mathcal{L}(N_S) = \frac{1}{n} \sum_{i=1}^{n} \left[ \mathcal{H}(o^t_S, y^i) + \frac{\lambda}{2} \|o^t_S - o^t_T\|^2 \right]
\]

(2)

where the second term helps the student network to extract knowledge from the teacher, \( \mathcal{H} \) refers to the cross-entropy loss, \( o^t_i \) indicates the output of the \( i \)th example in \( \mathcal{X} \) by the student network, \( y^i \) refers to the corresponding label, and \( \lambda \) is the coefficient to balance two terms in the function. The teacher and student networks can be significantly different in architecture, and thus, it is difficult to expect features extracted by these two networks for the same example to be the same. Hence, KD [17], as an effective alternative, was proposed to distill knowledge from classification results to minimize

\[
\mathcal{L}_{KD}(N_S) = \frac{1}{n} \sum_{i=1}^{n} \left[ \mathcal{H}(o^t_S, y^i) + \lambda \mathcal{H}(\tau(o^t_S), \tau(o^t_T)) \right]
\]

(3)

where the second term \( \mathcal{H}(\tau(o^t_S), \tau(o^t_T)) \) aims to enforce the student network to learn from the softened output of the teacher network. \( \tau(\cdot) \) is a relaxation function defined as follows:

\[
\begin{align*}
\tau(o_T) &= \text{softmax} \left( \frac{o_T}{\tau} \right) \\
\tau(o_S) &= \text{softmax} \left( \frac{o_S}{\tau} \right)
\end{align*}
\]

(4)

\( \tau \) is introduced to make sure that the second term in equation (3) can play a different role compared with the first one. This is because \( o_T \) might be extremely similar to the one-hot code representation of the ground truth labels, while a softened version of output is different from the true labels. Moreover, the softened version of output could also provide more information to guide the learning of the student, as the cross-entropy loss and softened version output will enhance the influence of classes other than the true label one.

Although KD loss in (3) allows the student network to access the knowledge from the teacher network, the significant reduction in the number of parameters decreases the capability of the student network and makes it more vulnerable to input disturbances. The learned student network might achieve reasonable performance on clean data, but it would suffer from a serious performance drop when encountering perturbation on the data in real-world applications. To solve this issue, it is, therefore, necessary to enforce the robustness of the student network when applied to a practical scenario.

IV. ROBUST STUDENT NETWORK LEARNING

We take a multi-class classification problem over \( k \) classes as an example to introduce our robust student network learning. Given a teacher network \( N_T \) and a student network \( N_S \), an example \( x \) can then be classified by two networks \( o^t_T(x) = N_T(x) \) and \( o^t_S(x) = N_S(x) \), respectively. Denote \( o^t_{j_T}(x) \) and \( o^t_{j_S}(x) \) as the \( j \)-th value of the \( k \)-dimensional vectors \( o^t_T(x) \) and \( o^t_S(x) \), respectively. Then, we define \( f_T(x) = o^t_T(x) \) and \( f_S(x) = o^t_S(x) \) as the scores produced by two networks for the ground truth label \( y \) of example \( x \), respectively. If a classifier has more confidence in its prediction, the predicted score will be higher. With the help of the teacher network, the student network is supposed to be more confident in its prediction, so that

\[
f_S(x) > f_T(x).
\]

(5)

Confidence, here, refers to the predicted score of the neural network for an input image. If the network classifies the image to a category with a higher prediction score, we suggest the network is confident in its prediction.

A. Theoretical Analysis

The above relationship holds in an ideally noise-free scenario. In the practical scenario, perturbations on examples are unavoidable, and the student network is expected to resist the unexpected influence and bring in robust prediction

\[
f_S(x + \delta) > f_T(x + \delta), \quad \|\delta\|_2 \leq R \quad \text{and} \quad x + \delta \in C
\]

(6)

where \( \delta \) is a perturbation added to \( x \). The perturbation can come from various sources. There could be electrical noise, photon noise, thermal noise, and so on. As noise degrades the quality of an image, the performance of neural networks in image classification task could be seriously influenced. The proposed robust student network aims to handle unexpected noises in images and to preserve consistent decisions with or without noises. To achieve this, we restrict this perturbation in a spherical space of radius \( R \), and \( C \) is a constraint set that specifies some requirements for the input, e.g., an image input should be in \([0, 1]^d\), where \( d \) is the dimension. We define the ball as \( B_p(x, R) = \{ z \in \mathbb{R}^d \mid \|x - z\|_p \leq R \} \).

We aim to discover a student network that stands on the shoulder of the teachers to make a confident prediction not only for clean examples but also for examples with perturbations. The perturbation \( \delta \) exists on examples without influencing their corresponding ground truth labels. However, with the increase of perturbation intensity, the learning process of the student network would be seriously disturbed. Taking (6) as an auxiliary constraint in training the student network can be helpful for improving the robustness of the network. However, it is difficult and impossible to enumerate and try every possible \( \delta \) to form the constraint. To make the optimization problem tractable, we seek for some alternatives and proceed to study the maximum perturbation that can be defended by the system. Fig. 1 shows the framework of our approach.

**Theorem 1:** Let \( x \in \mathbb{R}^d \) be an example in \( \mathcal{X} \). \( f_S(x) \) and \( f_T(x) \) are functions adapted from the student and teacher networks to predict the label \( y \) of example \( x \), respectively. Given \( f_S(x) > f_T(x) \), for any \( \delta \in \mathbb{R}^d \) with

\[
\|\delta\|_p \geq \frac{f_S(x) - f_T(x)}{\max_{z \in B_p(x, R)} \|\nabla f_T(z) - \nabla f_S(z)\|_p}
\]

we have \( f_T(x + \delta) > f_S(x + \delta) \).

**Proof:** By the main theorem of calculus, we have

\[
f_S(x + \delta) = f_S(x) + \int_0^1 \langle \nabla f_S(x + t\delta), \delta \rangle dt
\]

(8)
and

$$f_T(x + \delta) = f_T(x) + \int_0^1 (\nabla f_T(x + t\delta), \delta)dt.$$  \hspace{1cm} (9)

If the perturbation $\delta$ is so significant that $f_T(x + \delta) > f_S(x + \delta)$, we get

$$0 < f_S(x) - f_T(x) \leq \int_0^1 (\nabla f_T(x + t\delta) - \nabla f_S(x + t\delta), \delta)dt.$$  \hspace{1cm} (10)

Consider the fact that

$$\int_0^1 (\nabla f_T(x + t\delta) - \nabla f_S(x + t\delta), \delta)dt \leq \|\delta\|_q \int_0^1 \|\nabla f_T(x + t\delta) - \nabla f_S(x + t\delta)\|_p dt$$  \hspace{1cm} (11)

where holder inequality is applied and $q$-norm is dual to the $p$-norm with $\frac{1}{p} + \frac{1}{q} = 1$. By combining (10) and (11), we have

$$f_S(x) - f_T(x) \leq \|\delta\|_q \int_0^1 \|\nabla f_T(x + t\delta) - \nabla f_S(x + t\delta)\|_p dt.$$  \hspace{1cm} (12)

Then, we can upper bound the left term of above equation as following:

$$f_S(x) - f_T(x) \leq \|\delta\|_q \int_0^1 \|\nabla f_T(x + t\delta) - \nabla f_S(x + t\delta)\|_p dt.$$  \hspace{1cm} (13)

Furthermore, we can obtain that

$$\|\delta\|_q \geq \frac{f_S(x) - f_T(x)}{\int_0^1 \|\nabla f_T(x + t\delta) - \nabla f_S(x + t\delta)\|_p dt}.$$  \hspace{1cm} (14)

where the denominator can be further upper bounded using the following inequality:

$$\int_0^1 \|\nabla f_T(x + t\delta) - \nabla f_S(x + t\delta)\|_p dt \leq \max_{\delta \in B_p(x,R)} \|\nabla f_T(z) - \nabla f_S(z)\|_p.$$  \hspace{1cm} (15)

As the 2-norm of $\delta$ is upper bounded by $R$ which is the radius of the Ball, and the value of $x + t\delta$ can get hardly out of this ball. The right inner term represents the max value in this ball. As a result, considering the left term of (15), the inner term of it is definitely less or equal than the inner term of the right term of equation (15). Integrated by $t$ will result in the right term. After that, we can finally conclude equation (15). Moreover, the lower bound for the $q$-norm of $\delta$ to break the robust prediction of the student network (i.e., equation (6)) is therefore

$$\|\delta\|_q \geq \frac{f_S(x) - f_T(x)}{\max_{\delta \in B_p(x,R)} \|\nabla f_T(z) - \nabla f_S(z)\|_p}.$$  \hspace{1cm} (16)

which completes the proof.

According to Theorem 1, maximizing the value of $f_S(x) - f_T(x)$ while minimizing the value of $\max_{\delta \in B_p(x,R)} \|\nabla f_T(z) - \nabla f_S(z)\|_p$, the lower bound over $\delta$ will be enlarged, so that the student network is able to tolerate more severe perturbation and become more robust to make confident prediction. $L_2$-norm is widely used in the optimization field because it is easier to calculate than other norms (such as $L_0$, $L_1$, and $L_p$), and more importantly, it is easier to be optimized by gradient descent in deep learning. In addition, in the field of the image process, some concepts are often defined in the Euclidean space, or the form of a power of 2, such as mean-square error and peak signal-to-noise ratio. Considering these above facts, we set $p = q = 2$.

### B. Method

Based on the analysis above, two new objectives are introduced into the teacher-student learning paradigm to achieve a robust student network. To encourage $f_S(x) > f_T(x)$, we plan to minimize the loss function

$$\mathcal{L}_S(N_S) = \frac{1}{n} \sum_{i=1}^n \max(0, \gamma + f_T(x^i) - f_S(x^i))$$  \hspace{1cm} (17)

where $\gamma > 0$ is a constant margin. $f_S(x)$ is supposed to be greater than $f_T(x) + \gamma$; otherwise, there will be a penalty for the student network. It is difficult to explicitly calculate the value of $\max_{\delta \in B_p(x,R)} \|\nabla f_T(z) - \nabla f_S(z)\|_2$ due to the existence of the max operation. However, by appropriately setting the radius $R$ and considering the sufficiently large training set, the data point in the ball $B_2(x,R)$ to reach the maximum value of $\|\nabla f_T - \nabla f_S\|_2$ would often have some closed examples in the training set. Hence, to minimize the value of $\max_{\delta \in B_p(x,R)} \|\nabla f_T(z) - \nabla f_S(z)\|_2$, we propose to minimize the difference between gradients of the student and the teacher networks with respect to the training examples as

$$\mathcal{L}_G(N_S) = \frac{1}{n} \sum_{i=1}^n \|\nabla r(f_S(x^i)) - \nabla r(f_T(x^i))\|^2$$  \hspace{1cm} (18)

where $r$ is the relaxation function explained in (3). In addition, we take the KD loss [17] into consideration, the resulting objective function of our robust student network learning algorithm can be written as

$$\mathcal{L}(N_S) = \mathcal{L}_{KD}(N_S) + C_1 \mathcal{L}_G(N_S) + C_2 \mathcal{L}_S(N_S)$$  \hspace{1cm} (19)

where $C_1$ and $C_2$ are the balanced coefficients of $\mathcal{L}_G(N_S)$ and $\mathcal{L}_S(N_S)$, respectively.

The process of training the student network can be found in Algorithm 1. After initialization of the student network, we train the student network according to the proposed algorithm. Next, we explain in detail the calculation of loss. For convenience, we set the batch size as 1, i.e., we first select a sample $\{x, y\}$ from the data set $\mathcal{X}$ and $\mathcal{Y}$ as input for forwarding propagation of the teacher network and the student network. Then, we calculate outputs of the two networks $o_T(x)$ and $o_S(x)$. Combining outputs $o_T(x)$ and $o_S(x)$ with the corresponding label $y$, we can calculate the first term in (19) $\mathcal{L}_{KD}$ according to (3) and (4). $o_T$ and $o_S$ are both $k$-dimensional vectors, which is the prediction score of the
Algorithm 1 Robust Student Network Learning

Input: A given neural network $N_T$; training data set $\mathcal{X}$ with $n$ instances; the corresponding $k$-label set $\mathcal{Y}$; parameters: $\lambda$, $\beta$, and $\tau$.

1. Initialize a neural network $N_S$, where the number of parameters in $N_S$ is significantly fewer than that in $N_T$;
2. repeat
3. Select an instance $x$ and its label $y$ randomly;
4. Employ the teacher network: $f_T \leftarrow N_T(x)$;
5. Employ the student network: $f_S \leftarrow N_S(x)$;
6. Calculate the loss function $L(N_S)$ w.r.t. equation (18);
7. Update weights in the student network $N_S$;
8. until reaching the limitation of training epoch;

Output: A robust student network $N_S$.

network for $k$ categories. With the help of label $y$, we can get the predicted scores for the label, $f_T$ and $f_S$ and calculate the second term in (19) $L_2$ according to (17). To get the value of $L_p$, we first calculate the derivative of $f_T$ and $f_S$ with respect to the input sample $x$. Automatic derivation tool integrated into PyTorch is applied here. Same as backpropagation algorithm, we can apply the chain rule to get these results. By doing so, we have obtained the gradients of $f_T$ and $f_S$ with respect to the input $x$, and now the loss mentioned in (17) can be easily calculated. After getting $f_T$ and $f_S$, we backpropagate the obtained loss through the network, which equals a second gradient, and we also leverage the automatic derivation tool of PyTorch to achieve this. Finally, the parameters in the student network are updated with the gradients obtained by the backpropagation algorithm.

In the literature, the overall noise produced by a digital sensor is usually considered as a stationary white additive Gaussian noise [35]. We report robustness of the learned networks against Gaussian noise in experiments. In addition, we also evaluate the performance of the learned networks against combinations of different types of noise on training and test sets, since it is difficult to know what types of noise could be before the test stage.

V. EXPERIMENTS

In this section, we experimentally investigate the effectiveness of the proposed robust student network learning algorithm. The learned student network is compared with the original teacher network, and student networks are learned through KD [17] and FitNet [33]. The experiments are conducted on three benchmark data sets: MNIST [36], CIFAR-10 [37], and CIFAR-100 [37].

A. Data Sets and Settings

MNIST [36] is a handwritten digit data set (from 0 to 9) composed of $28 \times 28$ greyscale images from ten categories. The whole data set of 70,000 images is split into 60,000 and 10,000 images for training and test, respectively. Following the setting in [33], we trained a teacher network of maxout convolutional layers reported in [38], which contains three convolutional maxout layers and a fully connected layer with 48-48-24-10 units, respectively. Later, we design the student network of six convolutional maxout layers and a fully connected layer, which is twice as deep as the teacher network. However, the parameters of the student network are only roughly 8% of the parameters of the teacher network. As reported in Table I, the architectures of the teacher and student networks were shown in detail in the first two columns.

CIFAR-10 [37] is a data set that consists of $32 \times 32$ RGB color images draw from ten categories. There are 60,000 images in the CIFAR-10 data set which are split into 50,000 training and 10,000 testing images. According to [38] and [33], we preprocessed the data using global contrast normalization (GCN) and zero-phase component analysis whitening and augmented the training data via random flipping. We followed the architecture used in Maxout [38] and FitNet [33] to train a teacher network with three maxout convolutional layers of 96-192-192 units. For a fair comparison, we designed a student network with a structure similar to FitNet which has 17 maxout convolutional layers followed by a maxout fully connected layer and a top softmax layer, and we also investigate KD method with the same architecture. The detailed architecture of the teacher was shown in the “Teacher (CIFAR-10)” column of Table I, and that of the student was shown as “Student 4” column.

CIFAR-100 data set [37] has images of the same size and format as those in CIFAR-10, except that it has 100 categories with only one-tenth as labeled images per category. More categories and fewer labeled examples per category indicate that classification task on CIFAR-100 is more challenging than that on CIFAR-10. We preprocessed images in CIFAR-100 using the same methods for CIFAR-10, and the teacher network and the student network share the same hyperparameters with those on the CIFAR-10 data set. In addition, the architecture of the teacher is also the same as that used for CIFAR-10, except that the number of units in the last softmax layer was changed to 100 to adapt to the number of categories.

The hyperparameters are tuned by minimizing the error on a validation set consisting of the last 10,000 training examples on each data set. Following the setting in FitNet [33], we set a batch size as 128, max training epoch as 500, learning rate as 0.17 for linear layers and 0.0085 for convolutional layers, and momentum as 0.35. According to the hint layer proposed in FitNet [33], we pretrained a classifier using the features in the middle layer of the teacher network, and then we apply the classifier with the student network features.

B. Robustness of Student Networks

We evaluated the robustness of the student network learned through different algorithms under different intensities of perturbation. Since it is difficult and sometimes impossible to know what test data can be in practice, the augmentation of training data with certain noise cannot be very helpful to resist the perturbation. Hence, we trained all networks using clean training set and introduced white Gaussian Noise (WGN) into test data as the perturbation. The intensity of the introduced noise was measured in terms of signal-to-noise ratio (SNR). We trained the proposed algorithm and compared it with


Table I

**Model Architecture for Data Sets**

<table>
<thead>
<tr>
<th>Teacher (MNIST)</th>
<th>Student (MNIST)</th>
<th>Teacher (CIFAR)</th>
<th>Student 1</th>
<th>Student 2</th>
<th>Student 3</th>
<th>Student 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>conv 8x8x48 pool 4x4</td>
<td>conv 3x3x16 pool 4x4</td>
<td>conv 8x8x96 pool 4x4</td>
<td>conv 3x3x16 pool 2x2</td>
<td>conv 3x3x16 pool 2x2</td>
<td>conv 3x3x32 pool 2x2</td>
<td>conv 3x3x32 pool 2x2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>conv 3x3x16 pool 4x4</td>
<td>conv 3x3x16 pool 4x4</td>
<td>conv 3x3x64 pool 4x4</td>
<td>conv 3x3x64 pool 4x4</td>
</tr>
<tr>
<td>conv 8x8x48 pool 4x4</td>
<td>conv 3x3x12 pool 4x4</td>
<td>conv 5x5x96 pool 4x4</td>
<td>conv 3x3x96 pool 8x8</td>
<td>conv 3x3x96 pool 8x8</td>
<td>conv 3x3x128 pool 8x8</td>
<td>conv 3x3x128 pool 8x8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>conv 3x3x96 pool 8x8</td>
<td>conv 3x3x96 pool 8x8</td>
<td>conv 3x3x128 pool 8x8</td>
<td>conv 3x3x128 pool 8x8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>conv 3x3x128 pool 8x8</td>
<td>conv 3x3x128 pool 8x8</td>
<td>conv 3x3x128 pool 8x8</td>
<td>conv 3x3x128 pool 8x8</td>
</tr>
</tbody>
</table>

The teacher network [38], and the student network from KD network [17] and FitNet [33] methods.

In Fig. 2, we investigated the accuracy of these networks on three data sets with different SNR values. As the classification task on MNIST is relatively easier, lower SNRs were chosen from 9 to 1. Lower SNR value indicates that more perturbations are added. It can be found from Fig. 2(a) that the accuracy of the proposed robust student network is superior to the other three networks nearly under all SNR values. When SNR equals 2, two student networks from KD and FitNet perform even worse than the original teacher network. How our proposed algorithm achieves obviously leading 98.17% accuracy. When SNR was down to 1, the accuracy drops of the teacher network and the student network from KD and FitNet are serious, up to 5.65%, 7.25%, and 3.23%, respectively. In contrast, the accuracy of our robust student network only drops 2.23%. Our method achieves better performance and shows more robustness when there was perturbation in the input.

A similar phenomenon can be observed in Fig. 2(b) and (c) on the CIFAR-10 and CIFAR-100 data sets. With the decrease of SNR, the accuracy of KD network and FitNet dropped faster than that of the teacher network, especially during the period when SNR drops from 12 to 10. Given the significant reduction in network complexity, the capacity of the student network can be seriously weakened, and the student network would be more vulnerable to perturbations on data if there is no appropriate response action. However, the student network learned from the proposed algorithm can be robust to serious perturbations.

In Fig. 3, we reported the predicted scores of example images by different methods on the CIFAR-10 data set. The clean image without noise looks fuzzy, since the images from the CIFAR-10 data set only a resolution of 32 x 32. As shown in Fig. 3(c), all student networks can confidently predict the ground truth class “cat” of the image. However, given the same image added with SNR=10 noise in Fig. 3(b) and (d) shows that though student networks from KD and FitNet methods...
reluctantly made the correct prediction, KD also thought the image is similar to “deer,” and FitNet trusted “bird” as the prediction with a higher confidence level. On the contrary, our robust student network confidently insists on its correct prediction even the quality of the image has been seriously influenced by the perturbation. In addition, given the “ship” image, the teacher network can stand against the perturbation, due to its strong capability coming from the complicated network structure. The KD method mistakes it as a “deer” image, while FitNet assigned a higher score to label “deer” for this “ship” image. By encouraging more confident predictions with help of the teacher network during the training stage, we derive the robust student network that can not only keep the highest prediction score on the “ship” label but also suppress the predictions on wrong categories [see label “cat” in Fig. 3(g) and (h)].

We conduct experiments on a much challengeable image classification data set ImageNet ILSVRC 2012 [1]. There are more than 1.28 M images for training and 50 K images for validation. In this part, we choose ResNet-101 [39], which contains about 128 M parameters as the teacher network and Inception-BN [40] which contain 32 M parameters as the student network. Referring to [41], we set the initial learning rate as 0.1, and it was divided by 10 at 30, 60, and 90 epochs, respectively. We optimize the network using Nesterov accelerated gradient with weight decay as $10^{-4}$ and momentum as 0.9. We train all models with clean data. In the test phase, we follow the method used in Section V-B to introduce noise into test data and validate these models with 1-crop. We report the Top-1 results in Table II. The teacher network achieves 28.77% Top-1 error. The proposed method achieves 30.15% Top-1 error, which outperforms student networks trained by comparison algorithms. In this data set, all student networks failed to outperform the teacher but still showed acceptable results which are not far away from the teacher.

### C. Comparison Under Different Perturbation

A neural network might handle noisy test data if similar noise also exists in the training set. However, in practice, it is difficult to guarantee the test data to have the same type of perturbation as the training data. We next proceed to evaluate the performance of different methods under different combinations of noisy training and test sets. The accuracies...
in different settings are presented in Table III. The first line in Table III is a description of the experiment settings. The first capital letter indicates the noise type introduced to the training set, and the second letter indicates that of the test set. It should be noted that the results of the Robust network listed in this table are all trained under the clean data set, but tested under the corresponding type of noise indicated by the first line. If both training and test data are clean, all networks can achieve more than 90% accuracy, as shown in the first column of Table III. If networks are trained on the clean data and tested on the data with Gaussian noise, the teacher and student networks from KD and FitNet will be seriously influenced and can only achieve less than 86% accuracy. However, the proposed robust student can still own more than 90% accuracy. A similar phenomenon can be observed when the networks are trained with Gaussian noise but tested with Poisson noise and the other way round, as shown in the fourth and last columns of Table III, respectively. If both training and test data are polluted with the same type of Gaussian noise, all networks would try to fit the noisy data as far as possible and receive only a slight performance drop. However, this rigorous constraint overtraining and test data cannot always be satisfied in real-world applications. It shows that adding limited kinds of noise to the training set is difficult to improve the robustness of neural network when facing various unexpected types of noises existing in practical applications.

Moreover, Table III shows that the accuracy of the teacher network is worse than that of “Robust” Network when training and test sets are both with Gaussian noise. The distributions of training data and test data would not be significantly different if they are both polluted by the Gaussian noise. Hence, the general teacher network and student network can well fit the noisy data and receive reasonable accuracy. However, the student network achieves some performance improvement, because of its deeper architecture than that of the teacher network. The depth encourages the reuse of features and leads to more abstract and invariant representations at higher layers. The proposed robust student network can successfully train a deeper network by exploiting information from the teacher network.

D. Complex Perturbation

In real-world applications, noise is not the only perturbation that may be encountered. Some more complex perturbation also challenges the robustness of neural networks. In this section, we investigate the robustness of the student network obtained by our proposed method on two more complex perturbations, i.e., image occlusion and domain adoption.

1) Image Occlusion: Considering the target object in the real environment is often blocked, and such perturbation often results in the loss of information in a continuous area, the performance of neural networks will be influenced more seriously by image occlusion. In order to investigate the robustness of our method under this disturbance, we took image occlusion as a more complex perturbation. To simulate the occlusion in the real-world applications, we randomly selected a small rectangular area in an image and set pixels covered by the rectangle as zeros. Five different block sizes, i.e., $2 \times 2$, $4 \times 4$, $6 \times 6$, $8 \times 8$, and $10 \times 10$, are used in experiments. We implemented this experiment on the CIFAR-10 data set and the CIFAR-100 data set. The results are shown in Table IV. Given $4 \times 4$ blocks, teacher, KD, FitNet, and the proposed network, respectively, have accuracies of 89.03%, 89.53%, 89.94%, and 90.79%. Given $8 \times 8$ blocks, the corresponding accuracies are 85.65%,
Combining with these visual recovery algorithms, classification networks enjoy higher performance as the polluted input images are cleaned. The proposed method does not include recovery images directly, but show robustness for polluted images in the classification task. As the analysis in this paper, the proposed method trains the student network within a Ball not only at data points, which provides the student network the ability to classify the polluted image with high accuracy.

E. Comparison With State-of-the-Art Methods

Although the main purpose of this paper is to improve the robustness of the student network, instead of focusing on the performance of the student network on clean data, we also compared the proposed approach with a state-of-the-art teacher–student learning methods on clean data sets. For the clean data, the proposed algorithm can still achieve comparable accuracy as compared to others in Tables VI–VIII. Table VII summarizes the obtained results on three data sets: MNIST, CIFAR-10, and CIFAR-100. On the MNIST data set, the teacher network got 99.45% accuracy. With the assistance of KD, the student network achieved 99.46% accuracy. FitNet generated a slightly better student network with 99.49% accuracy, which has outperformed the teacher network. Although the proposed algorithm aims to enhance the robustness of the learned student network, it can also achieve comparable or even better accuracy than those of state-of-the-art methods. The accuracy obtained by the proposed method increased to 99.51% on the MNIST data set. Table VI shows results on the CIFAR-10 data sets, and the baseline teacher network achieved 90.25% accuracy. Without help from the teacher network, the student network finally achieved 86.78% accuracy. Accuracy of the student network generated by KD and FitNet was 91.07% and 91.64%, respectively. The proposed student network obtained 91.93% accuracy as shown in Table VI, which outperforms other student networks and teacher. This suggests that the proposed method is able to enhance the stability of the student network and then improve the performance of the network.

CIFAR-100 is similar but more challenging than CIFAR-10 because of its 100 categories. The accuracy obtained by the teacher network is only 63.49%. As a comparison, the accuracy of the teacher on CIFAR-10 is 90.25%, which is much better than that on CIFAR-100. The robust student network achieved 65.28% accuracy, which outperforms student networks trained by other strategies, i.e., the network trained by KD obtains a test accuracy of 64.13%, and the accuracy of FitNet is 64.86%. When compared to other methods, the student network generated by the proposed method provides nearly state-of-the-art performance. This result demonstrates that the proposed method succeeds in assisting to learn a student network with considerable performance.

F. Analysis on Structures of Student Network

We followed the experimental setting in FitNet [33] and designed four student networks with different configurations.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>MNIST2USPS</th>
<th>MNIST</th>
<th>USPS</th>
<th>USPS2MNIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teacher [38]</td>
<td>99.45%</td>
<td>96.41%</td>
<td>86.88%</td>
<td></td>
</tr>
<tr>
<td>KD [17]</td>
<td>99.35%</td>
<td>93.25%</td>
<td>96.26%</td>
<td>82.74%</td>
</tr>
<tr>
<td>FitNet [33]</td>
<td>99.49%</td>
<td>94.12%</td>
<td>96.56%</td>
<td>87.23%</td>
</tr>
<tr>
<td>Robust (proposed)</td>
<td>99.55%</td>
<td>95.02%</td>
<td>96.71%</td>
<td>89.14%</td>
</tr>
</tbody>
</table>

83.37%, 84.62%, and 85.92%, respectively. According to these results, larger blocks indicate more serious perturbations of images, which will degrade the performance of neural networks. However, the student network obtained by the proposed method stably stays ahead, because of its robustness.

2) Domain Adaptation: In practical applications, not only unexpected noise and occlusion but also the unexpected distribution shift could challenge the robustness of neural networks. It is also an important indicator to evaluate the adaptability of this algorithm in the task of domain adaptation.

In this experiment, we took the USPS data set obtained from the scanning of handwritten digits from envelopes by the U.S. Postal Service. The images in this data set are all $16 \times 16$ grayscale images, and the values have been normalized. The whole data set has 9298 handwritten numeric images, of which 7291 are for training, and the remaining 2007 are for validation. Similar to MNIST, the USPS data set has ten categories, but it has different numbers of samples per category. In addition, considering the picture size in the MNIST data set is 28 \times 28, for convenience, we pad the images in the USPS data set to the same size. Moreover, we preprocessed the USPS data sets in the same way as MNIST.

In this section, we train student networks on the MNIST data set and test them on the USPS data set. Similarly, we train networks on the USPS data set and test them on MNIST. The results are shown in Table V. The first two columns show the result of adapting MNIST to USPS, and the performance of adapting USPS to MNIST was reported in the last two columns of this table. According to the results, the proposed algorithm achieves an accuracy of 95.02%, while the comparison methods KD and FitNet only get 93.25% and 94.12%, respectively. This demonstrates that the proposed robust student network can preserve its robustness advantage over comparison methods when faced with more complex perturbation of data in domain adaptation task. The similar phenomenon can be observed in the results of “USPS to MNIST.” With the similar accuracy on USPS data set, the Robust Network outperforms networks obtained by the other algorithms. Moreover, the results tested on USPS data set while trained on MNIST data set are much better than those tested on MNIST and trained on USPS. This is because the number of pictures in the MNIST data set is much larger than that of USPS. The networks trained by MNIST data set could extract more useful information from a larger amount of data and thus has better generalization capabilities.

3) Difference With Image Recovery: The mainstream visual recovery algorithms focus on recover images which are polluted, such as image denoising [44], [45], image deblur [46], and image inpainting [47], [48]. These methods aim to restore images from polluted images with the help of prior knowledge. Combining with these visual recovery algorithms, classification networks enjoy higher performance as the polluted input images are cleaned.
of parameters and layers. The teacher network has the same structure as that used on the CIFAR-10 data set. We design four student networks of different sizes and structures, and the detailed structure of these networks can be found in Table I. From “Student 1” to “Student 4,” the volume of the network has gradually increased, and the performance of the network has gradually increased as well. Table IX reported the performance of four student networks and the teacher network on the CIFAR-10 data set. The compression ratio and speed-up ratio compared with the teacher, and the number of parameters and multiplications can also be found in Table IX.

From Table IX, we find that the proposed robust student network outperforms FitNet under all four different student structures. Although there is no perturbation on the data, the proposed method can achieve higher accuracy, which indicates the effectiveness of encouraging the student network to make confident predictions with the help of the teacher network. In addition, the smallest network Student 1 has the biggest compression and speed-up ratios, but it can still achieve a test accuracy of 89.62%, which is fairly close to the 90.25% of teacher and outperforms the 89.07% obtained by FitNet. Student 1 contains significantly fewer parameters than those of the teacher, improving the accuracy of such a network with limited capacity is challenging, which in turn reflects the effectiveness of the proposed method.

Although there are significantly fewer parameters contained in the student network learned by the proposed method, these student networks are still regular networks which can be further compressed and accelerated by existing sparsity-based deep neural network compression technologies, such as deep compression [10] and feature compression [11].

**G. Hyperparameters and Running Time Analysis**

We choose hyperparameters following the principle of cross-validation. First, a validation set is sampled from the training set. Then, when the training progress is converged, the student network is evaluated on this validation set. Hyperparameters which provide the best accuracy on the validation set will be chosen.

Fig. 4 shows the accuracies obtained on the CIFAR-10 data set with different values of hyperparameters $C_1$ and $C_2$. It can be found in Fig. 4 that the too large or too small value of these hyperparameters will lead to a network of lower accuracy. This is because smaller values of hyperparameters will reduce the influence of losses proposed in this paper, and larger values could lead the student network to ignore the classification loss function. We choose $C_1$ and $C_2$ with the best performance on the validation set.
TABLE VIII
TEN-CLASS CLASSIFICATION RESULTS OF DIFFERENT NETWORKS ON THE CIFAR-10 DATA SET

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>plane</th>
<th>car</th>
<th>bird</th>
<th>cat</th>
<th>deer</th>
<th>dog</th>
<th>frog</th>
<th>horse</th>
<th>ship</th>
<th>truck</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teacher [38]</td>
<td>90.1%</td>
<td>93.8%</td>
<td>86.0%</td>
<td>74.6%</td>
<td>93.5%</td>
<td>86.2%</td>
<td>95.2%</td>
<td>92.6%</td>
<td>95.3%</td>
<td>95.2%</td>
</tr>
<tr>
<td>KD [17]</td>
<td>90.0%</td>
<td>95.2%</td>
<td>83.2%</td>
<td>84.4%</td>
<td>93.2%</td>
<td>87.1%</td>
<td>95.0%</td>
<td>91.6%</td>
<td>97.3%</td>
<td>93.7%</td>
</tr>
<tr>
<td>FitNet [33]</td>
<td>90.7%</td>
<td>97.6%</td>
<td>91.0%</td>
<td>82.7%</td>
<td>93.8%</td>
<td>86.2%</td>
<td>92.7%</td>
<td>93.8%</td>
<td>94.6%</td>
<td>93.5%</td>
</tr>
<tr>
<td>Robust (proposed)</td>
<td>91.0%</td>
<td>97.0%</td>
<td>90.3%</td>
<td>83.6%</td>
<td>92.4%</td>
<td>87.2%</td>
<td>95.4%</td>
<td>93.2%</td>
<td>95.1%</td>
<td>94.1%</td>
</tr>
</tbody>
</table>

TABLE IX
PERFORMANCE OF THE PROPOSED METHOD ON STUDENT NETWORKS WITH VARIOUS ARCHITECTURES

<table>
<thead>
<tr>
<th>Network</th>
<th>#layers</th>
<th>#params</th>
<th>#mult</th>
<th>Speed-up Ratio</th>
<th>Compression Ratio</th>
<th>FitNet</th>
<th>Robust</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teacher [38]</td>
<td>5</td>
<td>~ 9M</td>
<td>~ 725M</td>
<td>×1</td>
<td>×1</td>
<td>90.25%</td>
<td></td>
</tr>
<tr>
<td>Student 1</td>
<td>11</td>
<td>~ 250K</td>
<td>~ 30M</td>
<td>×13.17</td>
<td>×36</td>
<td>89.07%</td>
<td>89.62%</td>
</tr>
<tr>
<td>Student 2</td>
<td>11</td>
<td>~ 862K</td>
<td>~ 108M</td>
<td>×4.56</td>
<td>×10.44</td>
<td>91.02%</td>
<td>91.37%</td>
</tr>
<tr>
<td>Student 3</td>
<td>13</td>
<td>~ 1.6M</td>
<td>~ 392M</td>
<td>×1.40</td>
<td>×5.62</td>
<td>91.16%</td>
<td>91.50%</td>
</tr>
<tr>
<td>Student 4</td>
<td>19</td>
<td>~ 2.5M</td>
<td>~ 382M</td>
<td>×1.58</td>
<td>×3.60</td>
<td>91.64%</td>
<td>91.93%</td>
</tr>
</tbody>
</table>

TABLE X
EXECUTION TIMES OBTAINED ON THE CIFAR-10 DATA SET

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Training Time To converge</th>
<th>Test Time on 10,000 samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teacher</td>
<td>150 min</td>
<td>About 55 seconds</td>
</tr>
<tr>
<td>KD</td>
<td>233 min</td>
<td></td>
</tr>
<tr>
<td>FitNet</td>
<td>400 min</td>
<td>About 8 seconds</td>
</tr>
<tr>
<td>Robust</td>
<td>472 min</td>
<td></td>
</tr>
</tbody>
</table>

We conduct our experiments with PyTorch on an NVIDIA 1080 Ti GPU. Table X reports the execution times of different networks on the CIFAR-10 data set. We find that the computation demand of the student network is much less than that of the teacher network. Since comparison methods share the same architecture of the student network, they have a similar test time. The training time of FitNet and the proposed method are more than that of the teacher network and KD method. FitNet introduced additional networks to teach the student network, while the proposed method needs to calculate gradients of input images and weights in an iteration. Although the training cost of the proposed algorithm is larger, our learned student network receives better forward cost, as shown in Table X. In addition, teacher–student learning mechanism aims to learn a student network that has fewer parameters and faster forward speed. Once the student network has been well trained, we can keep the benefits of less computation, storage cost, and faster forward speed, and do not need to update any parameters of the network during the inference.

VI. CONCLUSION

We proposed to learn a robust student network with the guidance of the teacher network. The proposed method prevented the student network from being disturbed by the perturbations on input examples. Through rigorous theoretical analysis, we proved a lower bound of perturbations that will weaken the confidence of the student network in its prediction. We introduced new objectives based on prediction score and gradients of examples to maximize this lower bound and then improved the robustness of the learned student network to resist perturbations on examples. Experimental results on several benchmark data sets demonstrate the proposed method is able to learn a robust student network with satisfying accuracy and compact size.

REFERENCES


Chang Xu received the Ph.D. degree from Peking University, Beijing, China. He is currently a Lecturer and the ARC DECA Fellow with the School of Computer Science, University of Sydney, Sydney, NSW, Australia. He has authored or coauthored more than 50 papers in prestigious journals and top tier conferences, including the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (T- PAMI), the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS (T-NNSLs), the IEEE TRANSACTIONS ON IMAGE PROCESSING (T-IP), IJCAI, NIPS, and AAAI. His current research interests include machine learning and computer vision.

Tianyu Guo received the B.E degree from Tianjin University, Tianjin, China, in 2016. He is currently pursuing the Ph.D. degree with the Key Laboratory of Machine Perception, Ministry of Education, Peking University, Beijing, China. His current research interests include machine learning and computer vision.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.
Shiyi He received the B.E degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2016. She is currently pursuing the master’s degree with the Key Laboratory of Machine Perception, Ministry of Education, Peking University, Beijing. Her research interests include machine learning and computer vision.

Boxin Shi (M'14) received the B.E. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2007, the M.E. degree from Peking University, Beijing, in 2010, and the Ph.D. degree from the University of Tokyo, Tokyo, Japan, in 2013. From 2013 to 2016, he with the MIT Media Lab, Singapore University of Technology and Design, Nanyang Technological University, Singapore, where he was involved in was postdoctoral research. From 2016 to 2017, he was a Researcher with the National Institute of Advanced Industrial Science and Technology, Tokyo. He is currently Boya Young Fellow Assistant Professor with Peking University, where he leads the Camera Intelligence Group. Dr. Shi was a recipient of the Best Paper Runner-Up Award at International Conference on Computational Photography in 2015. He has served as the Area Chair for ACCV 2018, BMVC 2019, and 3DV 2019.

Chao Xu received the B.E. degree from Tsinghua University, Beijing, China, in 1988, the M.S. degree from University of Science and Technology of China, Hefei, China, in 1991, and the Ph.D. degree from the Institute of Electronics, Chinese Academy of Sciences, Beijing, in 1997. From 1991 and 1994, he was an Assistant Professor with the University of Science and Technology of China, Hefei. Since 1997, he has been with School of Electronics Engineering and Computer Science (EECS), Peking University, Beijing, where he is currently a Professor. His current research interests include image and video coding, processing, and understanding. He has authored or coauthored more than 80 publications and 5 patents in these fields.

Dacheng Tao (F’15) is a Professor of computer science and the ARC Laureate Fellow with the School of Computer Science, Faculty of Engineering, and the Inaugural Director of the UBTECH Sydney Artificial Intelligence Center, University of Sydney, Darlington, NSW, Australia. His research results in artificial intelligence have expounded in 1 monograph and 200+ publications at prestigious journals and prominent conferences, such as IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (T-PAMI), the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS (T-NNLS), IJCV, JMLR, AAAI, IJCAI, NIPS, ICML, CVPR, ICCV, ECCV, ICDM, and KDD, with several best paper awards. Dr. Tao is a fellow of the Australian Academy of Science. He was a recipient of the 2018 IEEE ICDM Research Contributions Award.