KNOWLEDGE DISTILL VIA LEARNING NEURON MANIFOLD

Anonymous authors
Paper under double-blind review

ABSTRACT

Although deep neural networks show their extraordinary power in various tasks, they are not feasible for deploying such large models on embedded systems due to high computational cost and storage space limitation. The recent work knowledge distillation (KD) aims at transferring model knowledge from a well-trained teacher model to a small and fast student model which can significantly help extending the usage of large deep neural networks on portable platform. In this paper, we show that, by properly defining the neuron manifold of deep neuron network (DNN), we can significantly improve the performance of student DNN networks through approximating neuron manifold of powerful teacher network. To make this, we propose several novel methods for learning neuron manifold from DNN model. Empowered with neuron manifold knowledge, our experiments show the great improvement across a variety of DNN architectures and training data. Compared with other KD methods, our Neuron Manifold Transfer (NMT) has best transfer ability of the learned features.

1 INTRODUCTION

In recent years, deep neural networks become more and more popular in computer vision and neural language processing. A well-trained learning model shows its power on tasks such as image classification, object detection, pattern recognizing, live stream analyzing, etc. We also have the promise that given enough data, deeper and wider neural networks can achieve better performance than the shallow networks(Ba & Caruana (2014)). However, these larger but well-trained networks also bring in high computational cost, and leave large amount of memory footprints which make these models very hard to travel and reproduce(Geoffrey et al. (2015)). Due to this drawback, a massive amount of trainable data gathered by small devices such as mobiles, cameras, smart sensors, etc. is unable to be utilized in the local environment which can cause time-sensitive prediction delay and other impractical issues.

To address the above issues, recently, there are extensive works proposed to mitigate the problem of model compression to reduce the computational burden on embedded system. Back to the date 2006, Buciluǎ et al. first proposed to train a neural network to mimic the output of a complex and large ensemble. This method uses ensemble to label the unlabeled data and trains the neural network with the data labeled by the ensemble, thus mimicking the function which learned by the ensemble and achieves similar accuracy. Based on the idea of (Buciluǎ,Geoffrey et al.) originally introduced a student-teacher paradigm in transferring the knowledge from a deeper and wider network (teacher) to a shallow network (student). They call this student-teacher paradigm as knowledge distillation (KD). By properly defining the knowledge of teacher as softened softmax (soft target), the student learns to mimic soft target distribution for each class. Thanks to Hinton's pioneer work, a series of subsequent works have sprung up by utilizing different forms of knowledge. Zagoruyko & Komodakis (2017) regard the spatial attention maps of a convolution neural network as network knowledge. However, an implicit assumption that they make is that the absolute value of a hidden neuron activation can be used as an indication about the importance of that neuron w.r.t. the specific input which limited their application only fit for image classification tasks. Another assumption that has been widely used is from Ba & Caruana (2014) that deeper networks always learn better representation. Based on that, FitNets (Romero et al. (2014)) tries to learn a thin deep network using a shallow one with more parameters. They believe that the convolution regressor is the network knowledge which can inherit from the teacher to its student. In 2017, researcher from TuSimple (TuSimple

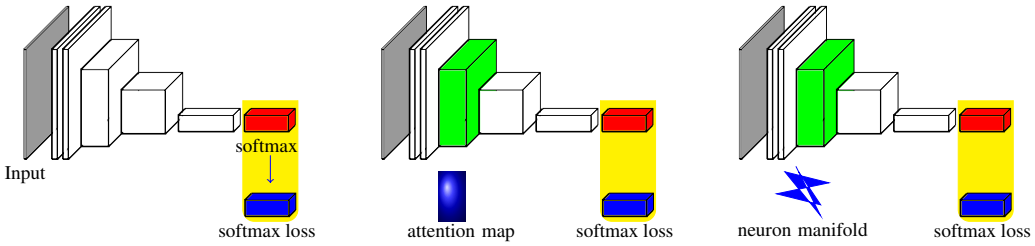

Figure 1: Three popular distillation methods. From left to right, traditional KD method in Geoffrey et al. (2015), using softmax as knowledge, the student network mimics teacher's softmax and minimize the loss on soft target. Middle one, mentioned in Zagoruyko & Komodakis (2017), named as attention transfer, an additional regularizer has been applied known as attention map, student needs to learn the attention map and soft target. The right part is our neuron manifold transfer, where we take neuron manifold as knowledge, and it reduces the computational and space cost.

(2015)) introduces two new definitions of network knowledge, Huang & Wang (2017) takes the advantage of Maximum Mean Discrepancy (MMD) to minimize the distance metric in probability distributions, and they regard network knowledge as class distribution. Chen et al. (2017) propose to transfer the cross sample similarities between the student and teacher to improve the performance of transferred networks. We notice that in the DarkRank the network knowledge is defined as cross sample similarities.

By reviewing extensive KD works, we notice that the key point in knowledge transfer is how we define the network knowledge, and in fact, a well-defined network knowledge can greatly improve the performance of the distilled network. Moreover, in our perspective, a perfect knowledge transfer method must allow us to transfer one neural network architecture into another, while preserving other generalization. A perfect transfer method, however, would use little observations to train, optimally use the limited samples at its disposal. Unfortunately, to our best knowledge, due to the complexity of large DNN, simply mimicking the teacher logit or a part of teacher features properties is far away to be benefited. Therefore, if we look back and consider the essence of DNN training, we notice that, another point of view to look at the distribution of neuron features is the shape of that feature. Here, the shape of neuron features include the actual value and the relative distance between two features. That is, during the process of knowledge transfer, student network not only learns the numerical information but also inherits the geometric properties. Therefore, in order to track the change of large DNN feature knowledge, a manifold approximation technique is vying for our attention. Manifold learning has been widely used in Topological Data Analysis (TDA) (Chazal & Michel (2017)), and this technique can project the high dimensional data to a lower dimensional manifold and preserving both numerical feature properties and geometric properties. In previous works, feature mapping causes computational resource waste, and class distribution matching is limited the usage. However, using the neuron manifold information, we not only collect as much as possible feature properties which can greatly represent feature, but also preserve inter-neuron characteristics (spatial relation). Since manifold projection can greatly reduce the dimension of teacher feature, we compress the teacher model and make student model more reliable. To summarize, the contributions of this paper are three folds:

- We introduce a new type of knowledge that is the d-dimensional smooth sub-manifold of teacher feature maps called neuron manifold.
- We formalize manifold space in feature map, and implement in details.
- We test our proposed method on various metric learning tasks. Our method can significantly improve the performance of student networks. And it can be applied jointly with existing methods for a better transferring performance.

We test our method on MNIST, CIFAR-10, and CIFAR-100 datasets and show that our Neuron Manifold Transfer (NMT) improves the students performance notably.

2 RELATED WORKS

In recent years, there are extensive works proposed to explore the knowledge transfer problems. As we mentioned the core idea behind knowledge transfer is properly defining the network knowledge and existing efforts can be broadly categorized into following 3 types, each of which is described below.

Soft target knowledge was first proposed by Geoffrey et al., because of their extraordinary pioneer work, network knowledge distillation (KD) becomes more and more popular and is being more practical. The trick in Hinton's work is that the network knowledge is defined as softened outputs of the teacher network, therefore, student network only need to mimic teacher's softened output to receive a good performance. The intuition behind is the one-hot labels (hard target) aim to project the samples in each class into one single point in the label space, while the softened labels project the samples into a continuous distribution. Inspired by Hinton's work, Sau & Balasubramanian (2016) use a perturbation logit to create a multiple teachers environment. By using noise-based regularizer, in their experiment, they show the reduction of intra-class variation. A proper noise level can help the student to achieve better performance. However, soft target knowledge's drawback is also very obvious: it only fits for some classification tasks and relies on the number of classes. For example, in a binary classification problem, KD could hardly improve the performance since almost no additional supervision could be provided.

Network feature knowledge is proposed to tackle the drawbacks of KD by transferring intermediate features. Romero et al. proposed FitNet to transfer a wide and shallow network to a thin and deep one. They think that deep convolution nets are significantly more accurate than shallow convolution models, when given the same parameter budget. In their adventurous work, FitNet makes the student mimic the full feature maps of the teacher. The computational cost, therefore, can not be ignored and such settings are too strict since the capacities of the teacher and student may differ greatly. In certain circumstances, FitNet may adversely affect the performance and convergence(Huang & Wang (2017)). Then Zagoruyko & Komodakis proposed Attention Transfer (AT) by define the network knowledge as spatial attention map of input images. To reduce the computational cost, they introduce an activation-based mapping function which compresses a 3D tensor to a 2D spatial attention map. Similarly, they make a questionable assumption that the absolute value of a hidden neuron activation can be used as an indication about the importance of that neuron. Another network feature knowledge is defined by Huang & Wang, instead of mimic softened output, Zehao aim to minimize the distribution of softened output via Maximum Mean Discrepancy method. A similar work applied in vision field is proposed by Yim et al. (2017), they call their knowledge as Flow of Solution Procedure (FSP) which computes the Gram matrix of features from two different layers. Network feature knowledge provides more supervision than simple KD method.

Jacobian knowledge is quite different from the above two classic type approaches. Soft target knowledge and network feature knowledge are defined as layer wise consideration, however, Jacobian knowledge generates the full picture of DNN and transfer the knowledge from function perspective. Sobolev training (Czarnecki et al. (2017)) proposed Jacobian-based regularizer on Sobolev spaces to supervise the higher order derivatives of teacher and student network. The subsequent work Srinivas & Fleuret (2018) deals with the problem of knowledge transfer using a first-order approximation of the neural network. Despite their novelty in knowledge transfer, Jacobian based knowledge is very hard in practical use because large DNNs are complex.

Although the above approaches show their potential power in knowledge distillation, we still think the idea of knowledge distillation should be revisited due to its complexity in both structure and computational property and the fact that deeper networks tend to be more non-linear.

3 PRELIMINARIES

In this section, we brief review the previous knowledge transfer methods. We also introduce the notations to be used in following sections. In practical, given a well defined deep neural network, for example, let us consider a Convolution Neural Network (CNN) and refer a teacher network as T and student network as S. Figure 1 illustrates three popular knowledge transfer methods and we explain in below.

3.1 KNOWLEDGE DISTILLATION

Assume we have a dataset of elements, with one such element denoted x, where each element has a corresponding one-hot class label: denote the one-hot vector corresponding to x by y. Given x, we have a trained teacher network $t = T(x)$ that outputs the corresponding logits, denoted by t; likewise we have a student network that outputs logits $s = S(x)$. To perform knowledge distillation we train the student network to minimize the following loss function (averaged across all data items):

$$\mathcal{L}_{KD} = (1-\alpha)\mathcal{L}_{CE}(p,q) + 2T^2\alpha\mathcal{L}_{CE}(\delta(\frac{t}{T}),\delta(\frac{s}{T})) \tag{1}$$

where $\delta(\cdot)$ is the softmax function, T is a temperature parameter and α is a parameter controlling the ratio of the two terms. The first term $L_{CE}(p,q)$ is a standard cross entropy loss penalizing the student network for incorrect classifications. The second term is minimized if the student network produces outputs similar to that of the teacher network. The idea is from that the outputs of the teacher network contain additional, beneficial information beyond just a class prediction.

3.2 ATTENTION TRANSFER

Consider a teacher network T has layers $i = 1, 2, \cdots, L$ and the corresponding layers in the student network. At each chosen layer i of the teacher network, we collect the spatial map of the activations for channel j into the vector a^t_{ij}. Let A^t_i collect a^t_{ij} for all j. Likewise for the student network we correspondingly collect into a^s_{ij} and A^s_i. Now given some choice of activation-based mapping function f(Ai) that maps each collection of the form A_i into a vector, attention transfer involves learning the student network by minimizing:

$$\mathcal{L}_{AT} = \mathcal{L}_{CE}(W,x) + \beta\sum_{i=1}^{N_L}||\frac{f(A^t_i)}{||f(A^t_1)||} - \frac{f(A^s_i)}{||f(A^t_1)||}||_2 \tag{2}$$

where β is a hyper-parameter. Zagoruyko & Komodakis (2017) recommended using $f(A^t_i) = \frac{1}{N_{A_i}}\sum_{j=1} a^2_{ij}$ where N_{A_i} is the number of channels at layer i. In other words, the loss targeted the difference in the spatial map of average squared activation, where each spatial map is normalized by the overall activation norm.

4 NEURON MANIFOLD TRANSFER

In this section, we will illustrate how to approximate neuron manifold from CNN features. Manifold approximation has been widely used to to avoid the curse of dimensionality, frequently encountered in Big Data analysis(Sober & Levin (2016)). There was a vast development in the field of linear and nonlinear dimension reduction. This techniques assume that the scattered input data is lying on a lower dimensional manifold, therefore, they aim to harvest this geometrical connection between the points, in order to reduce the effective number of parameters needed to be optimized(Sober et al. (2017)).

Determining the neuron manifold of a given feature is not a trivial task. As we mentioned in section 1, an efficient knowledge can greatly affect the performance of transfer learning. To our best knowledge, in the recent year, most of manifold approximation are learning based, which is not applied on our case due to high computational costs. Therefore, a simple but useful manifold approximation method is needed. Inspired by Sober et al. (2017), we can approximate the neuron manifold by using Moving Least Squares Projection(MLSP) mentioned in Sober & Levin (2016) with $O(n)$ run-time complexity.

In order to use MLSP, the given features should meet some criterion. Let assume the feature points $\{f_i\}^I_{i=1} \in \mathcal{F}$ are bounded, that is, there exist a distance h such that

$$h = \min_{f, f_i \in F} ||f - f_i|| \tag{3}$$

And we also assume the feature points are compact with density ρ. That is

$$size(\{F \cap \bar{B}(m, qh)\}) \leq \rho \cdot q^d, q \geq 1, m \in \mathcal{M}^d \tag{4}$$

where $\bar{B}(m, r)$ is a closed ball of radius r and centered at m such that $||f_i - f_j|| \leq h\sigma$ for $1 \leq i \leq j \leq I$ and $\sigma > 0$. Once we make the above assumption, according to Theorem 2.3 in Sober & Levin (2016), we can minimize the error bound of our approximation to $||\mathcal{M}^d - \mathbf{m}|| < k \cdot h^{m+1}$, where $\mathcal{M}^d$ is our approximated manifold, and $\mathbf{m}$ is ground truth sub-manifold of $\mathcal{R}^n$, k is some adjust factor.

Now we can approximate the neuron manifold by using Moving Least Squares Projection(MLSP). Let $\mathcal{M} \in \mathcal{R}^d$ be the neuron manifold we would like to find, and let $\{f_i\}_{i=1}^I$ be the feature points situated near $\mathcal{M}$. To find the neuron manifold of given feature, two following steps are required, first, we need to find a local d-dimensional affine space $H = H(f_i)$ as our local coordinate system (Algorithm 1 in Sober et al. (2017)), second, by utilizing the local coordinate defined by H, we project the feature points onto the coordinate system H and minimize the weighted least squares error to retrieve the target points as our neuron manifold features.

Determine local d-dimensional affine space, given the feature map of certain CNN layer, let us say $\mathcal{F}_n \in \mathcal{R}^{C \times W \times H}$ and all feature points $f_i \in F_n$, we would like to find a local d-dimensional affine space $H = H(f_i)$ with a point $q = q(f_i)$ on H, such that the following constrained problem is minimized:

$$\mathcal{L}(f, q, H) = \sum_{i=1}^{N} d(f_i, H)^2 \theta(||f_i - q||) \quad s.t. \quad r - q \perp H \tag{5}$$

where $d(f_i, H)$ is the Euclidean distance between the point f_i and the subspace H. We find the affine space H by an iterative procedure and we initialize the basis vectors of H_1 randomly. The reason we doing this is because the second term on right side of equation (6) $\theta(\cdot)$ is a weight function such that $\lim_{x \to \infty} \theta(x) \to 0$. That is when the feature f_i is far away to the affine space H, the influence of this feature to the H is less. Therefore, this local hyperplane H is passing through the features as much as possible.

Neuron manifold projection. Then we define the neuron manifold projection function as $p : R^f \to R^d$ such that $p_i = p(f_i) = f_i$. So that the approximation of p is performed by a weighted least squares vector valued polynomial function $m(x) = (m_1(f), \cdots m_n(f))^T$. Let x_i be the orthogonal projections of f_i onto $H(f_i)$. We formulated $m(x)$ as follow:

$$m(x) = argmin \sum_{i=1}^{N} (m_i(x_i) - p)^2 \theta(||f_i - q||) \tag{6}$$

$\theta(s)$ is a non-negative weight function (rapidly decreasing as $s \to 0$), and $|| \cdot ||$ is the Euclidean norm. Once we solve the above equation, we collect the projected point and mark it as our manifold feature point.

Neuron Manifold Transfer Given a output feature map of a layer in CNN by $\mathcal{F} \in \mathcal{R}^{C \times W \times H}$ which consists of C feature planes with spatial dimensions $H \times W$. And for each hyperplane feature $\mathcal{F}$, it has a sample set of a lower dimensional manifold $\mathcal{M}^d$ where d is the intrinsic dimension of $\mathcal{M}$ and $d \ll C \times W \times H$. Let $\mathcal{F}_T$ and $\mathcal{F}_S$ be the feature maps from certain layers of the teacher and student network, and $\mathcal{M}_T^{\mathcal{F}}$ and $\mathcal{M}_S^{\mathcal{F}}$ be lower dimensional manifold of teacher and student feature map respectively. Without loss of generality, we assume $\mathcal{F}_T$ and $\mathcal{F}_S$ have the same spatial dimensions. The feature maps can be interpolated if their dimensions do not match.

We can compute teacher network neuron manifold $\mathcal{M}_T^{\mathcal{F}}$ from feature dimension $\mathcal{F} \in \mathcal{R}^{C \times W \times H}$ by solving equation (5) and (6). Then, we train the student network parameters from some selected feature as well as the regressor parameters by minimizing the following loss function:

$$\mathcal{L}_{NMT} = \mathcal{L}_{CE}(W, x) + \frac{\lambda}{2} \mathbf{d}(\mathcal{M}_T^{\mathcal{F}}, \mathcal{M}_S^{\mathcal{F}}) \tag{7}$$

where $\mathbf{d}(m^1, m^2)$ is a manifold to manifold distance that is $\mathbf{d}(m_1, m_2) = \frac{1}{k} \sum^k (||m_i^1 - m_i^2||^p)$ where $m_i^1 \in m^1$ and $m_i^2 \in m^2$.

5 EXPERIMENTS

In the following section we explore neuron manifold transfer on various image classification datasets include MNIST LeCun & Cortes (1998), CIFAR-10 Krizhevsky et al. (2009), CIFAR-100 Krizhevsky et al. (2009) and ImageNet-ILSVRC-2013 Deng et al. (2014). On MNIST dataset,

KD Method	Top-1(%)	Top-5(%)	Top-1-C(%)	Top-5-C(%)
KD	98.5	99.99	96	99
FitNet	92.4	97.0	93	96
NST	97.3	99.7	93	96
NMT	98.6	99.99	99	99

Table 1: We validate different state-of-the-art knowledge transfer methods applied on MNIST. Top-1 indicate the **original** validation set. Top-5-C is the result of top 5 accuracy on our **Customized** validation set.

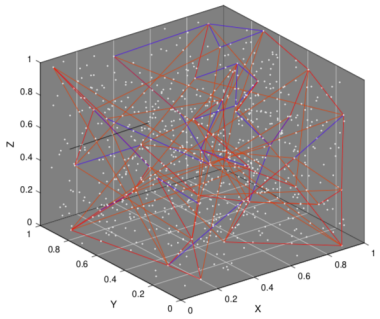

Figure 2: Neuron Manifold of Hinton-1200. White dots are feature points. Blue and red line form the neuron manifold.

we use a very classical CNN model mentioned in Geoffrey et al. (2015). It only has two hidden layers with 1200 rectified linear hidden units refer as Hinton-1200. We set this model as a pre-trained teacher and a smaller net that has same network architecture but only 800 rectified linear hidden units refer as Hinton-800 are used to be the student model. On CIFAR datasets, a middle level deep neuron network, ResNet-34 is used to be teacher, and as a result, we transfer the knowledge to a shallow and fast net known as ResNet-18. We also adopt a pre-activation version of VGG-19 with batch normalization from TorchVision TorchVision (2018) as teacher and create a modified version of AlexNet Krizhevsky et al. (2012) who has 8 hidden layers as student.

To further validate the effectiveness of our method, we compare our NMT with several state-of-the-art knowledge transfer methods, including traditional KD Geoffrey et al. (2015), attention transfer Zagoruyko & Komodakis (2017), HintNet Romero et al. (2014) and Neuron Selectivity Transfer Huang & Wang (2017). For KD, we set the temperature for softened softmax to 4 and $\alpha = 16$, following Geoffrey et al. (2015). For AT, the $\beta = 64$ and the spatial attention mapping function is defined as sum of absolute values. It is worth emphasizing that original AT is built on wide-residual-networks Zagoruyko & Komodakis (2016), therefore we modified the original settings of AT to achieve same results mentioned by Zagoruyko & Komodakis. As for our NMT, we set manifold approximation function's $\theta(s)$ as $\theta(s) = \frac{1}{s^2}$ and $\lambda = 22$ to achieve best performance. The number of sample points are various depend on the different network. We will make our implementation publicly available if the paper is accepted.

5.1 MNIST

We start our toy experiment on MNIST a handwritten digit recognition dataset with 10 classes (0-9) to evaluate our method. The training set contains 50000 images and validation set contains 10000 images. All samples are 28×28 in gray-scale images. In fact, Hinton-1200 has good performance on MNIST that we train it within 60 epochs and reach 98.6% accuracy on top-1 and 99.9% accuracy on top-5. Its student model Hinton-800 results show in Table 1. We collect 100 handwriting digit not included in original MNIST validation set. We can clearly see that NMT still has good performance. What need to be mentioned is that the AT can not be applied here, because AT is based on the attention map of input images, therefore, handwritten digit images with single channel leave zero information to their attention map.

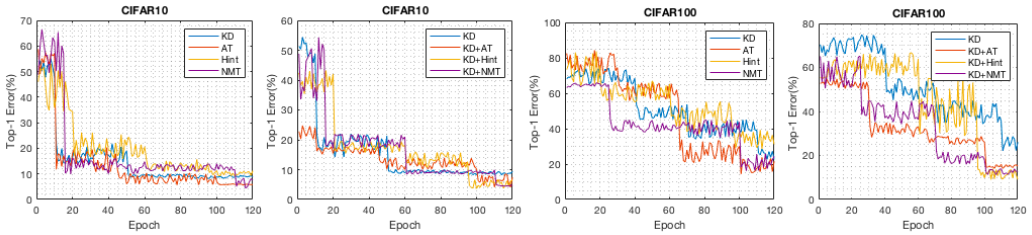

Figure 3: Different knowledge transfer methods on CIFAR10 and CIFAR100. Best view in color.

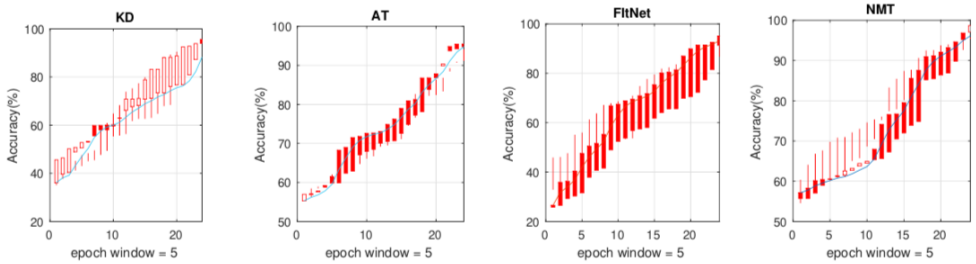

Figure 4: CIFAR10 Accuracy on validation set within 120 epochs, epoch window size = 5, total = 24 windows

To better understanding, we illustrate the Neuron Manifold Map as Figure 2. We extract out the fist layer of Hinton-1200, and normalize all value in between 0 to 1. We use Moving Least Squares method to approximate the true manifold of such layer. All white dots are feature points and form hyper-ball and the big black dots highlighted indicate the selected representative feature points which can best describe both feature properties and geometric properties(relative position and distance preserved).

5.2 CIFAR10 AND CIFAR 100

The results from experiments on CIFAR dataset is surprising. CIFAR-10 and CIFAR-100 datasets consist of 50K training images and 10K testing images with 10 and 100 classes, respectively. We take a 32×32 random crop from a zero-padded 40×40 image. For optimization, we use ADAM Kingma & Ba (2014) with a mini-batch size of 128 on a single GPU. We train the network in 120 epochs.

Figure 3 shows the training and testing curves of all the experiments on CIFR10 and CIFAR100. CIFAR10 contains 10 different categories, our NMT achieve most reliable classification result. Compared to other methods, even training epochs sufficiently large, the top-1 error not converge and being fluctuation. One possible reason is because AT and FitNet only transfer the knowledge of feature importance, however, NMT also focuses on the relation in between the neuron and NMT transfer the knowledge of inter-class relation. When training epochs increase, the neuron manifold changes slightly, and is more stable. Although classical KD has relative good performance, NMT can fast converge, which means using less epochs to have an accurate result. Two figures on right

KD Method	Top-1 Acc(%)	Top-5 Acc (%)	Best Top 1 Acc	Speed Up	Run Time
KD	22.34	28.12	36.78	x6.77	143
AT	31.41	42.2	55.4	x2.35	330
FitNet	33.11	30.2	48.2	0 (baseline)	452
NMT	35.56	44.92	59.7	x6.26	291
T-Model	VGG-19		S-Model	AlexNet-8-layers	

Table 2: VGG-19 to AlexNet, run time in second pre-epoch

KD Method	Kernel Size	Kernel Run Time	Utilization
AT	167MB	127s	0.82
FitNet	455MB	644s	0.34
NMT	104MB	269s	0.89

Table 3: Kernel Size Comparison for different knowledge transfer method

show the different transfer method performance on CIFAR-100. When class increases, from 10 to 100, our NMT show its preponderance. NMT can reach small error in first 20 epochs and improve the student performance into 10%. If we keep training to 200 epoch, NMT can have best result. Another advantage NMT has is that NMT do not rely on soft target. In Figure 3, we notice that large number of classes hurt the performance of KD. And we also mentioned that by using NMT the knowledge transferring time remarkably reduces. This is due to the computational cost of AT and FitNet are much higher than NMT.

Figure 4 shows the accuracy on validation set when epochs increase. One important fact that we can not neglect is that knowledge transfer aims to help the big model travel and deploy the small model on embedded system. We would like to reduce the time of knowledge transferring process and without accuracy loss. NMT has great work on training CIFAR10 and has best converge speed. During the early stage, say epochs between 0 to 40, the training result varies, but the result performance is above the average. We can clearly see the FitNet is under performance in full transfer period.

5.3 VGG-19 TO ALEXNET

VGG training is much more challenging. The standard VGG-19 is in linear structure, therefore, instead of transferring the neuron knowledge between each group in ResNet, we should be really careful to select the feature blocks for computing the neuron manifold. We optimize the network using adam with a mini-batch size of 128 on 2 GPUs. We train the network for 100 epochs. The initial learning rate is set to 0.1, and then divided by 10 at the 25, 50 and 75 epoch, respectively. Table 2 summaries the training result on CIFAR-100. In this section, we mainly focus on the result of system resource usage. FitNet would match all features between teacher and student, therefore, the matching time is the slowest and we set it as our baseline. Although there is an overhead due to computing neuron manifold, out NMT still has x6.26 speed up comparing to standard FitNet and AT. From the training perspective, although KD is the fast one with 143 second per epoch, it has the worst training result. And both At and FitNet are not as effective as transfer method. Table 3 summaries the different knowledge transfer methods kernel size. It is very clearly that FitNet failed in this task because FitNet is trying to match all features and as a consequence it has large kernel run time. Our NMT automatically chooses the the features to be transferred and result in an acceptable kernel run time. Compared with AT, computing the neuron manifold introduces overhead.

6 CONCLUSION

In this paper, we propose a novel method for knowledge transfer and we define a new type network knowledge named neuron manifold. By utilizing the state of art technique in Topological Data Analysis, we extract the DNN's feature properties and its geometric properties. We test our NMT on various dataset and the results are quite promising, thus further confirming that our knowledge transfer method could indeed learn better feature representations. They can be successfully transferred to high level vision task in the future. We believe that our novel view will facilitate the further design of knowledge transfer methods. In our future work, we plan to explore more applications of our NMT methods, especially in various regression problems, such as super resolution and optical flow prediction, etc.

ACKNOWLEDGMENT

REFERENCES

Lei Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pp. 2654–2662, Cambridge, MA, USA, 2014. MIT Press. URL http://dl.acm.org/citation.cfm?id=2969033.2969123.

Cristian Buciluǎ. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, 2006.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pp. 535–541, New York, NY, USA, 2006. ACM. ISBN 1-59593-339-5. doi: 10.1145/1150402.1150464. URL http://doi.acm.org/10.1145/1150402.1150464.

Frédéric Chazal and Bertrand Michel. An introduction to Topological Data Analysis: fundamental and practical aspects for data scientists. October 2017. URL https://hal.inria.fr/hal-01614384. working paper or preprint.

Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Darkrank: Accelerating deep metric learning via cross sample similarities transfer. *CoRR*, abs/1707.01220, 2017. URL http://arxiv.org/abs/1707.01220.

Wojciech Marian Czarnecki, Simon Osindero, Max Jaderberg, Grzegorz Swirszcz, and Razvan Pascanu. Sobolev training for neural networks. *CoRR*, abs/1706.04859, 2017. URL http://arxiv.org/abs/1706.04859.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2014.

Hinton Geoffrey, Vinyals Oriol, and Dean Jeffrey. Distilling the knowledge in a neural network. *arXiv preprint:1503.02531*, 2015.

Zehao Huang and Naiyan Wang. Like what you like: Knowledge distill via neuron selectivity transfer. *arXiv preprint arXiv:1707.01219*, 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL http://arxiv.org/abs/1412.6980.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The cifar-10 and cifar-100 dataset, 2009. URL https://www.cs.toronto.edu/~kriz/cifar.html.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pp. 1097–1105. Curran Associates, Inc., 2012. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

Yann LeCun and Corinna Cortes. The mnist database of handwritten digits, 1998. URL http://yann.lecun.com/exdb/mnist/.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *CoRR*, abs/1412.6550, 2014. URL http://arxiv.org/abs/1412.6550.

Bharat Bhusan Sau and Vineeth N. Balasubramanian. Deep model compression: Distilling knowledge from noisy teachers. *CoRR*, abs/1610.09650, 2016. URL http://arxiv.org/abs/1610.09650.

Barak Sober and David Levin. Manifolds' projective approximation using the moving least-squares (MMLS). *CoRR*, abs/1606.07104, 2016. URL http://arxiv.org/abs/1606.07104.

Barak Sober, Yariv Aizenbud, and David Levin. Approximation of functions over manifolds: A moving least-squares approach. *CoRR*, abs/1711.00765, 2017. URL http://arxiv.org/abs/1606.07104.

Suraj Srinivas and Francois Fleuret. Local affine approximations for improving knowledge transfer. Idiap-Com Idiap-Com-01-2018, Idiap, 3 2018. URL https://lld-workshop.github.io/papers/LLD_2017_paper_28.pdf.

TorchVision. Torchvision models, 2018. URL https://pytorch.org/docs/stable/torchvision/models.html.

TuSimple. Tusimple, 2015. URL http://www.tusimple.com/index-en.html.

J. Yim, D. Joo, J. Bae, and J. Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7130–7138, July 2017. doi: 10.1109/CVPR.2017.754.

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

Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017. URL https://arxiv.org/abs/1612.03928.