Lipschitz Continuity Guided Knowledge Distillation

Yuzhang Shang[1]*, **Bin Duan**[1]*, **Ziliang Zong**[2], **Liqiang Nie**[3], **Yan Yan**[1]†

[1]Department of Computer Science, Illinois Institute of Technology, USA
[2]Department of Computer Science, Texas State University, USA
[3]School of Computer Science and Technology, Shandong University, China

{yshang4, bduan2}@hawk.iit.edu, ziliang@txstate.edu
nieliqiang@gmail.com, yyan34@iit.edu

Abstract

Knowledge distillation has become one of the most important model compression techniques by distilling knowledge from larger teacher networks to smaller student ones. Although great success has been achieved by prior distillation methods via delicately designing various types of knowledge, they overlook the functional properties of neural networks, which makes the process of applying those techniques to new tasks unreliable and non-trivial. To alleviate such problem, in this paper, we initially leverage Lipschitz continuity to better represent the functional characteristic of neural networks and guide the knowledge distillation process. In particular, we propose a novel Lipschitz Continuity Guided Knowledge Distillation framework to faithfully distill knowledge by minimizing the distance between two neural networks' Lipschitz constants, which enables teacher networks to better regularize student networks and improve the corresponding performance. We derive an explainable approximation algorithm with an explicit theoretical derivation to address the NP-hard problem of calculating the Lipschitz constant. Experimental results have shown that our method outperforms other benchmarks over several knowledge distillation tasks (e.g., classification, segmentation and object detection) on CIFAR-100, ImageNet, and PASCAL VOC datasets. Our code is available at https://github.com/42Shawn/LONDON/tree/master.

1. Introduction

Recently, deep learning models have driven great advances in computer vision [13, 8], natural language process [36, 33], information retrieval [42, 43] and multimodal modelling [19, 18]. To meet the buoyant demand of equipping those cumbersome models in resource-constrained edge devices, researchers have proposed several network compression paradigms, such as network pruning [24, 12], network quantization [20] and knowledge distillation (KD) [16]. Among these compression methods, KD helps the training process of a smaller network (student) by transferring knowledge from a larger one (teacher). As one of the first innovators, Hinton *et al.* [16] proposed using soft labels of the larger networks to supervise the training process of the smaller ones. These soft labels are usually interpreted as a form of unseen knowledge distilled from teachers.

Apart from treating soft labels as distilled knowledge, various kinds of knowledge are designed in [45, 14, 38, 41]. For example, Romero *et al.* [34] presented to train intermediate layers of students with guidance of the corresponding layers of teachers, which initiates the subsequent flourishing studies on feature-based knowledge distillation. Researchers [45, 25, 39] also modulated the relations among adjacent feature maps as additional knowledge to assist the training of student networks. Unfortunately, most of these feature-based KD methods solely focus on aligning the shallow information but overlook the high-level information of both networks, *i.e.*, the students mechanically mimicking teachers' actions while neglecting their interior qualities. Thereby, previous studies consider networks as black-boxes and heuristically select features without any functional properties [38, 41, 48], which impedes a universal representative of knowledge to be distilled. To address this problem, we argue that leveraging networks' functional properties to derive high-level knowledge is able to strengthen the performance of KD.

In this paper, we incorporate Lipschitz continuity into KD, considering neural networks as functions rather than black-boxes. By definition in Eq. 4, Lipschitz constant[1] is the upper bound of the relationship between input pertur-

[*] Equal contribution. † Corresponding author.

[1] The Lipschitz constant of a function $\|f\|_{Lip}$ is the maximum norm of its gradient in the domain set, which reflects Lipschitz continuity of the function.

bation and output variation for a given distance, representing the robustness and expressiveness of neural networks [1, 29, 28]. Specifically, authors in [29, 46] demonstrated the effectiveness of the Lipschitz constant by constraining the weights of the discriminator in a generative adversarial network (GAN). Besides, many studies in representation learning [2, 37] demonstrate that deep neural networks are competent in learning high-level information with increasing abstraction. Inspired by this, we devise a scheme to capture the Lipschitz continuity (*i.e.*, calculate the Lipschitz constant for every intermediate block) of the teacher networks and adopt the captured continuity as knowledge to guide the training of student networks. It is worth noting that Lipschitz constant computation is a NP-hard problem [40]. We address this problem by proposing an approximation algorithm with a tight upper bound. In particular, we design a Transmitting Matrix ($\mathbf{TM}$) for each block and calculate the spectral norm of $\mathbf{TM}$ through an adopted iteration method to avoid the high complexity of learning large intermediate matrices. We then aggregate all Lipschitz constants calculated from $\mathbf{TM}$s as the knowledge of the Lipschitz continuity that are transferred to student networks. Importantly, Lipschitz continuity loss function is backpropagation-friendly for training deep networks because of its differentiability.

Overall, the contributions of this paper are four-fold:

- To the best of our knowledge, we are the first on utilizing a high-level functional property, Lipschitz continuity in knowledge distillation, to supervise student networks' training process. In addition, we theoretically explain the effectiveness of our method from the perspective of network regularization and then empirically consolidate this explanation.
- We propose a novel knowledge distillation framework, **L**ipschitz c**ON**tinuity Guided Knowledge **D**istillati**ON** (**LONDON**) for distilling knowledge from the Lipschitz constant.
- To avoid the NP-hard Lipschitz constant calculation, we devise a Transmitting Matrix to numerically approximate the Lipschitz constant of networks in the KD process.
- We perform experiments on different knowledge distillation tasks such as classification, object detection, and segmentation. Our proposed method achieves the state-of-the-art results in these tasks on CIFAR-100, ImageNet, and VOC datasets.

2. Related Work

Lipschitz Continuity and Spectral Norm of Neural Network. The study of adversarial machine learning [23, 31] shows that neural networks are highly vulnerable to attacks based on small modifications of the input to the model at test time, and estimating the regularity of such architectures is essential for practical applications and generalization improvement. Previous efforts [40, 29, 30] have studied one of the critical characteristics to assess the regularity of deep networks: the Lipschitz continuity of deep learning architectures.

Lipschitz constants, which upper bound the relationship between input perturbation and output variation for a given distance, are introduced to secure the robustness of neural networks to small perturbations. This Lipschitz constant $\|f\|_{Lip}$ can be seen as a norm to measure the function's degree of Lipschitz continuity. Apart from some theoretical studies [1, 27, 30] explaining that novel generalization bounds critically rely on the Lipschitz constant of the neural network, Lipschitz continuity of neural networks is widely studied for achieving the state-of-the-art performance in many deep learning topics: (i) In image synthesis [29, 46], researchers used spectral normalization on each layer, an optional approach to constrain the Lipschitz constant of the discriminator for training a GAN on ImageNet, like a regularization term to smooth the discriminator function. And (ii) in adversarial attack machine learning [44], authors propose constraining local Lipschitz constants of neural networks to avoid adversarial attacks.

Aforementioned efforts underline the significance of Lipschitz constant in neural networks' expressiveness and robustness. Particularly, deliberately constraining Lipschitz continuity (constant) in an appropriate range is proven to be a powerful technique for smoothing networks, which can enhance the model's robustness. On account of this, Lipschitz constant, the functional information of neural networks should be introduced into knowledge distillation model for regularizing the training of student networks.

Knowledge Distillation. Apart from the seminal design of soft labels [16], the alignment of intermediate feature maps is also transferred as knowledge to student networks [34]. Researchers continued digging into feature-based outputs and proposed various designs of feature maps' transformation and combination to define the feature-based knowledge, which largely promotes the performance of KD. For example, Heo *et al.* [15, 14] designated an activation boundary of hidden neurons in different positions of networks as knowledge for distillation. In [45], Gram matrixes of neural networks' adjacent feature maps, representing the relation between intermediate layers, are also adopted as a form of knowledge. Authors [25, 4, 39] constructed a similarity measurement for feature representations using singular value decomposition (SVD) to elicit relations between different layers as transferred knowledge.

Inspired by those ideas, many methods are proposed for precisely capturing feature-wise knowledge by artlessly piling up complicated mechanisms on knowledge distillation model. For instance, Wang *et al.* [41] introduced an attention mechanism to assign weights to different CNNs' channels for dynamically determining the critical features to distill. Furthermore, Tian *et al.* [38] introduced contrastive learning to capture correlations and higher-order output dependencies for supervising student network training. This dynamically aligned knowledge almost fully explores potential of distilling networks' output information for supervision.

However, all those feature-based knowledge distillation methods treat neural networks as black-boxes, which are deficient in exploring the functional properties of neural networks via capturing the high-level information. This limitation hinders the applicability and impedes performance improvement. To alleviate the limitation, we introduce Lipschitz continuity to knowledge distillation.

3. Method

In this section, we introduce our proposed knowledge distillation framework. We only elaborate the key derivations in this section due to the limited space. Detailed discussions and technical theorems can be found in the supplemental materials. Here, we focus on capturing the functional property of neural networks as knowledge and transferring it in our distillation method in a numerically accessible way.

3.1. Preliminary

We first define a fully-connected neural network with L layers of widths $d_1, \cdots d_L (d = \sum_{k=1}^{L} d_k)$ as the form of function $f : \mathbb{R}^{d_0} \longmapsto \mathbb{R}^{d_L}$:

$$f(\mathbf{x}) = (T^L \circ \sigma \circ T^{L-1} \circ \cdots \circ \sigma \circ T^1)(\mathbf{x}), \tag{1}$$

where each $T^{(k)} : \mathbb{R}^{d_{k-1}} \longmapsto \mathbb{R}^{d_k}$ is an affine function (d_0 and d_L are the sizes of network's input and output feature maps) and σ performs element-wise activation for feature maps. For k-th layer of the networks, $T^{(k)}(\mathbf{u}) = \mathbf{W}^k\mathbf{u} + \mathbf{b}^k$, where $\mathbf{W}^k$ and $\mathbf{b}^k$ stand for the weight matrix and bias vector, respectively. For generality purpose, we discard the bias term of the network, so that the network can be simplified as:

$$f(\mathbf{W}^1, \cdots, \mathbf{W}^L; \mathbf{x}) = (\mathbf{W}^L \circ \sigma \circ \mathbf{W}^{L-1} \circ \cdots \circ \sigma \circ \mathbf{W}^1)(\mathbf{x}). \tag{2}$$

Notably, it is sufficient to consider networks with the most straightforward fully-connected layers, since layer with complex structures such as convolution layer can also be denoted as the form of matrix multiplication. We consider a convolution layer with i input channels and o output channels, and the size of the kernel is $w \times h$, resulting in $iowh$ parameters. We can re-arrange the parameters to a matrix of size $o \times ihw$, such that this convolution layer can also be processed in the same way as the other fully-connected layers do. Hence, our analysis has no loss for generality in this configuration of function f.

Following Eq. 2, we define the function form of the teacher network as $f_T(\mathbf{W}_T^1, \cdots, \mathbf{W}_T^{L_T}; \mathbf{x})$, and the student network as $f_S(\mathbf{W}_S^1, \cdots, \mathbf{W}_T^{L_S}; \mathbf{x})$, such that the feature-based KD paradigm can be interpreted as:

$$\forall \mathbf{x} \in \mathbf{Data}, \quad \underset{\mathbf{W}_S^1, \cdots, \mathbf{W}_S^{L_S}}{\arg\min} \quad Dist(\mathcal{T}(f_T(\mathbf{x})), \mathcal{T}(f_S(\mathbf{x}))), \tag{3}$$

where given the same data, the ultimate goal of KD paradigm is to minimize the distance between teacher and student for optimizing the latter's parameters $\{\mathbf{W}_S^i\}$. Particularly, $Dist()$ is a distance function, and $\mathcal{T}()$ is certain transformation approach to turn feature maps into more measurable and learnable knowledge. By utilizing those designed knowledge, the student network is forced to mimic the teacher network and hopefully obtains comparable performance with lighter architecture.

Here, we introduce Lipschitz Continuity into KD paradigm as universal information of neural networks based on the functional property of networks. To make Lipschitz constant calculation numerically feasible, we further propose an approximation for the Lipschitz constant and use power iteration method to calculate this approximation.

3.2. The functional information of neural networks: Lipschitz Continuity

Definition 1. A function $f : \mathbb{R}^n \longmapsto \mathbb{R}^m$ is called Lipschitz continuous if there exists a constant L such that:

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n, \|f(\mathbf{x}) - f(\mathbf{y})\|_2 \leq L\|\mathbf{x} - \mathbf{y}\|_2. \tag{4}$$

The smallest L that can hold the inequality is called Lipschitz constant of function f, denoted as $\|f\|_{Lip}$. By Definition 1, $\|f\|_{Lip}$ has an excellent property of upper bounding the relationship between input perturbation and output variation for a given distance (generally L2 norm), thus it is considered as a metric to evaluate the robustness of neural networks to small perturbations [27, 40, 1]. However, computing the exact Lipschitz constant of neural networks in the knowledge distillation process is a NP-hard problem [40]. To solve this problem, we propose a feasible and effective method to approximate the Lipschitz constants in KD.

We first define the affine function for the k-th layer $T^k : \mathbf{fm}^{k-1} \longmapsto \mathbf{fm}^k$, in which $\mathbf{fm}^{k-1} \in \mathbb{R}^{d_{k-1}}$ and $\mathbf{fm}^k \in \mathbb{R}^{d_k}$ are the feature maps out of the $(k-1)$th and the kth layer, respectively.

By **Lemma** 1 as in Supplemetary Appendix, we have $\|T^k\|_{Lip} = \sup_{\mathbf{fm}} \|\nabla T^k(\mathbf{fm})\|_{SN}$, where $\|\cdot\|_{SN}$ is the spectral norm of matrix. And the matrix spectral norm $\|\cdot\|_{SN}$ is formally defined by

$$\|\mathbf{W}\|_{SN} \triangleq \max_{\mathbf{x}:\mathbf{x}\neq\mathbf{0}} \frac{\|\mathbf{W}\mathbf{x}\|_2}{\|\mathbf{x}\|_2} = \max_{\|\mathbf{x}\|_2 \leq 1} \|\mathbf{W}\mathbf{x}\|_2, \tag{5}$$

where the spectral norm of matrix $\mathbf{W}$ is equivalent to its largest singular value. Thus, for the linear layer

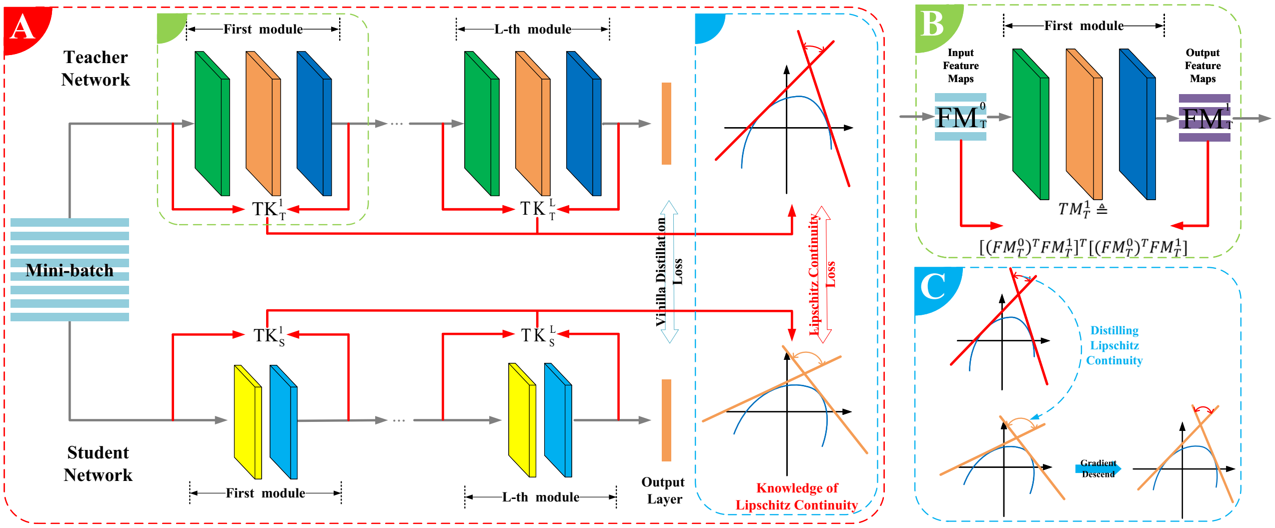

Figure 1. An overview of our proposed LONDON is indicated in **A**. For the teacher-student backbone, besides the traditional knowledge distillation loss, our proposed Lipschitz continuity distillation loss is the key element. The input and output feature maps of each module are used to format the Transmitting Matrix $\mathbf{TM}^k$ for approximating the module's Spectral Norm as demonstrated in **B**. Those Spectral Norms are combined to calculate the Lipschitz constant of networks for further distillation via our designed Lipschitz continuity loss function $\mathcal{L}_{Lip}$, which regularizes the student training in a high level showed in **C**.

$T^k(\mathbf{fm}^{k-1}) = \mathbf{fm}^k$, based on Lemma 2 in Supplemetary Appendix, its Lipschitz constant is given by

$$\|T^k\|_{Lip} = \sup_{\mathbf{fm}} \|\nabla T^k(\mathbf{fm})\|_{SN} = \|\mathbf{W}\|_{SN}. \quad (6)$$

Additionally, most activation functions such as ReLU, Leaky ReLU, Tanh, Sigmoid as well as max-pooling, have a Lipschitz constant equal to 1. As for other common neural network layers such as dropout, batch normalization and other pooling methods, they all have simple and explicit Lipschitz constants [10]. This fixed Lipschitz constant property renders our derivation applicable to most network architectures, such as ResNet [13] and MobileNet [17].

Thereafter, we use the inequality (concluded by Eq. 7 in [1]) $\|T^k \circ T^{k+1}\|_{Lip} \leq \|T^k\|_{Lip} \cdot \|T^{k+1}\|_{Lip}$ to derive the following bound for $\|f\|_{Lip}$:

$$\begin{aligned} \|f\|_{Lip} &\leq \|T^L\|_{Lip} \cdot \|\sigma\|_{Lip} \cdot \|T^{L-1}\|_{Lip} \cdots \|T^1\|_{Lip} \\ &= \prod_{k=1}^{L} \|T^k\|_{Lip} = \prod_{k=1}^{L} \|\mathbf{W}^k\|_{SN}. \end{aligned} \quad (7)$$

In this way, we transfer the teacher's Lipschitz constant to the student through a sequence of spectral norm of intermediate layers in the network. Moreover, the upper bound of Lipschitz constant also ensures the quality of knowledge to be transferred.

3.3. Transmitting Matrix

Given the derived tight upper bound of the Lipschitz constant, we design a novel loss to distill Lipschitz continuity from teacher to student by narrowing the distance between corresponding $\|\mathbf{W}_T^k\|_{SN}$ and $\|\mathbf{W}_S^k\|_{SN}$ down. The first problem is how to calculate each spectral norm. Calculating the spectral norm of weight matrix $\mathbf{W}^k$ in neural networks by SVD is inaccessible. Specifically, for the complex network structures such as convolutions layers or residual modules, though they can be re-arranged matrix-wisely, their spectral norm's computation is impractical. Therefore, we propose using Transmitting Matrix (TM) to bypass the complicated calculation of the spectral norm $\mathbf{W}^k$. This approximate calcuation allows feasible computation to distill Lipschitz constant and its further use as a loss function.

For training data of batch size N, after a forward process for the (k-1)th layer, we have a batch of corresponding feature maps as

$$\mathbf{FM}^{k-1} = (\mathbf{fm}_1^{k-1}, \mathbf{fm}_2^{k-1}, \cdots, \mathbf{fm}_n^{k-1}) \in \mathbb{R}^{d_{k-1} \times N}, \quad (8)$$

where $\mathbf{W}^k\mathbf{FM}^{k-1} = \mathbf{FM}^k$ for each $k \in \{1, \ldots, L\}$.

Studies [3, 39] about similarity of feature maps illustrate that for well-trained networks, their batch of feature maps in the same layer $\{\mathbf{fm}_i^{k-1}\}, i \in \{1, \ldots, n\}$ have strong mutual linear independence. We formalize the relevance of feature maps in the same layer as

$$\forall i \neq j \in \{1, \cdots, N\}, (\mathbf{fm}_i^{k-1})^{\top}\mathbf{fm}_j^{k-1} \approx 0, \quad (9)$$

$$\forall i \in \{1, \cdots, N\}, (\mathbf{fm}_i^{k-1})^{\top}\mathbf{fm}_i^{k-1} \neq 0. \quad (10)$$

We further normalize the feature maps by $\forall i \in \{1, \cdots, N\}, \mathbf{fm}_i^{k-1} = \frac{\mathbf{fm}_i^{k-1}}{\|(\mathbf{fm}_i^{k-1})\|_2}$ such that a batch of feature maps can be expressed in a vector representation that

$$(\mathbf{FM}^{k-1})^{\top}\mathbf{FM}^{k-1} \approx \mathbf{I}, \quad (11)$$

where $\mathbf{I}$ is an unit matrix.

With all the aforementioned equations, we are ready to define the transmitting matrix $\mathbf{TM}^k$ for calculating the spectral norm of matrix $\mathbf{W}^k$ as calculating the spectral norm of $\mathbf{TM}^k \triangleq \left[(\mathbf{FM}^{k-1})^\mathsf{T}\mathbf{FM}^k\right]^\mathsf{T}\left[(\mathbf{FM}^{k-1})^\mathsf{T}\mathbf{FM}^k\right]$

$$\begin{aligned}&= (\mathbf{W}^k\mathbf{FM}^{k-1})^\mathsf{T}(\mathbf{FM}^{k-1})\left[(\mathbf{FM}^{k-1})^\mathsf{T}\mathbf{W}^k\mathbf{FM}^{k-1}\right]\\ &= (\mathbf{FM}^{k-1})^\mathsf{T}(\mathbf{W}^k)^\mathsf{T}(\mathbf{FM}^{k-1})(\mathbf{FM}^{k-1})^\mathsf{T}\mathbf{W}^k\mathbf{FM}^{k-1}.\end{aligned} \tag{12}$$

Eq. 11 and 12 together yield the result as

$$\mathbf{TM}^k \approx (\mathbf{FM}^{k-1})^\mathsf{T}(\mathbf{W}^{k\mathsf{T}}\mathbf{W}^k)\mathbf{FM}^{k-1}. \tag{13}$$

Theorem 1. If matrix $\mathbf{U}$ is an orthogonal matrix, such that $\mathbf{U}^\mathsf{T}\mathbf{U} = \mathbf{I}$, where $\mathbf{I}$ is an unit matrix, the largest eigenvalues of $\mathbf{U}^\mathsf{T}\mathbf{H}\mathbf{U}$ and $\mathbf{H}$ are equivalent.

$$\sigma_1(\mathbf{U}^\mathsf{T}\mathbf{H}\mathbf{U}) = \sigma_1(\mathbf{H}), \tag{14}$$

where $\sigma_1(\cdot)$ is the largest eigenvalue of a matrix. Based on Theorem 1 and Eq. 13, our defined transmitting matrix $\mathbf{TM}^k$ has the same largest eigenvalue with $\mathbf{W}^{k\mathsf{T}}\mathbf{W}^k$, *i.e.*$\sigma_1(\mathbf{TM}^k) = \sigma_1(\mathbf{W}^{k\mathsf{T}}\mathbf{W}^k)$. Thus, combining the definition of spectral norm $\|\mathbf{W}^k\| = \sigma_1(\mathbf{W}^{k\mathsf{T}}\mathbf{W}^k)$, we can achieve the spectral norm of matrix $\mathbf{W}^k$ by calculating the largest eigenvalue of $\mathbf{TM}^k$, $\sigma_1(\mathbf{TM}^k)$, which is solvable.

For networks with more complicated layers such as residual blocks, by considering the block as an affine mapping from front feature maps to back feature maps, this approximation is applicable to calculate the spectral norm block-by-block instead of layer-by-layer, which makes our spectral norm calculation more efficient. To this end, we define the Transmitting Matrix $\mathbf{TM}$ for residual blocks as

$$\mathbf{TM}_m^k \triangleq \left[(\mathbf{FM}^f)^\mathsf{T}\mathbf{FM}^l\right]^\mathsf{T}\left[(\mathbf{FM}^f)^\mathsf{T}\mathbf{FM}^l\right], \tag{15}$$

where the $\mathbf{FM}^f$ and $\mathbf{FM}^l$ are the front feature maps and latter feature maps of a residual block.

3.4. Approximating the Spectral Norm with Power Iteration Method

Following the aforementioned steps, we next need to calculate the spectral norms of two matrices (teacher and student) and then calculate the loss between those two. The intuitive approach is using SVD to compute the spectral norm, which results in overloaded computation. Importantly, the SVD calculation is non-differentiable, making it impossible to train the deep networks. Instead of using SVD, we utilize power iteration method [9, 46, 29] to approximate the spectral norm of the targeted matrix with a small trade-off of accuracy, as presented in Algorithm 1.

Algorithm 1 Compute spectral norm using power iteration

```
Input: targeted matrix $\mathbf{TM}$, stop condition $res_{stop}$.
Output: the spectral norm of matrix $\mathbf{TM}$, $\|\mathbf{TM}\|_{SN}$.
initialize $\mathbf{v}_0 \in \mathbb{R}^m$ with a random vector.
while $res \geq res_{stop}$ do
    $\mathbf{v}_{i+1} \leftarrow \mathbf{TM}\mathbf{v}_i / \|\mathbf{TM}\mathbf{v}_i\|_2$
    $res = \|\mathbf{v}_{i+1} - \mathbf{v}_i\|_2$
end while
return $\|\mathbf{TM}\|_{SN} = \mathbf{v}_{i+1}^\mathsf{T}\mathbf{TM}\mathbf{v}_i$
```

In this way, we have a feasible approach to calculate the spectral norms of $\mathbf{TM}$s which can faithfully approximate the Lipschitz constant of networks.

3.5. Overall Loss Function

By using Algorithm 1, we obtain the spectral norms for teacher and student networks, respectively: $\|\mathbf{TM}_T^i\|_{SN}$ and $\|\mathbf{TM}_S^i\|_{SN}$ for each $i \in \{1, \ldots, L\}$. We define our novel lipschitz continuity loss function $\mathcal{L}_{Lip}$ as

$$\mathcal{L}_{Lip} = \sum_{i=1}^{L-1}\left(\frac{\|\mathbf{TM}_T^i\|_{SN} - \|\mathbf{TM}_S^i\|_{SN}}{\beta^{L-1-i}}\right)^2, \tag{16}$$

where β is a coefficient greater than 1. Hence, the β^{L-1-i} decreases with i increasing and consequently the $\frac{\|\mathbf{TM}_T^i\|_{SN} - \|\mathbf{TM}_S^i\|_{SN}}{\beta^{L-1-i}}$ increases. In this way, we give more weight on higher layer features since they are closer to the features performing tasks.

Combined with the cross entropy loss $\mathcal{L}_{CE}$ and vanilla knowledge distillation loss $\mathcal{L}_{KD}$, we are ready to propose our novel loss function as

$$\mathcal{L} = \frac{\lambda}{2} \cdot \mathcal{L}_{Lip} + \mathcal{L}_{KD} + \mathcal{L}_{CE}, \tag{17}$$

where λ is used to control the degree of distilling the Lipschitz constant. We use $\frac{\lambda}{2}$ because when taking derivative of $\mathcal{L}_{Lip}$, the denominator part can be easily eliminated.

3.6. Explaining the Effectiveness from a Regularization Perspective

The derivative of the loss function $\mathcal{L}$ with respect to W:

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial\mathbf{W}} &= \frac{\partial(\mathcal{L}_{CE} + \mathcal{L}_{KD})}{\partial\mathbf{W}} + \frac{\partial(\mathcal{L}_{Lip})}{\partial\mathbf{W}}\\ &= \mathbf{M} - \lambda\sum_{i=1}^{L-1}\left(\frac{\|\mathbf{TM}_T^i\|_{SN} - \|\mathbf{TM}_S^i\|_{SN}}{\beta^{L-1-i}}\right)\frac{\partial\|\mathbf{TM}_S^i\|_{SN}}{\partial\mathbf{W}}\\ &\approx \mathbf{M} - \lambda\sum_{i=1}^{L-1}\left(\frac{\|\mathbf{TM}_T^i\|_{SN} - \|\mathbf{TM}_S^i\|_{SN}}{\beta^{L-1-i}}\right)\frac{\partial\|W_S^i\|_{SN}}{\partial\mathbf{W}}\\ &= \mathbf{M} - \lambda\sum_{i=1}^{L-1}\left(\frac{\|\mathbf{TM}_T^i\|_{SN} - \|\mathbf{TM}_S^i\|_{SN}}{\beta^{L-1-i}}\right)\mathbf{u}_1^i(\mathbf{v}_1^i)^\mathsf{T},\end{aligned} \tag{18}$$

where $\mathbf{M} \triangleq \frac{\partial(\mathcal{L}_{CE}+\mathcal{L}_{KD})}{\partial \mathbf{W}}$, $\mathbf{u}_1^i$ and $\mathbf{v}_1^i$ are respectively the first left and right singular vectors of $\mathbf{W}_S^i$. For $\mathbf{W}_S^i$, using SVD, we have

$$\mathbf{W}_S^i = \sum_{j=1}^{d_i} \sigma_j(\mathbf{W}_S^i)\mathbf{u}_j^i\mathbf{v}_j^i, \tag{19}$$

where d_i is the rank of $\mathbf{W}_S^i$, $\sigma_j(\mathbf{W}_S^i)$ is the j-th biggest singular value, $\mathbf{u}_j^i$ and $\mathbf{v}_j^i$ are correspondingly left and singular vectors, respectively.

In Eq. 18, the first term $\mathbf{M}$ is the same as the derivative of the loss function of vanilla knowledge distillation. As for the second term, based on Eq. 19, it can be seen as the regularization term penalizing the vanilla knowledge distillation loss with an adaptive regularization coefficient

$$\gamma \triangleq \lambda \frac{\|\mathbf{TM}_T^i\|_{SN} - \|\mathbf{TM}_S^i\|_{SN}}{\beta^{L-1-i}}, \tag{20}$$

which constrains the weights of the student networks by utilizing teacher networks' $\|\mathbf{TM}_T^i\|_{SN}$ as a prior supervision information. In other words, our method prevents the student networks from trapping into local minima. In this way, it ensures better training of student networks. We demonstrate performance by designing a corresponding experiment in Section 4.4, showing that our proposed method prevents student networks from over-fitting the dataset.

4. Experiments

In this section, we conducted experiments on three computer vision tasks, image classification, object detection, and segmentation, to validate the effectiveness of our proposed distillation method. In addition to comparing our method with the state-of-the-art methods, we also designed a series of ablation studies to verify the effectiveness and highlight the regularization property of our proposed technique. All experiments are implemented using PyTorch [32].

4.1. Classification

We chose CIFAR-100 [22] for classification. This is because it is commonly used for comparing KD methods and its relatively small size provides flexibility of implementing different combinations of teacher and student architectures. Besides CIFAR-100, we conducted experiments on ImageNet [6], a larger dataset, to verify the stability of our distillation method.

CIFAR-100 [22] is the most widely-used image classification dataset, which consists of 50K training images and 10K testing images of size 32×32 divided into 100 classes. Specifically, we designed various combinations of architectures for teacher and student networks. Table 1 summarizes the settings of each experiment, model size and compression ratio, involving architectures such as Residual Network (ResNet) [13], Wide Residual Network (WideResNet) [47], and Deep Pyramidal Residual Networks (PyramidNet) [11]. Experimental results of different settings are shown in Table 2, where it is obvious that our method achieves state-of-the-art performance in all seven settings, for both depth and channel compression (**a, b, c**) and different architectures (**d, e, f, g**). Especially, in the setting of depth compression and channel compression (a) and (b), the student networks trained by LONDON even outperform the teacher networks, which further demonstrates the efficacy of our Lipschitz continuity method as a regularization function.

Overall, our proposed method consistently shows comparable or better performance regardless of different compression rates or other network architecture types, which endows our approach with more implementation flexibility. We noted exciting improvements in student networks along with a high compression ratio. Therefore, our results present the potential of using Lipschitz continuity distillation to compress large networks into more resource-efficient ones with acceptable accuracy drop. For example, when the setting (g) is a 17× compression from teacher network to student network with completely different architecture, the student network still benefits from the teacher network via our method. In general, our proposed method can be applied to small networks (fewer parameters) and large networks with satisfactory performance.

ImageNet [6] is a large-scale dataset with 1.2 million training images and 50k validation images divided into 1,000 classes. Compared to other classification datasets such as CIFAR-100, ImageNet has greater diversity, and its image is larger in scale (average 469×387). For all experiments, we reported both the top-1 and top-5 accuracies. Images are cropped to the size of 224×224 for training and validation. The student networks are trained for 100 epochs, and the learning rate begins at 0.1 multiplied by 0.1 at every 30 epochs. To ensure a fair comparison, we used the pre-trained models in the PyTorch library as the teacher networks. Two combinations of network architectures are settled for demonstration. For the first combination, we chose ResNet152 [13] as the teacher network and ResNet50 as the student network. As the second one, for testing the knowledge distillation capacity across different network architectures, we chose ResNet50 as the teacher network, and MobileNet [17] as the student network. The results are displayed in Table 3. Compared to strong methods such as [15, 14], our method still exhibits a great improvement. In particular, our method makes ResNet50 outperform the teacher network ResNet152, which is a remarkable achievement. Besides, regarding the compression ability, our method makes a considerable improvement in the lightweight architecture, MobileNet, where the error rate of 27.64% of our method is better than any network reported in the paper of MobileNet [17].

Setup	Compression type	Teacher network	Student network	# of params teacher	# of params student	Compress ratio
(a)	Depth	WideResNet 28-4	WideResNet 16-4	5.87M	2.77M	47.2%
(b)	Channel	WideResNet 28-4	WideResNet 28-2	5.87M	1.47M	25.0%
(c)	Depth & channel	WideResNet 28-4	WideResNet 16-2	5.87M	0.70M	11.9%
(d)	Different architecture	WideResNet 28-4	ResNet 56	5.87M	0.86M	14.7%
(e)	Different architecture	PyramidNet-200 (240)	WideResNet 28-4	26.84M	5.87M	21.9%
(f)	Different architecture	PyramidNet-200 (240)	PyramidNet-110 (84)	26.84M	3.91M	14.6%
(g)	Different architecture	PyramidNet-200 (240)	ResNet 56	26.84M	0.86M	5.8%

Table 1. Seven experimental settings with different network topological structures on CIFAR-100.

Setup	Teacher	Baseline	KD [16]	FitNets [34]	AT [48]	Jacobian [35]	FT [21]	AB [15]	OFD [14]	AFD [41]	LONDON (**ours**)
(a)	21.09	22.72	21.69	21.85	22.07	22.18	21.72	21.36	20.89	21.15	**20.33**
(b)	21.09	24.88	23.43	23.94	23.80	23.70	23.41	23.19	21.98	21.79	**20.71**
(c)	21.09	27.32	26.47	26.30	26.56	26.71	25.91	26.02	24.08	24.21	**23.46**
(d)	21.09	27.68	26.76	26.35	26.66	26.60	26.20	26.04	24.44	24.67	**23.78**
(e)	15.57	21.09	20.97	22.16	19.28	20.59	19.04	20.46	17.80	18.24	**17.54**
(f)	15.57	22.58	21.68	23.79	19.93	23.49	19.53	20.89	18.89	19.32	**18.21**
(g)	15.57	27.68	26.82	26.10	26.64	26.43	26.29	25.70	24.49	24.53	**23.52**

Table 2. Top-1 error rates (%) of 7 different combinations in Table 1 on CIFAR-100 test set. Lower is better. *Baseline* represents a result without distillation. For all the results we used author-provided code or author-reported results. Each result is averaged over 5 runs.

Network	# of params (ratio)	Method	Top-1 error	Top-5 error
ResNet152	60.19M	Teacher	21.69	5.95
ResNet50	25.56M (42.5%)	Baseline	23.72	6.97
		KD [16]	22.85	6.55
		AT [48]	22.75	6.35
		FT [21]	22.80	6.49
		AB [15]	23.47	6.94
		OFD [14]	21.65	5.83
		AFD [41]	22.08	6.30
		LONDON (**ours**)	**21.12**	**5.47**
ResNet50	25.56M	Teacher	23.84	7.14
MobileNet	4.23M (16.5%)	Baseline	31.13	11.24
		KD [16]	31.42	11.02
		AT [48]	30.44	10.67
		FT [21]	30.12	10.50
		AB [15]	31.11	11.29
		OFD [14]	28.75	9.66
		AFD [41]	28.61	9.81
		LONDON (**ours**)	**27.64**	**8.97**

Table 3. Top-1 and Top-5 error rates (in percentage) of different combinations of the student and teacher's network structures on ImageNet validation set. 'baseline' represents a result without distillation technique. Lower is better.

Network	# of params (ratio)	Method	mAP
ResNet50-SSD	36.7M	Teacher	76.79
ResNet18-SSD	20.0M (54.5%)	Baseline	71.61
		OFD [14]	73.08
		AFD [41]	72.78
		LONDON (**ours**)	**73.82**
MobileNet-SSD	6.5M (18.7%)	Baseline	67.58
		OFD [14]	68.54
		AFD [41]	68.63
		LONDON (**ours**)	**69.09**

Table 4. Object detection results in PASCAL VOC2007 testing set. Results are described in mean Average Precision (mAP). Higher is better.

4.2. Object Detection

We applied our proposed method on the most popular high-speed detector, Single Shot Detector (SSD) [26]. All models are trained with the training set of VOC2007 and VOC2012 [7] where the backbone networks are pre-trained using the ImageNet dataset. All models are trained for 120k iterations with a batch size of 32. We set the SSD trained with no distillation as our baseline and SSD detector with ResNet50 as the teacher network. As for the student networks, we used SSD with ResNet18, or MobileNet [17]. We evaluated the detection performance in the VOC2007 testing set. The result is presented in Table 4. Both trained student networks outperform other methods. This implies that our method can be applied to object detector. Furthermore, we found that the distillation between similar structures has better quality than the different ones by comparing the performance of ResNet18 to MobileNet.

4.3. Semantic Segmentation

In this section, we conducted knowledge distillation on semantic segmentation task. It is worth noting that implementing KD on semantic segmentation is extremely difficult for the penultimate feature maps of the segmentation

Backbone	# of params (ratio)	Method	mIoU
ResNet101	59.3M	Teacher	77.39
ResNet18	16.6M (28.0%)	Baseline	71.79
		OFD [14]	73.24
		AFD [41]	72.81
		LONDON (**ours**)	**73.62**
MobileNet	5.8M (9.8%)	Baseline	68.44
		OFD [14]	71.36
		AFD [41]	71.56
		LONDON (**ours**)	**71.97**

Table 5. Semantic segmentation on the PASCAL VOC 2012 testing set. Results are described in mean Intersection over Union (mIoU). Higher is better.

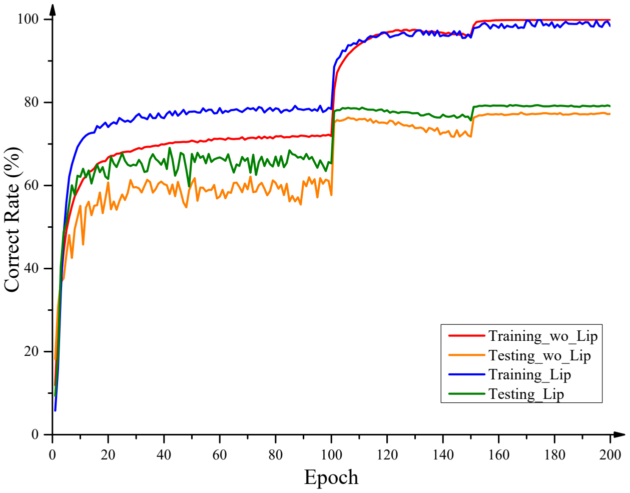

Figure 2. Our proposed loss can mitigate overfitting.

model, which has higher dimensions than common network architectures. In particular, the widely-used DeepLabV3+ [5] is taken as our study case for semantic segmentation. We used DeepLabV3+ with the backbone of ResNet101 as the teacher, and DeepLabV3+ based on ResNet18[13] and MobileNetV2 [17] as the students. The results shown in Table 5 provide clear evidence that our proposed method can greatly improve the performance of both ResNet18 and MobileNet.

In general, most KD studies are only experimentally justified over the task of image classification. In our case, experiments on detection and segmentation verify that our method can be applied to not only image classification but also other computer vision tasks. The flexibility without significant model modifications is an advantage of our high-level knowledge distillation so that our proposed method has a wide range of potential applications.

4.4. Analyses

Mitigate Overfitting. As demonstrated in Section 3.6, our Lipschitz distillation loss can be seen as a regularization term, which constrains the search space around the point inferred by the teacher so as to prevents overfitting the target dataset. To consolidate this theoretical demonstration, we design a corresponding experiment. We use the setting (b) in Table 1 to study this regularization phenomenon. The results are shown in Figure 2. It is noteworthy that when turning off the Lipschitz continuity loss module, the performance on the validation set drops while the training correct rate stays at the same level. This overfit-reduction phenomenon verifies that our proposed method improves the student network training by regularization.

λ Pair	0	0.1	0.4	1.6	3.2	6.4
(a)	21.69	21.36	21.54	21.11	**20.33**	21.87
(b)	23.43	22.04	22.05	21.88	**21.48**	22.35
(c)	26.47	24.39	23.77	**23.56**	23.62	24.87
(d)	27.68	24.18	24.42	23.82	**23.78**	25.22

Table 6. Ablation study of our method. The results are presented in the form of error rate (%). Lower is better.

Ablative Experiments. We conducted an ablation study of our proposed method in CIFAR-100 with the teacher and student architecture pairs in Table 1. By adjusting the coefficient λ in the loss function $\mathcal{L}_{London}$ (Eq. 16, 17), where $\lambda = 0$ equals to no Lipschitz continuity distilled as our baseline. The results are shown in Table 6. With λ increasing, the performance improvements show the effectiveness of our designed Lipschitz continuity loss. However, when the ratio of $\mathcal{L}_{Lip}$ in $\mathcal{L}_{London}$ is greater than 20% (on average), LONDON's performance drops. A well-trained student network should have both the ability to align low-level feature maps and capture the high-level information. Therefore, we believe that putting too much weight on high-level and universal information loses the aligning ability that the network would have.

5. Conclusion

We investigate the knowledge distillation and Lipschitz continuity of neural networks. Specifically, we present a novel KD method, named LONDON, which numerically calculates and transfers the Lipschitz constant as knowledge. Compared to standard KD methods considering neural networks as black-boxes, our KD method captures the functional property of neural networks as high-level knowledge for training student networks, which further prevents the students networks from overfitting the datasets by extending the representational capability of KD.

Acknowledgements. This research was partially supported by NSF CNS-1908658, NeTS-2109982 and the gift donation from Cisco. This article solely reflects the opinions and conclusions of its authors and not the funding agents.

References

[1] Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In *NeurIPS*, 2017.

[2] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *TPAMI*, 2013.

[3] Dapeng Chen, Zejian Yuan, Jingdong Wang, Badong Chen, Gang Hua, and Nanning Zheng. Exemplar-guided similarity learning on polynomial kernel feature map for person reidentification. *IJCV*, 2017.

[4] Hanting Chen, Yunhe Wang, Chang Xu, Chao Xu, and Dacheng Tao. Learning student networks via feature embedding. *TNNLS*, 2020.

[5] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *TPAMI*, 2018.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[7] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *IJCV*, 2015.

[8] Ross Girshick. Fast r-cnn. In *CVPR*, 2015.

[9] Gene H Golub and Henk A Van der Vorst. Eigenvalue computation in the 20th century. *JCAM*, 2000.

[10] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*. 2016.

[11] Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. In *CVPR*, 2017.

[12] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *ICLR*, 2016.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[14] Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *ICCV*, 2019.

[15] Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *AAAI*, 2019.

[16] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NeurIPS*, 2014.

[17] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. In *NeurIPS*, 2017.

[18] Yupeng Hu, Meng Liu, Xiaobin Su, Zan Gao, and Liqiang Nie. Video moment localization via deep cross-modal hashing. *TIP*, 2021.

[19] Yupeng Hu, Liqiang Nie, Meng Liu, Kun Wang, Yinglong Wanga, and Xiansheng Hua. Coarse-to-fine semantic alignment for cross-modal moment localization. *TIP*, 2021.

[20] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *NeurIPS*, 2016.

[21] Jangho Kim, SeoungUK Park, and Nojun Kwak. Paraphrasing complex network: Network compression via factor transfer. In *NeurIPS*, 2018.

[22] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[23] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *ICLR*, 2017.

[24] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. In *NeurIPS*, 1989.

[25] Seung Hyun Lee, Dae Ha Kim, and Byung Cheol Song. Self-supervised knowledge distillation using singular value decomposition. In *ECCV*, 2018.

[26] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016.

[27] Ulrike von Luxburg and Olivier Bousquet. Distance-based classification with lipschitz functions. *JMLR*, 2004.

[28] Jiancheng Lyu, Shuai Zhang, Yingyong Qi, and Jack Xin. Autoshufflenet: Learning permutation matrices via an exact lipschitz continuous penalty in deep convolutional neural networks. In *SIGKDD*, 2020.

[29] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018.

[30] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *NeurIPS*, 2017.

[31] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv:1605.07277*, 2016.

[32] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

[33] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

[34] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.

[35] Suraj Srinivas and François Fleuret. Knowledge transfer with jacobian matching. In *ICML*, 2018.

[36] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NeurIPS*, 2014.

[37] Haiman Tian, Yudong Tao, Samira Pouyanfar, Shu-Ching Chen, and Mei-Ling Shyu. Multimodal deep representation learning for video classification. In *WWW*, 2019.

[38] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *ICLR*, 2020.

[39] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *CVPR*, 2019.

[40] Aladin Virmaux and Kevin Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. In *NeurIPS*, 2018.

[41] Kafeng Wang, Xitong Gao, Yiren Zhao, Xingjian Li, Dejing Dou, and Cheng-Zhong Xu. Pay attention to features, transfer learn faster cnns. In *ICLR*, 2020.

[42] Yinwei Wei, Xiang Wang, Weili Guan, Liqiang Nie, Zhouchen Lin, and Baoquan Chen. Neural multimodal cooperative learning toward micro-video understanding. *TIP*, 2019.

[43] Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua. Mmgcn: Multi-modal graph convolution network for personalized recommendation of micro-video. In *ACM MM*, 2019.

[44] Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. Evaluating the robustness of neural networks: An extreme value theory approach. In *ICLR*, 2018.

[45] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *CVPR*, 2017.

[46] Yuichi Yoshida and Takeru Miyato. Spectral norm regularization for improving the generalizability of deep learning. *arXiv:1705.10941*, 2017.

[47] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016.

[48] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *NeurIPS*, 2017.