ATLANTIS PRESS

Advances in Computer Science Research, volume 70

2nd International Conference on Mechatronics Engineering and Information Technology (ICMEIT 2017)

Compression of Conditional Deep Learning Network for Fast and Low Power Mobile Applications

Lijie Li [1,2,a], Yan Zhang [1,2,b], Pengfei Wang [1,2,c]

[1]Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China;

[2]Beijing Key Laboratory of Computational Intelligence and Intelligent System, Beijing University of Technology, Beijing 100124, China

[a]wy13622037317@sina.com, [b]591139047@qq.com, [c]wpf19901012@163.com

Keywords: CDLN, one-shot whole network compression scheme, module size.

Abstract. CDLN(Conditional Deep Learning Network)is a structure of convolution neural network with multiple classifiers. CDLN could improve the speed for the task of classification while the module of the network is still too large for mobile devices. To address this issue, a method for compressing CDLN, which is named one-shot whole network compression scheme. In the experiments, the module size and time cost are significantly reduced while the accuracy of the network losses a little.

1. Convolutional Neural Network

Convolutional Neural Network (CNN), which is widely applied in computer vision, is a representative method of deep learning due to its excellent learning ability for high dimensional data feature[1]. Recent years, with the emergence of related learning techniques, optimization techniques and hardware technology, convolution neural network has explosively developed[2]. ImageNet Large Scale Visual Recognition Challenge (ILSVRC) is a standard challenge for large-scale recognition. CNN has been widely used in classification activities of ImageNet and has achieved excellent classification results[3]. From the 8-layer AlexNet[4] to the 19-layer VGGNet[5] and the 152-layer ResNet[6], CNN is going deeper and deeper and the top-5 error reduces from 15.3% to 6.8% and 3.57%. However, the cost time and energy of forward propagation when training the network increase drastically[7]. For example, the running time of VGGNet is 20 times of AlexNet when performing classification tasks in the same dataset and experimental condition[8]. In addition, engineers and developers usually need to take time cost in concern in the context of industrial and commercial applications[9]. For instance, the online search engines need to response rapidly, and the cloud service needs to deal with thousands of pictures per second. Otherwise, the applications for scene recognition on smart phones and portable devices, which are lack of powerful ability for computing, need to response quickly.

In 2011, Vanhoucke et al[10] research the method of the code optimization to speed up execution of CNN to reduce the runtime of the network.

In 2013, Mathieu et al[11] convolute the convolution value as the dot product of the Fourier domain, when repeat the use of the same transform feature map, for the aid of reducing the runtime.

In 2015, Kim et al[12] apply Tucker decomposition to extract the shared information between the convolution layer and the execution rank selection. This method reduces the number of network parameters for fast inference at the expense of the accuracy.

In 2016, Panda p et al[13] propose a Conditional Deep Learning Network(CDLN) which adding extra linear classifiers behind the convolution layers. Through monitoring the output of the liner classifiers the ones that are easy to classify is classified in advance and exit the network for fast inference. The structure of the network is modified in CDLN which is a novel way.

In this paper, a method for compressing the module of the network is applied to compress the CDLN for fast and low power mobile applications.

2. Convolution Neural Network with multiple classifiers

2.1 The CDLN Network

The convolution neural network with multiple classifiers is based on CDLN. The structure of the network is showed in Fig.1[13]. There are three convolution layers which followed by liner classifiers. The first two linear classifiers are followed by activation functions which mainly contains a confidence value δ that determines the samples exit the network or not.

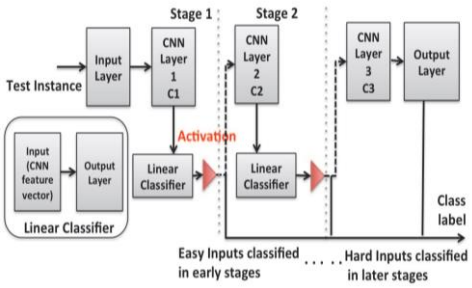

Fig.1 The architecture of CDLN

The training method of CDLN is as follows:

a. Training a baseline CNN that contains three convolution layers.

b. Extracting the convolution feature of the first two convolution layers.

c. Training the liner classifiers using the convolution feature and the loss function is mean square error function.

2.2 Compression of CDLN

In the convolution neural network, the input of the convolution layer is a third order tensor χ, and the size is $H \times W \times S$. After the convolution of the convolution layer, the output tensor is the third order tensor Y and its size is $H' \times W' \times S'$. The convolution operation formulas are as follow,

$$y_{h',w',t} = \sum_{i=1}^{D}\sum_{j=1}^{D}\sum_{s=1}^{S} k_{i,j,s,t}\chi_{h_i,w_j,s} \tag{1}$$

$$h_i = (h'-1)\Delta + i - P \tag{2}$$

$$w_j = (w'-1)\Delta + j - P \tag{3}$$

Where k is the 4-channel tensor, the size is $D \times D \times S \times T$, Δ is the convolution network training stride, and P is the zero-filling dimension.

The four-way tensor k can be decomposed by the following formula,

$$k_{i,j,s,t} = \sum_{r_1=1}^{R_1}\sum_{r_2=2}^{R_2}\sum_{r_3=1}^{R_3}\sum_{r_4=1}^{R_4} c'_{r_1,r_2,r_3,r_4} U^{(1)}_{i,r_1} U^{(2)}_{j,r_2} U^{(3)}_{s,r_3} U^{(4)}_{t,r_4} \tag{4}$$

Where $c'$is the kernel tensor of size $R_1 \times R_2 \times R_3 \times R_4$, $U^{(1)}$,$U^{(2)}$,$U^{(3)}$ and $U^{(4)}$ are the factor matrices, which are $D \times R_1$, $D \times R_2$, $S \times R_3$ and $T \times R_4$.

The decomposition formula becomes as follow,

$$k_{i,j,s,t} = \sum_{r_3=1}^{R_3}\sum_{r_4=1}^{R_4} c_{i,j,r_3,r_4} U^{(3)}_{s,r_3} U^{(4)}_{t,r_4} \tag{5}$$

Three formulas for approximating convolution layers are obtained as follow,

$$Z_{h,w,r_3} = \sum_{s=1}^{S} U^{(3)}_{s,r_3}\chi_{h,w,s} \tag{6}$$

$$Z'_{h',w',r_4} = \sum_{i=1}^{D}\sum_{j=1}^{D}\sum_{r_3=1}^{R_3} C_{i,j,r_3,r_4} Z_{h_i,w_i,r_3} \tag{7}$$

$$y_{h',w',r_4} = \sum_{r_4=1}^{R_4} U^{(4)}_{t,r_4}, Z'_{h',w',r_4} \tag{8}$$

Where Z and $Z'$are intermediate tensors, and the sizes are H × W × R_3 and $H' \times W' \times R_4$.

In the formula, the convolution operation requires D^2ST parameters and $D^2STH'W'$ sub-multiplication addition. By Tucker-2 method of compression, the compressed network than the original network to reduce M times, the speed increase E times. The formula is as follows,

$$\mathrm{M} = \frac{D^2ST}{SR_3 + D^2R_3R_4} \tag{9}$$

$$\mathrm{E} = \frac{D^2STH'W'}{SR_3HW + D^2R_3R_4H'W' + TR_4H'W'} \tag{10}$$

Its compression value and speed increase value generally does not exceed ST/R_3R_4.

The rank (R_3, R_4) in 2.3.1 is a very important super parameter of the convolutional neural network. Changes in these two values are closely related to network performance, such as storage space size and cost of consumption, and accuracy. Here we introduce a data-driven one-stop selection method, that is, data-driven one-shot. The method combines the empirical Bayes and automatic relevance determination. The empirical variable Bayesian learning is used to design probability Tucker model. Since the rank selection results in this model are heavily dependent on the initialization conditions, the noise variable evaluation strategy. Therefore, here we introduce a rank selection method based on VBMF (variational Bayesian matrix factorization) global analysis.

It can be concluded from 2.3.1 that the global analysis of VBMF is a very effective tool because it can automatically solve the noise variables and rank, and provide theoretical conditions for the complete rank restoration.

Therefore, VBMF is applied to obtain the matrix of the kernel tensor K in this paper.

3. Experiments and Results

LeNet-5 and AlexNet are the baseline networks in this paper. Then CDLNs are structured which are named CDLN-L and CDLN-A. CDLN-L and CDLN-A are compressed by using the compression method proposed in this paper.

The datasets of the experiments is CIFAR-10 dataset[14].

The equipment of the experiment are notebook with Inter(R) Core(TM) i3-2330M CPU @2.20GHz 2.19GHz and mobile phone.

The results of experiments are showed in table 1. The Time in the table is the average time of the forward propagation of the network.

The results show the compression method significantly compresses the network. Compressed CDLN-L is 70.50% and 72.60% faster than CDLN-L when the network performs the forward propagation on the notebook and MX5. Compressed CDLN-A is 72.14% and 58.48% faster than CDLN-A. The weights of CDLN-L and CDLN-A are compressed by 75.68% and 81.54%. The accuracy of the networks just loss a little.

Table 1. Comparison of CDLN and the compressed CDLN

Network	Accuracy	Weights	Time	
			Notebook	MX5
LeNet-5	49.32	35M	355ms	742ms
CDLN-L	50.98	37M	278ms	478ms
Compressed CDLN-L	50.21.	9M	82ms	131ms
AlexNet	65.33	61M	523ms	1264ms
CDLN-A	66.65	65M	341ms	855ms
Compressed CDLN-A	66.52	12M	95ms	355ms

4. Conclusion

This paper proposes a method for compressing CDLN for fast and lower power mobile applications. The main conclusions are as follows:

a. A method for compressing CDLN is proposed in this paper. The kernel tensor is decomposed and then the rank is selected based on VBMF global analysis. In this way, the module of the network can be significantly compressed.

b. In the experiments, LeNet-5 and AlexNet are the baseline network. CDLN-L and CDLN-A are constructed and compressed by the compression method. The results show Compressed CDLN-L is 70.50% and 72.60% faster than CDLN-L on the notebook and mobile device. Compressed CDLN-A is 72.14% and 58.48% faster than CDLN-A. The weights of CDLN-L and CDLN-A are compressed by 75.68% and 81.54%.

Therefore, one-shot whole network compression scheme can compress the module of CDLN and the accuracy losses a little.

References

[1]. Guo Y, Liu Y, Oerlemans A, et al. Deep learning for visual understanding: A review[J]. Neurocomputing, 2016, 187: 27-48.
[2]. He K, Zhang X, Ren S, et al. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification[C]//Proceedings of the IEEE International Conference on Computer Vision. 2015: 1026-1034.
[3]. Russakovsky O, Deng J, Su H, et al. Imagenet large scale visual recognition challenge[J]. International Journal of Computer Vision, 2015, 115(3): 211-252.
[4]. Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.
[5]. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.
[6]. He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[J]. arXiv preprint arXiv:1512.03385, 2015.
[7]. Teerapittayanon S, McDanel B, Kung H T. Branchynet: Fast inference via early exiting from deep neural networks[C]. ICPR, 2016.
[8]. Kim Y D, Park E, Yoo S, et al. Compression of deep convolutional neural networks for fast and low power mobile applications[J]. arXiv preprint arXiv:1511.06530, 2015.
[9]. He K, Sun J. Convolutional neural networks at constrained time cost[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 5353-5360.
[10]. Vanhoucke V, Senior A, Mao M Z. Improving the speed of neural networks on CPUs[J]. 2011.
[11]. Mathieu M, Henaff M, LeCun Y. Fast training of convolutional networks through FFTs[J]. arXiv preprint arXiv:1312.5851, 2013.
[12]. Kim Y D, Park E, Yoo S, et al. Compression of deep convolutional neural networks for fast and low power mobile applications[J]. arXiv preprint arXiv:1511.06530, 2015.
[13]. Panda P, Sengupta A, Roy K. Conditional Deep Learning for energy-efficient and enhanced pattern recognition[C]//2016 Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE, 2016: 475-480.
[14]. Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images[J]. 2009.