A SCALABLE CONVOLUTIONAL NEURAL NETWORK FOR TASK-SPECIFIED SCENARIOS VIA KNOWLEDGE DISTILLATION

Mengnan Shi[†] *Fei Qin*[†⋆] *Qixiang Ye*[†] *Zhenjun Han*[†] *Jianbin Jiao*[†]

[†]School of Electronics, Electrical and Communication Engineering
University of Chinese Academy of Sciences, Beijing, China.
[⋆]fqin1982@ucas.ac.cn

ABSTRACT

In this paper, we explore the redundancy in convolutional neural network, which scales with the complexity of vision tasks. Considering that many front-end visual systems are interested in only a limited range of visual targets, the removing of task-specified network redundancy can promote a wide range of potential applications. We propose a task-specified knowledge distillation algorithm to derive a simplified model with pre-set computation cost and minimized accuracy loss, which suits the resource constraint front-end systems well. Experiments on the MNIST and CIFAR10 datasets demonstrate the feasibility of the proposed approach as well as the existence of task-specified redundancy.

***Index Terms*—** convolutional neural networks, knowledge distillation, model compression, task-specified

1. INTRODUCTION

Visual algorithms in front-end systems play an important role to boost related industry, e.g., augmented reality systems in daily lives, vehicle recognition systems in transportation systems, and unmanned aerial vehicle (UAV) surveillance systems for security purpose. It is generally expected that the implementation of such systems will bring modern society a more convenient life.

This brilliant future has been guaranteed by the recent popular deep learning framework in computer vision community, represented by convolutional neural networks (CNN), with their significant performance, which also play as the chevaux-de-frise role for the front-end implementation. This is due to the fact that most deep algorithms are only optimized for the accuracy performance. As a result, these algorithms require thousands of GPU cores to process an image in real-time, and even more GPU cores to train a model in days or weeks. Although training cost can be ignored for front-end systems, the computation cost for the test stage is still too high to be afforded by most embedded systems without powerful GPUs. This is why the currently famous alpha-Go was really a huge chassis deployed in UK rather than in the game site.

While some researchers are working with the deeper and deeper fashion to keep improving the accuracy of deep learning framework [1–3], some others have noticed the redundancy of deep framework, i.e., most network components negligibly contribute to the overall performance. Several state-of-the-art works [4–15] have shown significant potential performance to derive a simplified CNN without accuracy loss in various usages. The works of [5, 8, 10, 16, 17] follow the pruning style, which propose different metrics to evaluate the contribution of network components. According to these contribution metrics, low response components, named the shared redundancy, will be disabled. Some other researchers [6, 7, 9, 18] propose to use knowledge distillation, which employs the acquired knowledge of the original model to train another, usually smaller, model. The smaller model, named student model, will mimic the original model, named teacher model, with comparable performance but higher efficiency.

Beyond these works, most front-end deployed visual algorithms are task-specified. For example, an UAV based surveillance system only focuses on the military targets and won't be interested in a cat or cup. But most popular convolutional models are learned from big data with thousands of object classes, the knowledge among which is more than necessary for these scenarios. Existing work [19] has demonstrated that the filters in a network have different responses to different targets. It is straightforward to make the hypothesis that the redundancy of CNN scales with the complexity of a given task, e.g., if the categories of interested targets significantly decreases, there will be corresponding redundancy, named task-specified redundancy, in the network, which can be further removed.

We propose a task-specified knowledge distillation algorithm to derive a simplified model. The knowledge distillation method relies on transferring the learned discriminative infor-

This research is supported in partial by Nature Science Foundation of China under Grant No.61401426, 61271433, 61671427, Beijing Municipal Science and Technology Commission under Grant No.Z161100001616005, and Science and Technology Innovation Foundation of Chinese Academy of Sciences under Grant No.CXJJ-16Q218.

mation from a teacher model to a student model. We first analyze and demonstrate the redundancy of the neural network related to *a priori* complexity of the given task. We then train a student model by re-defining the loss function from a subset of the relaxed target knowledge according to the task information. The new model can satisfy the constraints of both the computation cost and residue accuracy.

The organization of this paper is as follows: section 2 provides the problem description, proposed methodology is discussed in section 3, and section4 describes the experiment design and results analyses. The conclusion of this work as well as potential future work is discussed in section 5.

2. PROBLEM DESCRIPTION

A typical CNN in vision applications takes an image as input and processes it into a feature vector for image classification, scene classification, object detection, and object tracking, etc. It is worth noting that the core aim of CNN is to learn features, the capability of which is usually hard to be quantitatively determined. A plausible method is to employ the accuracy of classification task as an instead metric. This is reasonable, since the usage of classification is embedded in most visual algorithms as an essential tool. The accuracy of classification can be treated as a rough metric to express the capability of feature representation [19, 20].

As shown in fig.1, the input of a convolutional layer is a set of feature maps. The size of input of a convolutional layer is $C_i * I_s$, where C_i is the number of input channels, and I_s specifies the size of input feature maps. The output of the convolutional layer is $C_o * O_s$ feature maps, where C_o is the number of output channels and O_s the size of output feature maps. The process is implemented by convolutional operation with $C_o * C_i$ filters of size K_s. If the size of input and other convolutional parameters have been set, the computation complexity of a convolutional layer can be expressed as:

$$CP_{conv} \sim O(\textstyle\sum_{conv_{layer}}(C_i * C_o * K_s)) \tag{1}$$

The situation of the full connected layers will be slightly different, but still scaled with N_i and N_o, i.e., the number of input neurons and output neurons:

$$CP_{fc} \sim O(\textstyle\sum_{fc_{layer}}(N_i * N_o)) \tag{2}$$

This architecture leads to extremely high computation cost, say in billions of instructions in processor. Processing a standard size image in the popular AlexNet requires 1.5 billion floating point operations [5]. As reported [5, 8, 10], network components in a CNN model like connections, filters, and channels, may have different contributions to the overall feature representation. Remaining those with high contribution and pruning low contribution ones will only bring tiny influences to the overall accuracy. It is assured that

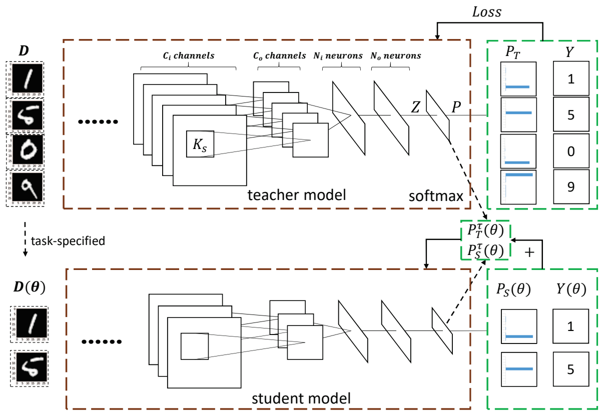

Fig. 1. The scheme of task-specified knowledge distillation

the redundancy in CNN models can be reduced by avoiding unimportant connections, filters and channels [5, 8, 10], as:

$$\begin{aligned} &\underset{C_i, C_o, N_i, N_o}{\arg\min} \ (CP_{conv} + CP_{fc}), \\ &subject\ to:\ acc_{loss} >= acc_{orig} - acc \end{aligned} \tag{3}$$

where acc represents the accuracy of compressed model, and acc_{orig} represents the accuracy of original model. acc_{loss} is the threshold. This can be understood to minimize the model complexity, and to guarantee the negligible accuracy loss in the compressed model.

Many popular CNNs with extraordinary performance have been trained with large datasets, like ImageNet with millions of samples covering thousands of classes. Nevertheless, in practical front-end systems, only few classes of targets will be interested in. By removing the redundancy representing these uninterested targets, the efficiency of new network could be further increased. Under this hypothesis, the problem should be extended to reduce task complexity related redundancy:

$$\begin{aligned} &\underset{C_i, C_o, N_i, N_o}{\arg\min} \ (CP_{conv} + CP_{fc}), \\ &subject\ to:\ Tacc_{loss} >= Tacc_{orig} - Tacc \end{aligned} \tag{4}$$

where $Tacc$ represents the accuracy of compressed model on specified task, and $Tacc_{orig}$ represents the accuracy of original model on specified task. $Tacc_{loss}$ is the threshold.

3. METHODOLOGY

Most recently, Hinton *et al.* [6] pointed out that the original cumbersome teacher models can be utilized to improve the performance of small student models by a transfer learning method, knowledge distillation.

In knowledge distillation, an additional, if not included in the original model, softmax layer is required at the top of

CNN, which converts the original feature vectors, i.e. logits Z, into probabilities for each class:

$$p_i^\tau = \frac{exp(z_i/\tau)}{\sum_j exp(z_j/\tau)} \tag{5}$$

where p_i^τ is the ith element of probability distribution vector P^τ; z_i is the ith element of logits vector Z; j sums all the classes; τ is a relaxation parameter, i.e. setting τ equal to 1 will be back compatible with normal CNN model. With the increase of τ, the probability distributions will be softened. The softened probability vector P^τ contains the relative similarity of the input class to other classes, which reveals how the CNN models discriminate and generalize between given classes. Although the softmax layer is must in knowledge distillation, the application does not have to use the softmax vector, e.g., they can still feed feature vectors of full-connected layer to the original classifier.

Let the original model be defined as the teacher model T and the expected small one as the student model S. Let P_S be the output probabilities of the student model. Let Y be the true label vector of a sample. P_T^τ from the teacher model is named as the soft targets for the student model. Let P_S^τ be the softened probabilities from the student model with the same relaxation τ. The cross entropy function is shown as follow:

$$H(P,Q) = -\sum_i p_i log(q_i) \tag{6}$$

where p_i is the ith element of vector P, and q_i likewise.

Aided by the soft targets from teacher model, the loss function in knowledge distillation is defined as:

$$L_{KD} = \frac{1}{N}\sum_N((1-\lambda)H(Y,P_S) + \lambda H(P_T^\tau, P_S^\tau)) \tag{7}$$

where N is the number of samples and λ is a tunable parameter to balance both cross entropies. By the introducing of $\lambda H(P_T^\tau, P_S^\tau)$, knowledge distillation helps the student model to learn how the original model treats the sample. Existing works [6, 9] have shown that compared with direct training, the student model converges with higher accuracy similar to the original one.

The extension of knowledge distillation based solution into the task scalable version is more than convenient. Set the dataset to train teacher model as D and the dataset to train student model as $D(\theta)$ named as the transfer set, which has less task complexity than former, and typically is a sub set of D, containing only task of interested targets. As shown in fig.1, a subset of corresponding soft targets can be obtained as $P_T^\tau(\theta)$. This has been shown in algorithm 1 as a variation form, since in the real implementation, this action is equal to directly acquiring the $P_T^\tau(\theta)$ from $D(\theta)$, which aims to avoid the intermedia step of P_T^τ. Similarly, we could obtain the task-specified version of $Y(\theta)$.

Now, the task-specified student model $S(\theta)$ will be trained with an modified loss function $L_{KD}(\theta)$:

$$\begin{aligned} L_{KD}(\theta) = \tfrac{1}{N}\textstyle\sum_N((1-\lambda)H(Y(\theta), P_S(\theta)) \\ +\lambda H(P_T^\tau(\theta), P_S^\tau(\theta))) \end{aligned} \tag{8}$$

where $P_S(\theta)$, and $P_S^\tau(\theta)$ are the outputs of student model in each iteration. The notation θ is to show the fact that they will ideally only response to $D(\theta)$.

The detailed algorithm has been provided in the following table:

Algorithm 1 Task-specified Knowledge Distillation

```
Input: $\boldsymbol{D}, \boldsymbol{T}, \boldsymbol{\theta}$
Output: $\boldsymbol{S(\theta)}$
1: add a softmax layer to $\boldsymbol{T}$
2: use $\boldsymbol{D}$ to train the teacher model $\boldsymbol{T}$
3: use $\boldsymbol{T}$ to capture soft targets $\boldsymbol{P_T^\tau(\theta)}$ from each sample in $\boldsymbol{D(\theta)}$
4: set the architecture of the student model $\boldsymbol{S(\theta)}$
5: train the student model with soft targets $\boldsymbol{P_T^\tau(\theta)}$ and $\boldsymbol{D(\theta)}$, iteratively until the accuracy converges
       $\boldsymbol{S(\theta) = \arg\min_{S(\theta)} L_{KD}(\theta)}$
```

Noting that, as the original model T may already exist, or even be equipped with a softmax layer as classifier, this algorithm may have some variations starting from different steps. It should also be noted that, as the size, or more precisely the architecture, of student model was manually deigned. It could be in the various forms, e.g. we could even remove a layer or add a layer if we would like to do so. As discussed in [1–3], the architecture not the size of parameters will decide the performance of the deep framework, for example a thinner but deeper network may bring better performance. The knowledge distillation method will secure the possibility to utilize the potential architecture gain in the future.

4. EXPERIMENTS

We conducted our experiments on two classification datasets, MNIST and CIFAR10. All the codes were deployed in a desktop PC with one K80 GPU card. The architecture of the teacher model for MNIST is 20-50-500-10, where the first two layers are convolutional and last two are full-connected. Similarly, the architecture used for CIFAR10 is 32-32-64-10, where the first three layers are convolutional and the last one is full-connected. For both models, outputs of last full-connected layer are treated as inputs of the softmax layer. In the experiments τ=3 is selected, although it works similarly in a larger range within [3:10]. λ is carefully selected as 0.9 to observe system performance with emphasis on transfer effect, i.e. transfer learning contributes more than true label training.

In the experiments, 10 classes of datasets will be divided into subsets to demonstrate the performance of task-specified student model. We set our transfer sets according to the following strategy. Take MNIST for example. The training dataset D for the teacher model covers the whole training samples, say all ten classes. And we set transfer datasets covering from two classes to nine classes, from $D(\theta_2)$ to

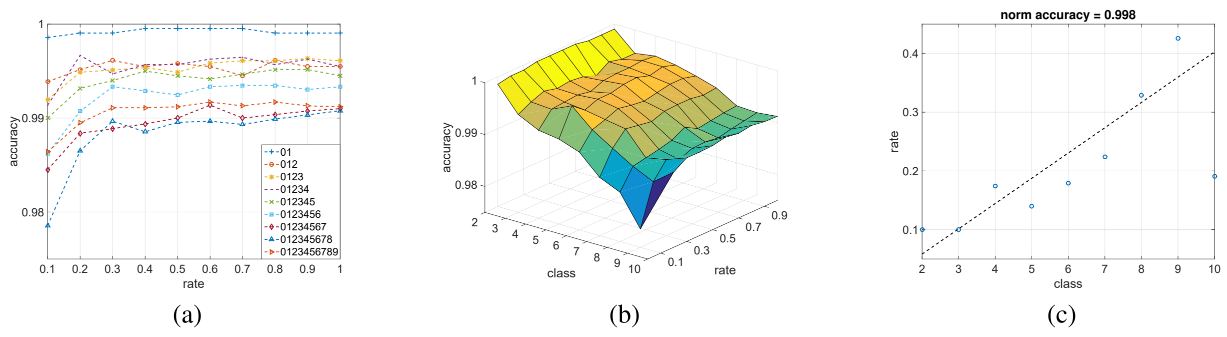

Fig. 2. (a) Results of MNIST; (b) re-drawed 3D version; (c) relationship between rate and class given normalized accuracy.

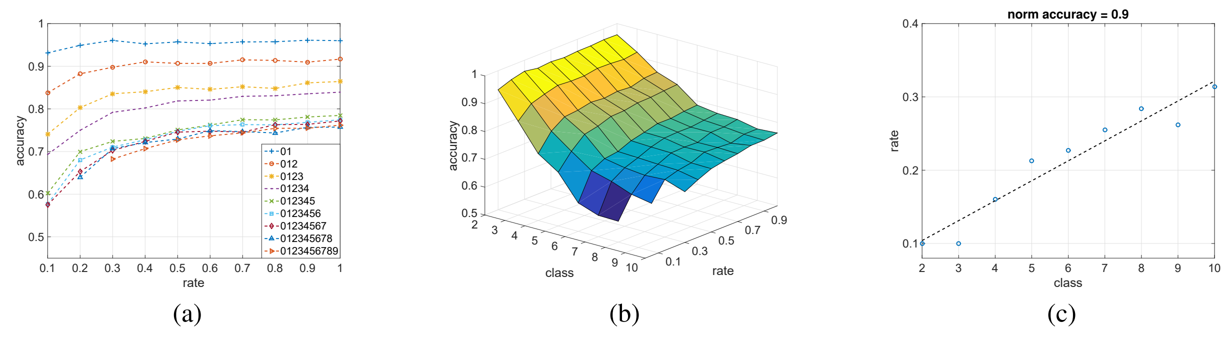

Fig. 3. (a) Results of CIFAR10; (b) re-drawed 3D version; (c) relationship between rate and class given normalized accuracy.

$D(\theta_9)$. Let $D(\theta_2)$ cover samples with label '0' and '1', $D(\theta_3)$ cover '0', '1' and '2', which will guarantee that $D(\theta_2) \subseteq D(\theta_3) \cdots \subseteq D$. Under these assumptions, the difficulty of sub tasks will be monotonically increasing.

The architecture of student model is manually designed in knowledge distillation. In the experiments, all layers except the last full-connected layer and the softmax layer are compressed with the same rate. For example, if the compression rate for MNIST model is 0.5, the structure is 10-25-250-10. All the retrained student models will be tested to show the performance of proposed algorithm.

Fig.2(a) and fig.3(a) are the direct results of MNIST and CIFAR10. The results brief the basic fact that the accuracy decreases with the size of model. Moreover, those compressed models for simpler task suffer less accuracy loss. As fig.2(a) shown, in the results of MNIST, the accuracy of the simplest task $D(\theta_2)$ drops 0.05 percent while the accuracy for the original D drops 0.48 percent, both with size scale of 0.1. Similarly, the accuracy for $D(\theta_2)$ drops 2.85 percent while the accuracy for $D(\theta_8)$ drops 19.69 percent for CIFAR10 with the same size scale of 0.1 ("rate" in the figure). These results illustrate that for lower task complexity, the models have more redundancy, which can be further compressed. Fig.2(b) and fig.3(b) are the re-drawed 3-D version of the experiment results to show a global performance change in the experiments.

From fig.2(b), and fig.3(b), we could derive fig.2(c) and fig.3(c), which provide an intuitive validation to our hypothesis. The performance of different $D(\theta)$ has been normalized, which shows the relative performance drop. Then, the estimated model sizes to achieve the same performance drop, i.e. a threshold, have been provided in fig.2(c) and fig.3(c). Easy to notice that for the threshold of 99.8%, a simpler task $D(\theta_2)$ could remove more redundancy than the original D for MNIST, while the situation for CIFAR10 is likewise.

5. CONCLUSION & FUTURE WORK

In this paper, we first propose the hypothesis that the redundancy among CNN models or other similar deep learning systems will correspond to task complexity, which means potential performance gain especially for the front-end deployed vision systems. We design the algorithm from the recently proposed knowledge distillation to further reduce task-specified redundancy. Experiments on MNIST and CIFAR10 have been provided to validate our hypothesis and algorithm. Due to the page limitation, detailed experiments will be provided in our future work.

Obviously, knowledge distillation benefits from its architecture of an *a priori* model size, which is critical to resource constraint embedded systems. But at the same time, the retrained student model may still have shared redundancy, i.e., the low contribution network components. A following pruning stage may be employed to further improve the performance in our future work. Also as discussed in section 4, the task complexity may not be monotonically scaled with the interested targets number. A carefully designed metric to better describe the task complexity will be examined as well.

6. REFERENCES

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[2] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR, abs/1409.1556*, 2014.

[3] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[4] Sajid Anwar, Kyuyeon Hwang, and Wonyong Sung, "Fixed point optimization of deep convolutional neural networks for object recognition," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2015, pp. 1131–1135.

[5] Song Han, Jeff Pool, John Tran, and William Dally, "Learning both weights and connections for efficient neural network," in *Advances in Neural Information Processing Systems*, 2015, pp. 1135–1143.

[6] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the knowledge in a neural network," *Advances in Neural Information Processing Systems*, 2014.

[7] Ping Luo, Zhenyao Zhu, Ziwei Liu, Xiaogang Wang, and Xiaoou Tang, "Face model compression by distilling knowledge from neurons," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[8] Adam Polyak and Lior Wolf, "Channel-level acceleration of deep face representations," *IEEE Access*, vol. 3, pp. 2163–2175, 2015.

[9] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio, "Fitnets: Hints for thin deep nets," *International Conference on Learning Representations*, 2015.

[10] Yi Sun, Xiaogang Wang, and Xiaoou Tang, "Sparsifying neural network connections for face recognition," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4856–4864, 2016.

[11] Vadim Lebedev and Victor Lempitsky, "Fast convnets using group-wise brain damage," *CoRR, abs/1506.02515*, 2015.

[12] Song Han, Huizi Mao, and William J Dally, "Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding," *CoRR, abs/1510.00149*, vol. 2, 2015.

[13] Jimmy Ba and Rich Caruana, "Do deep nets really need to be deep?," in *Advances in neural information processing systems*, 2014, pp. 2654–2662.

[14] Sajid Anwar, Kyuyeon Hwang, and Wonyong Sung, "Structured pruning of deep convolutional neural networks," *CoRR, abs/1512.08571*, 2015.

[15] Ilya Kalinowski and Vladimir Spitsyn, "Compact convolutional neural network cascade for face detection," *CoRR, abs/1508.01292*, 2015.

[16] Babak Hassibi and David G Stork, *Second order derivatives for network pruning: Optimal brain surgeon*, Morgan Kaufmann, 1993.

[17] Yann LeCun, John S Denker, Sara A Solla, Richard E Howard, and Lawrence D Jackel, "Optimal brain damage," in *Advances in neural information processing systems*, 1989, vol. 2, pp. 598–605.

[18] George Papamakarios, "Distilling model knowledge," *CoRR, abs/1510.02437*, 2015.

[19] Pulkit Agrawal, Ross Girshick, and Jitendra Malik, "Analyzing the performance of multilayer neural networks for object recognition," in *European Conference on Computer Vision*. Springer, 2014, pp. 329–344.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.