Computers and Electrical Engineering 000 (2018) 1–9

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Convolutional neural network simplification via feature map pruning☆

Junhua Zou[a], Ting Rui[a,b,*], You Zhou[c], Chengsong Yang[a], Sai Zhang[a]

[a] Army Engineering University of PLA, Nanjing, China
[b] State Key Lab. For Novel Software Technology, Nanjing University, Nanjing, China
[c] Jiangsu Institute of Commerce, Nanjing, China

ARTICLE INFO

Article history:
Received 20 August 2017
Revised 24 January 2018
Accepted 24 January 2018
Available online xxx

Keywords:
Convolutional neural network
Feature maps pruning
Discriminability
Critical points

ABSTRACT

Convolutional neural networks (CNNs) have been a focus area of machine learning in recent years, and they are widely used in vision and speech processing because of their superior performance. However, CNNs are usually resource-heavy to ensure higher accuracy, i.e., an accurate network with millions of parameters requires high performance computing devices. This prevents the use of CNNs in resource-limited hardware. In this paper, we propose a novel CNN simplification method to prune feature maps with relatively low discriminability magnitudes, which can produce a simplified CNN with reduced computational cost. Specifically, we define the critical points among the discriminability values of feature maps in each convolutional layer, and use these critical points to easily find the best pruning number of feature maps. Our experimental results show that in each convolutional layer of the VGG model, 15.6% to 59.7% of feature maps can be pruned without any loss of accuracy in classification tasks.

1. Introduction

Convolutional neural networks (CNNs) [1], a type of deep learning network architecture, have been widely used in various applications [2–7] such as image classification [8,9], object detection [10,11], saliency analysis [12], and target tracking [13]. To improve the efficiency of CNNs, weight sharing, local connection, and subsampling operations are used to greatly reduce the number of network weights and the network complexity. These three techniques also improve the adaptability and robustness of CNNs. CNNs are highly invariant to translation, rotation, and scaling. However, some recent studies show that there is still a considerable amount of redundancy in most existing CNNs [14–19].

With the continuous development of CNNs, their performance is rapidly increasing, and their depth, i.e., layer number, is also growing. As CNNs increase in depth, the computational speed and memory requirements increase proportionally. Past results of the ImageNet Large Scale Visual Recognition Competition (ILSVRC) show that the depth of a CNN has a positive impact on the performance, but the relationship between the network width (i.e., the number of feature maps per layer) and the performance of a CNN are not discussed. Considering the limitations of mobile phones in CPU/GPU performance, storage capacity, and computational power, it is currently difficult to implement CNNs on these devices. For instance, mobile

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Associate Editor Dr. Huimin Lu.

* Corresponding author at: Army Engineering University of PLA, No. 1 Haifu Lane, Qinhuai District, Nanjing, China.
E-mail addresses: 278287847@qq.com (J. Zou), rtinguu@sohu.com (T. Rui), 354442511@qq.com (Y. Zhou), ycsdongshan@163.com (C. Yang), 466908114@qq.com (S. Zhang).

https://doi.org/10.1016/j.compeleceng.2018.01.036
0045-7906/

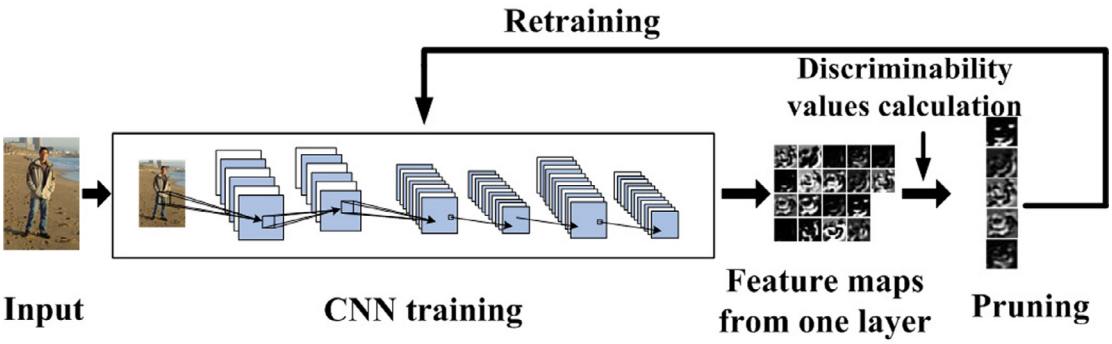

Fig. 1. Three stages of our CNN Simplification method: training, pruning, and retraining. Firstly, we train a VGG-like network which has achieved the best performance. Then, we prune feature maps which are redundant to the network, and also prune convolutional filters which correspond to the pruned feature maps. Finally, we retrain the network after pruning one layer and repeat these steps layer by layer from the bottom layer to the top layer.

phone apps on app stores are rarely larger than 200 MB in size. However, commonly used CNNs trained on the ImageNet dataset can achieve a parameter number of at least 95 million, and the computational cost of a CNN is even larger. Similarly, the storage capacity and operational performance of unmanned systems are limited. Hence, CNNs are usually too heavy for unmanned systems also. For example, over 95 MB memory and 3.8×10^9 float multiplications are required to launch the ResNet-50 [20] for processing an image.

In order to apply a CNN in real time, it is important to simplify it to improve its computational speed. The simplification of CNNs has recently become an emerging and important research topic in deep learning. A basic requirement in CNN simplification is that the performance of a CNN after simplification should be similar to its performance before simplification. Recent studies [14–16] show that the gap between large scale CNNs and devices with limited resources can be bridged. In a CNN, the network size (i.e., the number of filters and feature maps) should be adjusted as per the requirements of different tasks. Taking VGG-16 (a widely used deep CNN) as an example, the numbers of filters from the bottom layer to the top layer are 64, 64, 128, 128, 256, 256, 256, 512, 512, 512, 512, 512, and 512, respectively. Such a common network setting easily leads to redundancy for some tasks. Therefore, we aim to determine the contribution of feature maps on network performance by calculating their discriminability values.

In recent years, many works have been done on CNN simplification. As a summary of the literature, the current methods for CNN simplification are as follows:

1. Replacing complex networks with simple networks. In simple tasks such as binary classification, we can design a simple network to replace the complex network, which maintains classification accuracy. This approach requires the design of a new network, and the new network may not work when the task becomes complex.
2. Binarizing the complex network values [18]. The input, weights, and response parameters of complex network can be binarized to reduce the number of bits required to represent each weight. This approach can achieve shorter runtime, but the network performance will distinctly decrease.
3. Compressing trained complex networks [14]. Deep compression can achieve network simplification by network pruning, weights quantization, and Huffman coding. However, quantitative analysis and description of the extent of compression is still lacking.

Based on the above observation, we simplify a trained CNN via feature map pruning. We not only aim to reduce the network size by feature map pruning, but also analyze the feature maps quantitatively and find a way to quickly determine the feature map pruning number. With an effective pruning number determination method, we can avoid pruning feature maps one by one in each convolutional layer and save much time in retraining. In this paper, we need to prune feature maps in each convolutional layer. Hence, we design a feature map screening strategy to calculate their discriminability values, and define the concept of critical points which can quickly find the relatively best pruning number of feature maps. An overview of our method is shown in Fig. 1. The proposed method can be divided into the following three stages:

1. In CNN training stage, we train a VGG-like network with 3x3 filters and a padding of 1; these settings ensure the size of feature maps is only changed in max pooling layers. Batch normalization and dropout are both used in the network.
2. In the feature map pruning stage, we use a screening strategy to calculate discriminability values of feature maps, and prune feature maps which have small discriminability values. Together with feature map pruning, convolutional filters which correspond to pruned feature maps are also pruned. In the matter of feature map pruning number, we define the concept of a critical point which can quickly determine the relatively best pruning number of feature maps.
3. In the CNN retraining stage, we prune feature maps layer by layer from the bottom layer to the top layer, and retrain the CNN after pruning each layer until network performance is restored. We repeat these operations until all convolutional layers are pruned.

In summary, our CNN simplification method has the following contributions:

Please cite this article as: J. Zou et al., Convolutional neural network simplification via feature map pruning, Computers and Electrical Engineering (2018), https://doi.org/10.1016/j.compeleceng.2018.01.036