Like What You Like: Knowledge Distill via Neuron Selectivity Transfer

Zehao Huang Naiyan Wang
TuSimple
{zehaohuang18, winsty}@gmail.com

Abstract

Despite deep neural networks have demonstrated extraordinary power in various applications, their superior performances are at expense of high storage and computational costs. Consequently, the acceleration and compression of neural networks have attracted much attention recently. Knowledge Transfer (KT), which aims at training a smaller student network by transferring knowledge from a larger teacher model, is one of the popular solutions. In this paper, we propose a novel knowledge transfer method by treating it as a distribution matching problem. Particularly, we match the distributions of neuron selectivity patterns between teacher and student networks. To achieve this goal, we devise a new KT loss function by minimizing the Maximum Mean Discrepancy (MMD) metric between these distributions. Combined with the original loss function, our method can significantly improve the performance of student networks. We validate the effectiveness of our method across several datasets, and further combine it with other KT methods to explore the best possible results. Last but not least, we fine-tune the model to other tasks such as object detection. The results are also encouraging, which confirm the transferability of the learned features.

1. Introduction

In recent years, deep neural networks have renewed the state-of-the-art performance in various fields such as computer vision and neural language processing. Generally speaking, given enough data, deeper and wider networks would achieve better performances than the shallow ones. However, these larger and larger networks also bring in high computational and memory costs. It is still a great burden to deploy these state-of-the-art models into real-time applications.

This problem motivates the researches on acceleration and compression of neural networks. In the last few years, extensive work have been proposed in this field. These attempts can be roughly categorized into three types: network pruning [24, 14, 33, 18, 29, 25], network quantization [7, 34] and knowledge transfer (KT) [4, 19, 36, 38, 41, 30]. Network pruning iteratively prunes the neurons or weights of low importance based on certain criteria, while network quantization tries to reduce the precision of the weights or features. Nevertheless, it is worth noting that most of these approaches (except neuron pruning) are not able to fully exploit modern GPU and deep learning frameworks. Their accelerations need specific hardwares or implementations. In contrast, KT based methods directly train a smaller student network, which accelerates the original networks in terms of wall time without bells and whistles.

To the best of our knowledge, the earliest work of KT could be dated to [4]. They trained a compressed model with pseudo-data labeled by an ensemble of strong classifiers. However, their work is limited to shallow models. Until recently, Hinton *et al.* brought it back by introducing Knowledge Distillation (KD) [19]. The basic idea of KD is to distill knowledge from a large teacher model into a small one by learning the class distributions provided by the teacher via softened softmax. Despite its simplicity, KD demonstrates promising results in various image classification tasks. However, KD can only be applied in classification tasks with softmax loss function. Some subsequent works [36, 38, 41] tried to tackle this issue by transferring intermediate representations of teacher model.

In this work, we explore a new type of knowledge in teacher models, and transfer it to student models. Specifically, we make use of the selectivity knowledge of neurons. The intuition behind this model is rather straightforward: Each neuron essentially extracts a certain pattern related to the task at hand from raw input. Thus, if a neuron is activated in certain regions or samples, that implies these regions or samples share some common properties that may relate to the task. Such clustering knowledge is valuable for the student network since it provides an explanation to the final prediction of the teacher model. As a result, we propose to align the distribution of neuron selectivity pattern between student models and teacher models.

The illustration of our method for knowledge transfer is depicted in Fig. 1. The student network is trained to align the distribution of activations of its intermediate layer with

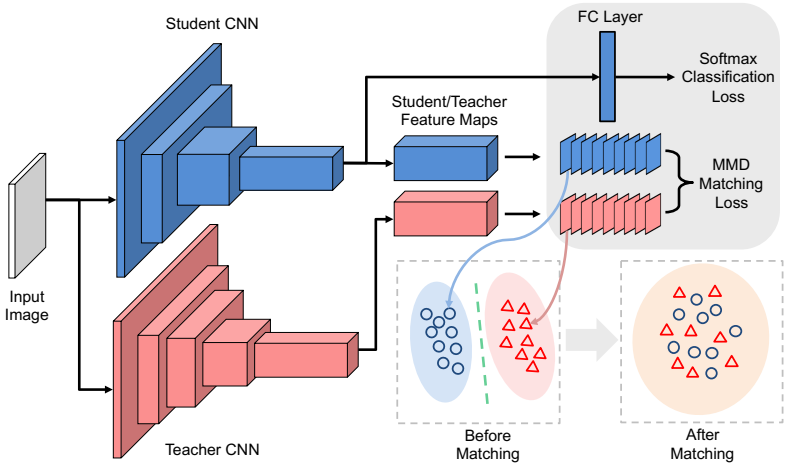


Figure 1. The architecture for our Neuron Selectivity Transfer: the student network is not only trained from ground-truth labels, but also mimics the distribution of the activations from intermediate layers in the teacher network. Each dot or triangle in the figure denotes its corresponding activation map of a filter.

that of the teacher. Maximum Mean Discrepancy (MMD) is used as the loss function to measure the discrepancy between teacher and student features. We test our method on CIFAR-10, CIFAR-100 and ImageNet datasets and show that our Neuron Selectivity Transfer (NST) improves the student's performance notably.

To summarize, the contributions of this work are as follows:

- We provide a novel view of knowledge transfer problem and propose a new method named Neuron Selectivity Transfer (NST) for network acceleration and compression.
- We test our method across several datasets and provide evidence that our Neuron Selectivity Transfer achieves higher performances than students significantly.
- We show that our proposed method can be combined with other knowledge transfer method to explore the best model acceleration and compression results.
- We demonstrate knowledge transfer help learn better features and other computer vision tasks such as object detection can benefit from it.

2. Related Works

Deep network compression and acceleration Many works have been proposed to reduce the model size and computation cost by network compression and acceleration. In the early development of neural network, network pruning [24, 15] was proposed to pursuit a balance between accuracy and storage. Recently, Han *et al.* brought it back to modern deep structures [14]. Their main idea is weights with small magnitude are unimportant and can be removed. However, this strategy only yields sparse weights and needs specific implementations for acceleration. To pursue efficient inference speed-up without dedicated libraries, researches on network pruning are undergoing a transition from connection pruning to filter pruning. Several works [33, 25] evaluate the importance of neurons by different selection criteria while others [32, 29, 42, 1, 18, 27] formulate pruning as a subset selection or sparse optimization problem. Beyond pruning, quantization [7, 34] and low-rank approximation [22, 8, 45] are also widely studied. Note that these acceleration methods are complementary to KT, which can be combined with our method for further improvement.

Knowledge transfer for deep learning Knowledge Distill (KD) [19] is the pioneering work to apply knowledge transfer to deep neural networks. In KD, the knowledge is defined as softened outputs of the teacher network. Compared with one-hot labels, softened outputs provide extra supervisions of intra-class and inter-class similarities learned by teacher. The one-hot labels 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. On one hand, softened labels could represent each sample by class distribution, thus captures intra-class variation; on the other hand, the inter-class similarities can be compared relatively among different classes in the soft target.

Formally, the soft target of a network T can be defined by $\boldsymbol{p}_T^\tau = \text{softmax}(\frac{\boldsymbol{a}_T}{\tau})$, where $\boldsymbol{a}$ is the vector of teacher logits (pre-softmax activations) and τ is a temperature. By increasing τ, such inter-class similarity is retained by driving the prediction away from 0 and 1. The student network is then trained by the combination of softened softmax and original softmax. However, its drawback is also obvious: Its

effectiveness only limits to softmax loss function, 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.

Subsequent works [36, 41, 38] tried to tackle the drawbacks of KD by transferring intermediate features. Lately, Romero *et al.* proposed FitNet [36] to compress networks from wide and shallow to thin and deep. In order to learn from the intermediate representations of teacher network, FitNet makes the student mimic the full feature maps of the teacher. However, such assumptions are too strict since the capacities of teacher and student may differ greatly. In certain circumstances, FitNet may adversely affect the performance and convergence. Recently, Zagoruyko *et al.* [38] proposed Attention Transfer (AT) to relax the assumption of FitNet: They transfer the attention maps which are summaries of the full activations. As discussed later, their work can be seen as a special case in our framework. Yim *et al.* [44] defined a novel type of knowledge, Flow of Solution Procedure (FSP) for knowledge transfer, which computes the Gram matrix of features from two different layers. They claimed that this FSP matrix could reflect the flow of how teachers solve a problem.

Domain adaptation belongs to the field of transfer learning [3]. In its mostly popular setting, the goal of domain adaptation is to improve the testing performance on an unlabeled target domain while the model is trained on a related yet different source domain. Since there is no labels available on the target domain, the core of domain adaptation is to measure and reduce the discrepancy between the distributions of these two domains. In the literature, Maximum Mean Discrepancy (MMD) is a widely used criterion, which compares distributions in the Reproducing Kernel Hilbert Space (RKHS) [12]. Several works have adopted MMD to solve the domain shift problem. In [20, 13, 10], examples in the source domain are re-weighted or selected so as to minimize the MMD between the source and target distributions. Other works like [2] measured MMD in an explicit low-dimensional latent space. As for applications in deep learning model, [28, 40] used MMD to regularize the learned features in source domain and target domain.

Note that, domain adaptation is not limited to the traditional supervised learning problem. For example, recently Li *et al.* casted neural style transfer [9] as a domain adaptation problem [26]. They demonstrated that neural style transfer is essentially equivalent to match the feature distributions of content image and style image. [9] is a special case with second order polynomial kernel MMD. In this paper, we explore the use of MMD for a novel application – knowledge transfer.

3. Background

In this section, we will start with the notations to be used in the sequel, then followed by a brief review of MMD which is at the core of our approach.

3.1. Notations

First, we assume the neural network to be compressed is a Convolutional Neural Network (CNN) and refer the teacher network as T and the student network as S. Let's denote the output feature map of a layer in CNN by $\mathbf{F} \in \mathbb{R}^{C\times HW}$ with C channels and spatial dimensions $H \times W$. For better illustration, we denote each row of $\mathbf{F}$ (i.e. feature map of each channel) as $\mathbf{f}^{k\cdot} \in \mathbb{R}^{HW}$ and each column of $\mathbf{F}$ (i.e. all activations in one position) as $\mathbf{f}^{\cdot k} \in \mathbb{R}^{C}$. Let $\mathbf{F}_T$ and $\mathbf{F}_S$ be the feature maps from certain layers of the teacher and student network, respectively. Without loss of generality, we assume $\mathbf{F}_T$ and $\mathbf{F}_S$ have the same spatial dimensions. The feature maps can be interpolated if their dimensions do not match.

3.2. Maximum Mean Discrepancy

In this subsection, we review the Maximum Mean Discrepancy (MMD), which can be regarded as a distance metric for probability distributions based on the data samples sampled from them [12]. Suppose we are given two sets of samples $\mathcal{X} = \{\boldsymbol{x}^i\}_{i=1}^N$ and $\mathcal{Y} = \{\boldsymbol{y}^j\}_{j=1}^M$ sampled from distributions p and q, respectively. Then the squared MMD distance between p and q can be formulated as:

$$\mathcal{L}_{\text{MMD}^2}(\mathcal{X}, \mathcal{Y}) = \| \frac{1}{N}\sum_{i=1}^{N}\phi(\boldsymbol{x}^i) - \frac{1}{M}\sum_{j=1}^{M}\phi(\boldsymbol{y}^j)\|_2^2, \quad (1)$$

where $\phi(\cdot)$ is a explicit mapping function. By further expanding it and applying the kernel trick, Eq. 1 can be reformulated as:

$$\begin{aligned}\mathcal{L}_{\text{MMD}^2}(\mathcal{X}, \mathcal{Y}) = &\frac{1}{N^2}\sum_{i=1}^{N}\sum_{i'=1}^{N}k(\boldsymbol{x}^i, \boldsymbol{x}^{i'}) \\ &+ \frac{1}{M^2}\sum_{j=1}^{M}\sum_{j'=1}^{M}k(\boldsymbol{y}^i, \boldsymbol{y}^{i'}) \\ &- \frac{2}{MN}\sum_{i=1}^{N}\sum_{j=1}^{M}k(\boldsymbol{x}^i, \boldsymbol{y}^j),\end{aligned} \quad (2)$$

where $k(\cdot, \cdot)$ is a kernel function which projects the sample vectors into a higher or infinite dimensional feature space.

Since the MMD loss is 0 if and only if $p = q$ when the feature space corresponds to a universal RKHS, minimizing MMD is equivalent to minimizing the distance between p and q [12].

(a) Monkey (b) Magnetic Hill

Figure 2. Neuron activation heat map of two selected images.

4. Neuron Selectivity Transfer

In this section, we present our Neuron Selectivity Transfer (NST) method. We will start with an intuitive example to explain our motivation, and then present the formal definition and some discussions about our proposed method.

4.1. Motivation

Fig. 2 shows two images blended with the heat map of one selected neuron in VGG16 Conv5_3. It is easy to see these two neurons have strong selectivities: The neuron in the left image is sensitive to monkey face, while the neuron in the right image activates on the characters strongly. Such activations actually imply the selectivities of neurons, namely what kind of inputs can fire the neuron. In other words, the regions with high activations from a neuron may share some task related similarities, even though these similarities may not intuitive for human interpretation. In order to capture these similarities, there should be also neurons mimic these activation patterns in student networks. These observations guide us to define a new type of knowledge in teacher networks: neuron selectivities or called co-activations, and then transfer it to student networks.

What is wrong with directly matching the feature maps? A natural question to ask is why cannot we align the feature maps of teachers and students directly? This is just what [36] did. Considering the activation of each spatial position as one feature, then the flattened activation map of each filter is an sample the space of neuron selectivities of dimension HW. This sample distribution reflects how a CNN interpret an input image: where does the CNN focus on? which type of activation pattern does the CNN emphasize more? As for distribution matching, it is not a good choice to directly match the samples from it, since it ignores the sample density in the space. Consequently, we resort to more advanced distribution alignment method as explained below.

4.2. Formulation

Following the notation in Sec. 3.1, each feature map $\mathbf{f}^{k\cdot}$ represents the selectivity knowledge of a specific neuron. Then we can define Neuron Selectivity Transfer loss as:

$$\mathcal{L}_{\text{NST}}(\mathbf{W}_S) = \mathcal{H}(\boldsymbol{y}_{\text{true}}, \boldsymbol{p}_S) + \frac{\lambda}{2}\mathcal{L}_{\text{MMD}^2}(\mathbf{F}_T, \mathbf{F}_S), \quad (3)$$

where $\mathcal{H}$ refers to the standard cross-entropy loss, and $\boldsymbol{y}_{\text{true}}$ represents true label and $\boldsymbol{p}_S$ is the output probability of the student network.

The MMD loss can be expanded as:

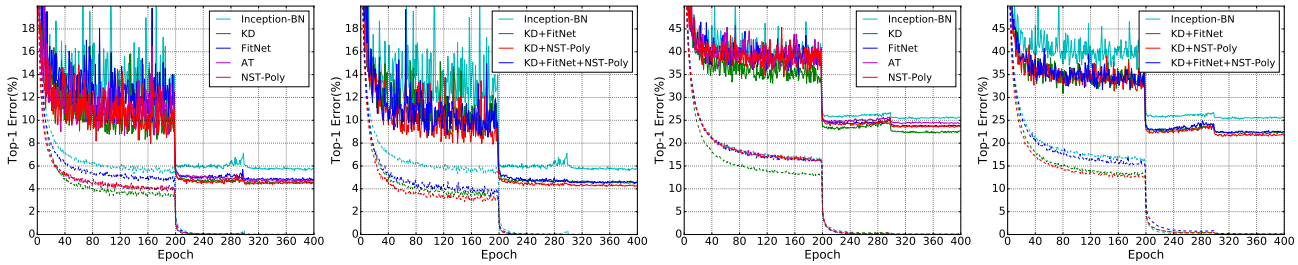
$$\begin{aligned}\mathcal{L}_{\text{MMD}^2}(\mathbf{F}_T, \mathbf{F}_S) = &\frac{1}{{C_T}^2}\sum_{i=1}^{C_T}\sum_{i'=1}^{C_T} k\left(\frac{\mathbf{f}_T^{i\cdot}}{\|\mathbf{f}_T^{i\cdot}\|_2}, \frac{\mathbf{f}_T^{i'\cdot}}{\|\mathbf{f}_T^{i'\cdot}\|_2}\right) \\ &+ \frac{1}{{C_S}^2}\sum_{j=1}^{C_S}\sum_{j'=1}^{C_S} k\left(\frac{\mathbf{f}_S^{j\cdot}}{\|\mathbf{f}_S^{j\cdot}\|_2}, \frac{\mathbf{f}_S^{j'\cdot}}{\|\mathbf{f}_S^{j'\cdot}\|_2}\right) \\ &- \frac{2}{C_T C_S}\sum_{i=1}^{C_T}\sum_{j=1}^{C_S} k\left(\frac{\mathbf{f}_T^{i\cdot}}{\|\mathbf{f}_T^{i\cdot}\|_2}, \frac{\mathbf{f}_S^{j\cdot}}{\|\mathbf{f}_S^{j\cdot}\|_2}\right).\end{aligned} \quad (4)$$

Note we replace $\mathbf{f}^{k\cdot}$ with its l_2-normalized version $\frac{\mathbf{f}^{k\cdot}}{\|\mathbf{f}^{k\cdot}\|_2}$ to ensure each sample has the same scale. Minimizing the MMD loss is equivalent to transferring neuron selectivity knowledge from teacher to student.

Choice of Kernels In this paper, we focus on the following three specific kernels for our NST method, including:

- Linear Kernel: $k(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^\top \boldsymbol{y}$
- Polynomial Kernel: $k(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x}^\top \boldsymbol{y} + c)^d$
- Gaussian Kernel: $k(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\frac{\|\boldsymbol{x}-\boldsymbol{y}\|_2^2}{2\sigma^2})$

For polynomial kernel, we set $d = 2$, and $c = 0$. For Gaussian kernel, the σ^2 is set as the mean of squared distance of the pairs.



(a) CIFAR10 (b) CIFAR10 (c) CIFAR100 (d) CIFAR100

Figure 3. Different knowledge transfer methods on CIFAR10 and CIFAR100. Test errors are in bold, while train errors are in dashed lines. Our NST improves final accuracy observably with a fast convergence speed. Best view in color.

4.3. Discussion

In this subsection, we discuss NST with linear and polynomial kernel in detail. Specifically, we show the intuitive explanations behind the math and their relationships with existing methods.

4.3.1 Linear Kernel

In the case of linear kernel, Eq. 4 can be reformulated as:

$$\mathcal{L}_{\text{MMD}^2_L}(\mathbf{F}_T, \mathbf{F}_S) = \| \frac{1}{C_T} \sum_{i=1}^{C_T} \frac{\mathbf{f}_T^{i\cdot}}{\|\mathbf{f}_T^{i\cdot}\|_2} - \frac{1}{C_S} \sum_{j=1}^{C_S} \frac{\mathbf{f}_S^{j\cdot}}{\|\mathbf{f}_S^{j\cdot}\|_2} \|_2^2. \tag{5}$$

Interestingly, we find the activation-based Attention Transfer (AT) in [38] define their transfer loss as:

$$\mathcal{L}_{\text{AT}}(\mathbf{F}_T, \mathbf{F}_S) = \|A(\mathbf{F}_T) - A(\mathbf{F}_S)\|_2^2, \tag{6}$$

where $A(\mathbf{F})$ is an attention mapping. Specifically, one of the attention mapping function in [38] is the normalized sum of absolute values mapping, which is defined as:

$$A_{\text{abssum}}(\mathbf{F}) = \frac{\sum_{k=1}^{C} |\mathbf{f}^{k\cdot}|}{\| \sum_{k=1}^{C} |\mathbf{f}^{k\cdot}| \|_2}, \tag{7}$$

and the loss function of AT can be reformulated as:

$$\mathcal{L}_{\text{AT}}(\mathbf{F}_T, \mathbf{F}_S) = \| \frac{\sum_{i=1}^{C_T} |\mathbf{f}_T^{i\cdot}|}{\| \sum_{i=1}^{C_T} |\mathbf{f}_T^{i\cdot}| \|_2} - \frac{\sum_{j=1}^{C_S} |\mathbf{f}_S^{j\cdot}|}{\| \sum_{j=1}^{C_S} |\mathbf{f}_S^{j\cdot}| \|_2} \|_2^2. \tag{8}$$

For the activation maps after ReLU layer, which are already non-negative, Eq. 5 is equivalent to Eq. 8 except the form of normalization. They both represent where the neurons have high responses, namely the "attention" of the teacher network. Thus, [38] is a special case in our framework.

4.3.2 Polynomial Kernel

Slightly modifying the explanation of second order polynomial kernel MMD matching in [26], NST with second order polynomial kernel with $c = 0$ can be treated as matching the Gram matrix of two vectorized feature maps:

$$\mathcal{L}_{\text{MMD}^2_P}(\mathbf{F}_T, \mathbf{F}_S) = \|\mathbf{G}_S - \mathbf{G}_T\|_F^2, \tag{9}$$

where $\mathbf{G} \in \mathbb{R}^{HW \times HW}$ is the Gram matrix, with each item g_{ij} as:

$$g_{ij} = (\mathbf{f}^{\cdot i})^T \mathbf{f}^{\cdot j}, \tag{10}$$

where each item g_{ij} in the Gram matrix roughly represents the similarity of region i and j (For simplicity, the feature maps $\mathbf{F}_T$ and $\mathbf{F}_S$ are normalized as we mentioned in Sec. 4.2). It guides the student network to learn better internal representation by explaining such task driven region similarities in the embedding space. It greatly enriches the supervision signal for student networks.

5. Experiments

In the following sections, we evaluate our NST on several standard datasets, including CIFAR-10, CIFAR-100 [23] and ImageNet LSVRC 2012 [37]. On CIFAR datasets, an extremely deep network, ResNet-1001 [17] is used as teacher model, and a simplified version of Inception-BN [21][1] is adopted as student model. On ImageNet LSVRC 2012, we adopt a pre-activation version of ResNet-101 [17] and original Inception-BN [21] as teacher model and student model, respectively.

To further validate the effectiveness of our method, we compare our NST with several state-of-the-art knowledge transfer methods, including KD [19], FitNet [36] and AT [38]. For KD, we set the temperature for softened softmax to 4 and $\lambda = 16$, following [19]. For FitNet and AT, the value of λ is set to 10^2 and 10^3 following [38]. The mapping function of AT adopted in our reimplementation is square sum, which performs best in the experiments of [38]. As for our NST, we set $\lambda = 5 \times 10^1, 5 \times 10^1$ and 10^2 for linear, polynomial and Gaussian kernel, respectively. All the experiments are conducted in MXNet [5]. We will make our implementation publicly available if the paper is accepted.

[1] https://tinyurl.com/inception-bn-small

Method	Model	CIFAR-10	CIFAR-100
Student	Inception-BN	5.80	25.63
KD [19]	Inception-BN	4.47	**22.18**
FitNet [36]	Inception-BN	4.75	23.48
AT [38]	Inception-BN	4.64	24.31
NST (linear)	Inception-BN	4.87	24.28
NST (poly)	Inception-BN	**4.39**	23.46
NST (Gaussian)	Inception-BN	4.48	23.85
Teacher	ResNet-1001	4.04	20.50

Table 1. CIFAR results of individual transfer methods.

Method	Model	CIFAR-10	CIFAR-100
KD+FitNet	Inception-BN	4.54	22.29
KD+NST*	Inception-BN	**4.21**	**21.48**
KD+FitNet+NST*	Inception-BN	4.54	22.25

Table 2. CIFAR results of combined transfer methods. NST* represents NST with polynomial kernel.

5.1. CIFAR

We start with the CIFAR dataset to evaluate our method. CIFAR-10 and CIFAR-100 datasets consist of 50K training images and 10K testing images with 10 and 100 classes, respectively. For data augmentation, we take a 32×32 random crop from a zero-padded 40×40 image or its flipping following [16]. The weight decay is set to 10^{-4}. For optimization, we use SGD with a mini-batch size of 128 on a single GPU. We train the network 400 epochs. The learning rate starts from 0.2 and is divided by 10 at 200 and 300 epochs.

For AT, FitNet and our NST, we add a single transfer loss between the convolutional layer output of "in5b" in Inception-BN and the output of last group residual block in ResNet-1001. We also try to add multiple transfer losses in different layers and find that the improvement over single loss is minor for these methods.

Table 1 summarizes our experiment results. Our NST achieves higher accuracy than the original student network, which demonstrates the effectiveness of feature maps distribution matching. The choice of kernel influences the performance of NST. In our experiments, polynomial kernel yields better result than both linear and Gaussian kernels. Comparing with other knowledge transfer methods, our NST is also competitive. In CIFAR-10, all these transfer methods achieve higher accuracy than the original student network. Among them, our NST with polynomial kernel performs the best. In CIFAR-100, KD achieves the best performance. This is consistent with our explanation that KD would perform better in the classification task with more classes since more classes provide more accurate information about intra-class variation in the softened softmax target.

We also try to combine different transfer methods to explore the best possible results. Table 2 shows the results of KD+FitNet, KD+NST and KD+FitNet+NST. Not surprisingly, matching both final predictions and intermediate representations improve over individual transfers. Particularly, KD combined with our NST performs best in these three settings. To be specific, we improve the performance of student network by about 1.6% and 4.2% absolutely, and reduce the relative error by **27.6%** and **16.4%**, respectively. The training and testing curves of all the experiments can be found in Fig. 3. All the transfer methods converge faster than student network. Among them, KD+NST is the fastest.

5.2. ImageNet LSVRC 2012

In this section, we conduct large-scale experiments on the ImageNet LSVRC 2012 classification task. The dataset consists of 1.28M training images and another 50K validation images. We optimize the network using Nesterov Accelerated Gradient (NAG) with a mini-batch size of 512 on 8 GPUs (64 per GPU). The weight decay is 10^{-4} and the momentum is 0.9 for NAG. For data augmentation and weight initialization, we follow the publicly available implementation of "fb.resnet" [2]. We train the network for 100 epochs. The initial learning rate is set to 0.1, and then divided by 10 at the 30, 60 and 90 epoch, respectively. We report both top-1 and top-5 validation errors on the standard single test center-crop setting. According to previous section, we only evaluate the best setting in our method – NST with second order polynomial kernel. The value of λ is set to 5×10^1. Other settings are the same as CIFAR experiments. All the results of our ImageNet experiments can be found in Table 3 and Fig. 4. Our method achieves 0.9% top-1 and 0.5% top-5 improvements compared with the student network. Interestingly, different from [38], we also

[2] https://github.com/facebook/fb.resnet.torch

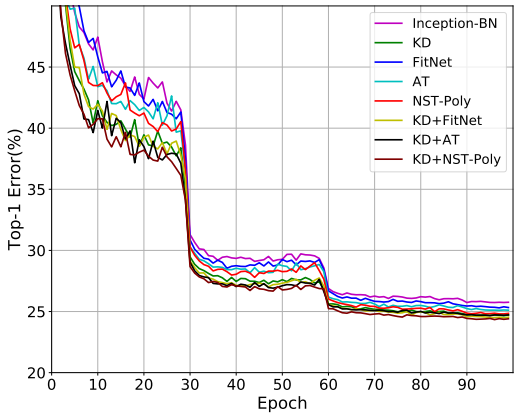


Figure 4. Top-1 validation error of different knowledge transfer methods on ImageNet. Best view in color.

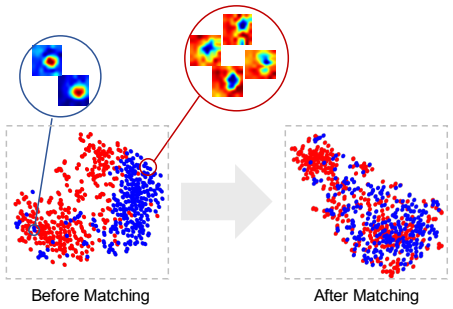


Figure 5. t-SNE [31] visualization shows that our NST Transfer reduces the distance between teacher and student activations distribution.

Method	Model	Top-1	Top-5
Student	Inception-BN	25.74	8.07
KD [19]	Inception-BN	24.56	7.35
FitNet [36]	Inception-BN	25.30	7.93
AT [38]	Inception-BN	25.10	7.61
NST*	Inception-BN	24.82	7.58
KD+FitNet	Inception-BN	24.48	7.27
KD+AT	Inception-BN	24.64	7.26
KD+NST*	Inception-BN	**24.34**	**7.11**
Teacher	ResNet-101	22.68	6.58

Table 3. ImageNet validation error (single crop) of multiple transfer methods. NST* represents NST with polynomial kernel.

find that in our experiments both KD and FitNet improve the convergence and accuracy of Inception-BN. This may be caused by the choice of weak teacher network (ResNet-34) in [38]. Among all the methods, KD performs the best. When combined with KD, our NST achieves the best accuracy, which improves top-1 and top-5 accuracy by 1.4% and 1%, respectively. These results further verify the effectiveness of our proposed NST in large scale application and its complementarity with other state-of-the-art knowledge transfer methods.

5.3. PASCAL VOC 2007 Detection

"Network engineering" plays an increasingly important role in visual recognition. Researchers focus on designing better network architectures to learn better representations. Several works have demonstrated that the improvement of feature learning in image classification could be successfully transferred to other recognition tasks [16, 43, 6], such as object detection and semantic segmentation. *However, can the gain from knowledge transfer in image classification task be transferred to other high level vision tasks?* We provide a preliminary investigation in object detection task.

Our evaluation is based on the Faster-RCNN [35] system on PASCAL VOC 2007 dataset. Following the settings in [35], we train the models on the union set of VOC 2007 *trainval* and VOC 2012 *trainval*, and evaluate them on the test set. Since our goal is to validate the effectiveness of base models, we make comparisons by only varying the pre-trained ImageNet classification models, while keeping other parts unchanged. The backbone network is Inception-BN with different KT methods. We extract features from the "4b" layer whose stride is 16 pixels. Standard evaluation metrics Average Precision (AP) and mean AP (mAP) are reported for evaluation.

Table 4 summarizes the detection results. All the models with KT achieve higher mAP than the baseline. Comparing with other transfer techniques, our NST improves most with 1.2 higher mAP. Combined with KD, the KD+NST yields 1.6 gain. These results demonstrate that KT could benefit object detection task without any modifications and extra computations to the original student model in testing. Consequently, they are powerful tools to improve performance in a wide range of applications for practitioners. Interestingly, though KD performs best in large-scale image classification task, feature map based mimicking methods, including FitNet, AT and our NST have greater advantages over it in object detection task. We owe it to the importance of spatial information in object detection. KD totally ignores it while other methods exploit it in certain extent.

6. Discussion

In this section, we first analyze the strengths and weaknesses of several closely related works based on the results from our experiment, and then discuss some possible extenstions of the proposed NST method.

6.1. Analysis of Different Transfer Methods

In Fig. 5, we visualize the distributions of student and teacher networks' activations before and after our NST

Method	mAP	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Baseline	75.6	**77.2**	79.2	75.0	62.2	61.7	83.7	84.9	87.0	60.5	81.6	66.9	**86.5**	**86.9**	78.7	78.8	49.2	**78.2**	78.2	81.7	74.9
KD	76.0	75.7	79.4	75.7	66.1	60.1	85.4	84.3	86.8	60.1	**84.0**	65.7	84.4	87.0	81.6	79.0	48.8	76.8	**79.5**	83.4	75.4
FitNet	76.6	75.5	82.8	77.7	67.2	58.7	84.8	**85.9**	86.7	**61.1**	81.6	70.1	85.3	86.0	81.4	79.0	51.8	78.1	78.8	**85.2**	74.1
AT	76.5	74.9	83.0	**77.8**	67.3	61.7	85.2	85.2	87.4	60.5	83.0	69.4	85.0	86.7	81.8	79.1	49.0	78.1	78.6	82.5	74.4
NST*	76.8	75.7	81.5	75.4	67.3	61.1	86.1	85.0	86.9	61.0	82.7	**71.5**	**86.5**	86.8	**84.3**	79.0	51.5	77.4	77.6	84.4	75.2
KD+NST*	**77.2**	75.7	**84.2**	77.2	**67.6**	**63.5**	**86.4**	85.7	**88.7**	61.0	83.1	69.7	85.4	85.2	83.8	**79.2**	**51.9**	76.0	78.4	82.9	**77.1**

Table 4. Detection results on the PASCAL VOC 2007 test set. The baseline is the standard Faster R-CNN system with Inception-BN model.

transfer in the CIFAR100 experiment using [31]. Each dot in the figure denotes an activation pattern of a neuron. As expected, MMD loss significantly reduces the discrepancy between teacher and student distributions, which makes the student network act more like the teacher network.

KD [19] achieves its best performance when there are a large number of classes. In that case, softened softmax can depict each data sample in the embedded label space more accurate than the case that the number of class is small. However, the drawback of KD is that it is fully based on softmax loss, which limits its applications in broader applications such as regression and ranking. Other compared methods do not have to meet such constraints.

As for FitNet [36], we find that its assumption is too strict in the sense that it forces the student network to match the full activations of teacher model as mentioned before. As we discussed in 4.1, directly matching samples ignores the density in the space. In certain circumstances, the training of FitNet will be influenced by noise seriously, which makes it hard to converge.

6.2. Beyond Maximum Mean Discrepancy

We propose a novel view of knowledge transfer by treating it as a distribution matching problem. Although we select MMD as our distribution matching method, other matching methods can also be incorporated into our framework. If we can formulate the distribution into a parametric form, simple moment matching can be used to align distribution. For more complex cases, drawing the idea of Generative Adversarial Network (GAN) [11] to solve this problem is an interesting direction to pursue. The goal of GAN is to train a generator network G that generates samples from a specific data distribution. During the training, a discriminator network D is used to distinguish that whether a sample comes from the real data or generated by G. In our framework, the student network can be seen as a generator. D is trained to distinguish whether features are generated by the student network or teacher. if G sucessfully confuses D, then the domain discrepancy is minimized. Similar ideas have already been exploited in domain adaptation area [39], we believe it can also be used in our application.

7. Conclusions

In this paper, we propose a novel method for knowledge transfer by casting it as a distribution alignment problem. We utilize an unexplored type of knowledge – neuron selectivity. It represents the task related preference of each neuron in the CNN. In detail, we match the distributions of spatial neuron activations between teacher and student networks by minimizing the MMD distance between them. Through this technique, we successfully improve the performance of small student networks. In our experiments, we show the effectiveness of our NST method on various datasets, and demonstrate that NST is complementary to other existing methods: Specifically, further combination of them yields the new state-of-the-art results. Furthermore, we analyze the generalizability of knowledge transfer methods to other tasks. The results are quite promising, thus further confirm that knowledge transfer methods could indeed learn better feature representations. They can be successfully transferred to other high level vision tasks, such as object detection task.

We believe our novel view will facilitate the further design of knowledge transfer methods. In our future work, we plan to explore more applications of our NST methods, especially in various regression problems, such as super-resolution and optical flow prediction, etc.

References

[1] J. M. Alvarez and M. Salzmann. Learning the number of neurons in deep networks. In *NIPS*, 2016. 2

[2] M. Baktashmotlagh, M. T. Harandi, B. C. Lovell, and M. Salzmann. Unsupervised domain adaptation by domain invariant projection. In *ICCV*, 2013. 3

[3] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010. 3

[4] C. Bucila, R. Caruana, and A. Niculescu-Mizil. Model compression. In *KDD*, 2006. 1

[5] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang. MXNet: A flexible and efficient machine learning library for heterogeneous distributed systems. In *NIPS Workshop*, 2015. 5

[6] Y. Chen, J. Li, H. Xiao, X. Jin, S. Yan, and J. Feng. Dual path networks. In *NIPS*, 2017. 7

[7] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or -1. In *NIPS*, 2016. 1, 2

[8] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In *NIPS*, 2014. 2

[9] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *CVPR*, 2016. 3

[10] B. Gong, K. Grauman, and F. Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML*, 2013. 3

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014. 8

[12] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(3):723–773, 2012. 3

[13] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf. Covariate shift by kernel mean matching. *Dataset Shift in Machine Learning*, 3(4):5, 2009. 3

[14] S. Han, J. Pool, J. Tran, and W. Dally. Learning both weights and connections for efficient neural network. In *NIPS*, 2015. 1, 2

[15] B. Hassibi and D. G. Stork. Second order derivatives for network pruning: Optimal brain surgeon. In *NIPS*, 1993. 2

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6, 7

[17] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *ECCV*, 2016. 5

[18] Y. He, X. Zhang, and J. Sun. Channel pruning for accelerating very deep neural networks. In *ICCV*, 2017. 1, 2

[19] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In *NIPS Workshop*, 2014. 1, 2, 5, 6, 7, 8

[20] J. Huang, A. Gretton, K. M. Borgwardt, B. Schölkopf, and A. J. Smola. Correcting sample selection bias by unlabeled data. In *NIPS*, 2007. 3

[21] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 5

[22] M. Jaderberg, A. Vedaldi, and A. Zisserman. Speeding up convolutional neural networks with low rank expansions. In *BMVC*, 2014. 2

[23] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Tech Report*, 2009. 5

[24] Y. LeCun, J. S. Denker, S. A. Solla, R. E. Howard, and L. D. Jackel. Optimal brain damage. In *NIPS*, 1990. 1, 2

[25] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf. Pruning filters for efficient ConvNets. In *ICLR*, 2017. 1, 2

[26] Y. Li, N. Wang, J. Liu, and X. Hou. Demystifying neural style transfer. In *IJCAI*, 2017. 3, 5

[27] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang. Learning efficient convolutional networks through network slimming. In *ICCV*, 2017. 2

[28] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015. 3

[29] J.-H. Luo, J. Wu, and W. Lin. ThiNet: A filter level pruning method for deep neural network compression. In *ICCV*, 2017. 1, 2

[30] P. Luo, Z. Zhu, Z. Liu, X. Wang, and X. Tang. Face model compression by distilling knowledge from neurons. In *AAAI*, 2016. 1

[31] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11):2579–2605, 2008. 7, 8

[32] Z. Mariet and S. Sra. Diversity networks. In *ICLR*, 2016. 2

[33] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz. Pruning convolutional neural networks for resource efficient inference. In *ICLR*, 2017. 1, 2

[34] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. XNOR-Net: ImageNet classification using binary convolutional neural networks. In *ECCV*, 2016. 1, 2

[35] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 7

[36] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. FitNets: Hints for thin deep nets. In *ICLR*, 2015. 1, 3, 4, 5, 6, 7, 8

[37] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, and A. C. Berg. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 5

[38] Z. Sergey and K. Nikos. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017. 1, 3, 5, 6, 7

[39] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *CVPR*, 2017. 8

[40] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv:1412.3474*, 2014. 3

[41] Z. Wang, Z. Deng, and S. Wang. Accelerating convolutional neural networks with dominant convolutional kernel and knowledge pre-regression. In *ECCV*, 2016. 1, 3

[42] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li. Learning structured sparsity in deep neural networks. In *NIPS*, 2016. 2

[43] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 7

[44] J. Yim, D. Joo, J. Bae, and J. Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *CVPR*, 2017. 3

[45] X. Zhang, J. Zou, X. Ming, K. He, and J. Sun. Efficient and accurate approximations of nonlinear convolutional networks. In *CVPR*, 2015. 2