Advances in Engineering Research (AER), volume 124
2nd International Symposium on Advances in Electrical, Electronics and Computer Engineering
(ISAEECE 2017)

Relationship between Complexity and Precision of Convolutional Neural Networks

Xiaolong Ke[1, a], Wenming Cao[2, b], Fangfang Lv [1*]

[1,2]College of Information Engineering, Shenzhen University, China

[a]kxlong@aliyun.com, [b]wmcao@szu.edu.cn,

*corresponding author email: 1517799809@qq.com

Keywords: Convolutional neural networks, image classification, complexity control, complexity-precision modeling

Abstract. Convolutional neural networks (CNNs) have been successfully applied to the computer vision areas in recent years. However, these high performing CNNs generally involve intensive computation, which is unaffordable for many real-time applications. In this paper, we study the impact of four important network parameters $\mathbf{T}$. Then we develop mathematical models to characterize the relationship and tradeoff between the complexity C and precision P of CNNs. Once the models C($\mathbf{T}$) and P($\mathbf{T}$) are obtained, we are able to perform complexity-precision optimization to minimize the CNN complexity while achieving the target precision level by selecting the optimal configuration of four network parameters $\mathbf{T}$.

1. Introduction

Although CNNs architectures are of great help to push the limits of precision [1, 2], it is unwise for practical applications in view of they are time-consuming. How to allocate the computational resources within the CNN system so as to maximize the performance under the computational resource constraint, or minimize the complexity while achieving the target precision is need to understand. To address the issue, we propose to develop a complexity-precision analysis and optimization framework for CNNs. Specifically, we identify a set of control parameters $\mathbf{T}$ to control the computational complexity C of the CNN to make it complexity-scalable. Certainly, these parameters will also have direct impact on the performance on CNN, often measured by its precision P in classification. Through extensive experiments of CNN training and testing with different configurations of $\mathbf{T}$, we develop mathematical models to characterize the relationship between complexity C, precision P and the control parameters $\mathbf{T}$.

2. Complexity-precision analysis

2.1 Complexity-Scalable CNNs. It has been observed that a majority of the computational resources are consumed by the convolutional layers [3] whose computational complexity depends on the following four major parameters: number of filters (f), input size (s), filter size (k), and number of layers (l) [4]. We denote the corresponding values of those complexity control parameters in the baseline CNN by $\mathbf{T}_0 = (f_0, s_0, k_0, l_0)$. The baseline architecture we used was one of Alex's setups [5].

To reduce the computational complexity of CNN, we start to dial down the values of these parameters until the model is useless to predict. For each configuration of the complexity control parameters $\mathbf{T} = (f, s, k, l)$, we re-train the CNN and test it on the test dataset, record average computational complexity C (run-time) of the forward test network.

2.2 Modeling the Complexity of CNNs. The convolutional layers contribute to most of the computational complexity in CNNs. The convolution layer performs filtering on the feature maps. The input to the filters is f_{i-1} feature maps from the previous layer i-1. There are f_i filters in the

Copyright © 2017, the Authors. Published by Atlantis Press.
This is an open access article under the CC BY-NC license (http://creativecommons.org/licenses/by-nc/4.0/).

current layer i. To produce a data point in the feature map of size $s_i \times s_i$, each filter with a kernel size of $k_i \times k_i$ will perform $k_i \times k_i$ multiplications between the filter coefficients and input data points.

$$C = \alpha_1 \cdot \left(\sum_{i=1}^{N_C} f_{i-1} \cdot k_i^2 \cdot f_i \cdot s_i^2\right) + \alpha_0, \quad (1)$$

It should be noted that the actual running time of the CNN depends on the actual implementation and the hardware. But, during our experiments, we observe that the running time is proportional to the complexity C given in (1). Therefore, we can set the run-time complexity of the baseline model as 1.0, then use (1) to predict the relative run-time complexity of the new CNN configured by the control parameters $T = (f, s, k, l)$. The model coefficients α_0 and α_1 are obtained from actual measurements. For the LFW dataset, we have α_0 = 0.014 and α_1 = 0.626×10^7. To test the accuracy of the complexity model, we randomly choose 50 different configurations of the control parameters, train and test the CNN, the result is shown in Fig.1 (left), with a relative estimation error less than 2.5%.

2.3 Precision Modeling of CNNs. In deep learning practice, the performance of CNNs depend many factors, it will be even more challenging to develop a theoretical model to predict its performance, specifically, the classification precision. Therefore, we choose an operational approach for CNN precision modeling. We first analyze and model the behaviors of P with respect to each individual control parameter. We then extend these individual 1-D models to construct compound models for P(T) with two control parameters, and finally establish the models for all control parameters. We start with the baseline CNN for the LFW dataset described in the above section. The baseline values of the control parameters are (1) f = 64, (2) s = 64, (3) k = 5 and (4) l = 3. We then progressively reduce the values of these control parameters to reduce the overall computational complexity. For example, for the f of each layer, we choose the candidate values of Ω_f = {64, 48, 40, 32, 24, 16, 12, 8, 4, 2}. For s, the candidate values are Ω_s = {64, 56, 48, 40, 32, 24, 20, 12}. For k, we set the candidate values to be Ω_k = {5, 4, 3}. For l, we choose Ω_l = {3, 2, 1}.

2.3.1 1-D Precision Models: Specifically, we vary one control parameter while keeping the rest at the baseline values. For example, we decrease the number of filters from its baseline value over the candidate set Ω_f , train and test the CNN model on the LFW dataset, and record the classification precision P(f). Using the same way, we can get the behaviors of P with respect to other control parameters, namely, the input size, filter size, and the number of layers. We find that these behaviors can be well approximated by the following model.

$$P(\gamma_n) = \beta_{n1} \cdot e^{\beta_{n2} \cdot 1/\gamma_n} + \beta_{n3} \cdot \gamma_n + \beta_{n4}, \quad (2)$$

where γ_n represents the n-th control parameters in $T = (f, s, k, l)$. β_{nm} are the model coefficients, which are listed in Table I.

TABLE I. COEFFICIENTS FOR PRECISION MODELS

Control Parameters	Notation	β_{n1}	β_{n2}	β_{n3}	β_{n4}
Number of Filters f	γ_1	0.9259	0.02671	-0.1088	0.3551
Input Image Size s	γ_2	0.9267	0.02357	-0.499	-0.1029
Filter Size k	γ_3	0.9164	0.03487	-0.2085	-0.7635
Number of Layers l	γ_4	2.384	-0.4167	0.9022	0.05827

2.3.2 2-D Precision Models: Based on the above 1-D models $P(\gamma_n)$ which characterize the relationship between precision P and each individual control parameters $\gamma_n \in T$, we move to the next step to understand the relationship between precision P and two control parameters (f, s) which are number of filters and input image size. Specifically, for each model, we let two parameters in T vary over their candidate value sets and fix other parameters at their baseline values. Though a series of experiments, we can get many real precision points values. We find out that these points can be well approximated by the following model which is an extension from the 1-D models developed in the above section:

$$P(f,s) = n_{11} \cdot e^{n_{12}\cdot(1/f)+n_{13}\cdot(1/s)} + n_{14} \cdot f + n_{15} \cdot s + n_{16}, \quad (3)$$

Following a similar procedure, we study and model the relationship between the precision P and other pairs of control parameters and construct models for P(f, k) with the number filters and filter kernel size k as control parameters, and for P(f, l) with number filters and the number of layers as control parameters. Respectively, and the corresponding models are given by

$$P(f,k) = n_{21} \cdot e^{n_{22}\cdot(1/f)+n_{23}\cdot(1/s)} + n_{24} \cdot f + n_{25} \cdot s + n_{26}, \quad (4)$$

$$P(f,l) = n_{31} \cdot e^{n_{32}\cdot(1/f)+n_{33}\cdot(1/s)} + n_{34} \cdot f + n_{35} \cdot s + n_{36}, \quad (5)$$

We can see that the actual samples obtained from CNN experiments can be well approximated by these models. Table II summarizes the model coefficients for the above three 2-D precision models.

2.3.3 An Integrated Model for the Precision: From the above 1-D and 2-D analysis and modeling of the DCNN precision behaviors with respect to control parameters T =(f,s,k,l), we know that they

TABLE II. COEFFICIENTS FOR THE 2-D PRECISION MODELS.

Model $P(f,s)$	n_{11}	n_{12}	n_{13}	n_{14}	n_{15}	n_{16}
Coefficients	0.972	-0.01964	-0.02158	0.007259	-0.09404	0.0001
Model $P(f,k)$	n_{21}	n_{22}	n_{23}	n_{24}	n_{25}	n_{26}
Coefficients	0.9952	-0.00125	-0.03308	0.02473	-0.0334	0.0025
Model $P(f,l)$	n_{31}	n_{32}	n_{33}	n_{34}	n_{35}	n_{36}
Coefficients	0.9377	-0.0011	-0.01028	0.0148	-0.04284	0.6571

follow an exponential behavior plus a linear shift. Based on this important observation, we propose the following integrated model for the precision

$$P(f,s,k,l) = \lambda_1 \cdot e^{\lambda_2\frac{1}{f}+\lambda_3\frac{1}{s}+\lambda_4\frac{1}{k}+\lambda_5\frac{1}{l}} + \lambda_6 \cdot f + \lambda_7 \cdot s + \lambda_8 \cdot k + \lambda_9 \cdot l + \lambda_{10}, \quad (6)$$

where λ_1 =1.0221, λ_2 =-0.0015, λ_3 =-0.0218, λ_4 =-0.0293, λ_5 =-0.0103, λ_6 =-0.16, λ_7 =-0.011, λ_8=-0.0192, λ_9=-0.0011 and λ_{10}=0.0042. To demonstrate that this model is able to capture and approximate the true precision behavior of the DCNN, we randomly choose 50 different configurations of the control parameters T =(f,s,k,l), train the DCNN model using T, run the test module with this model, and record the precision value. Then we compare the actual precision obtained from 50 CNN training-test runs and the value predicted by (6). The model is accurate, as shown in Fig.1 (right), with a relative estimation error of 3.1%

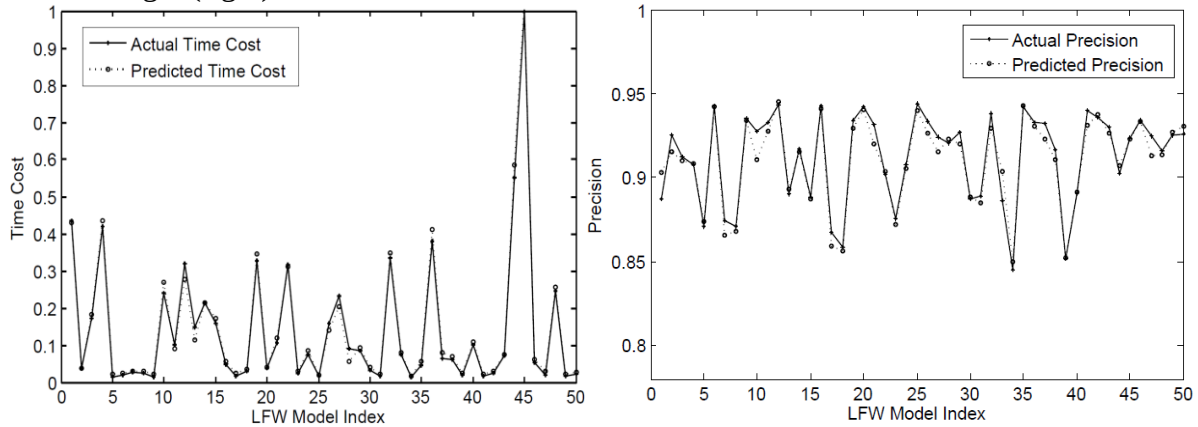

Fig. 1. Evaluating the complexity model (left) and the accuracy of the precision model (right) with 50 random configurations of the control parameters.

3. Optimal complexity control and minimization

The complexity-precision models C(T) and P(T) characterize the behaviors of C and P with respect to the complexity control parameters $\mathrm{T} = (f,s,k,l)$. With these models, we are able to perform optimal complexity control and minimization, which can be formulated as follows

$$\min_{\mathrm{T}} C(\mathrm{T}), \quad s.t. \quad P(\mathrm{T}) \geq P_{\mathrm{T}}, \quad (7)$$

Here, P_{T} is the target precision level. By default, P_{T} should be smaller than the precision of the

baseline DCNN. For example, for the LFW gender classification task, the baseline CNN precision is 0.95. During optimal complexity control and optimization, by allowing a slight degradation of precision, for example, P_T = 0.94 with a precision decrease by 0.01, we aim to reduce the complexity as much as possible. To this end, we can solve the above optimal complexity control problem in (7). We realize that it is difficult to obtain the close-from solution for the problem in (7). Instead, we obtain a number solution for the minimization problem. Specifically, using the mathematical models for P(T) and C(T), we can compute the precision P and complexity C for every configuration of complexity control parameters T =(f, s, k, l). Each parameter ranges from 0 to 1 with a step size of 0.05. As shown in Fig.2 (left), each dot represent the complexity-precision for one configuration of complexity control parameters. Based on these results, we can then find the lower envelope of these data points which represents the minimum complexity C required to achieve the target precision P. This is the optimal complexity-precision curve C(P), which is shown in Fig.2. We can see that, when the precision drops from the original 95% of the baseline DCNN model to 93% by 2%, the complexity is reduced by about 10 times, which is a dramatic saving in the computational complexity.

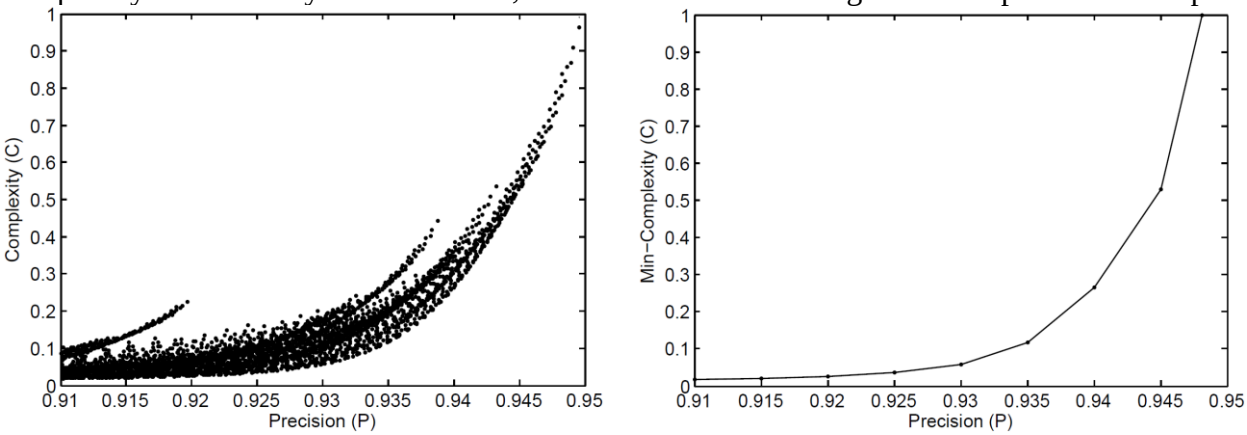

Fig. 2. The relationship between precision and time complexity (left) and the minimal time complexity model at the given precision (right).

4. Conclusion

In this paper, we have established models to understand and characterize the inherent relationship and trade-off between the complexity and classification performance of CNNs. This is one of the first systematic studies to develop mathematically models to understand the complexity-precision behaviors of CNNs. We have successfully developed an exponential-linear model to accurately represent the complexity-prevision behaviors of CNN on LFW dataset. Based on this model, we have developed an optimal computational resource allocation method which is able significantly reduce the computational complexity of CNN at with a minimum degradation of precision performance.

Our experimental results on the LFW benchmark datasets demonstrate that the proposed method is able to reduce the CNN complexity by about 10 times with a small degradation of precision of less than 2%, which is very attractive in practical implementation and use of CNNs attractive in practical implementation and use of the CNNs.

Acknowledgements

This work was supported by National Natural Science Foundation of China under (NSFC) Grant No.61375015, 61340036

References

[1] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[2] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus,and Y. LeCun. Overfeat: Integrated recognition, localizationand detection using convolutional networks. 2014.

[3] R. Rigamonti, A. Sironi, V. Lepetit, and P. Fua. Learning separable filters. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2754–2761.IEEE, 2013.

[4] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition.*arXiv:1409.1556*,2014.

[5] *cuda-convnet, High-performance C++/CUDA implementation of convolutional neural networks*. https://code.google.com/p/cuda-convnet/. Accessed:2015-04-09.