C-007

A Comprehensive Analysis of Low-impact Computations in Deep Learning Applications

Masahiko ATSUMI, Masato INUMA, Lin MENG
Dept. Electronic and Computer Engineering,
Ritsumeikan University, Shiga, Japan

WenWen WANG
Dept. of Computer Science
University of Georgia, USA

***Index Terms*—Deep learning, low-impact computations, Analysis**

I. Introduction

Recently, deep learning techniques have been widely used in many important applications, such as image classification, speech recognition, object detection, detection, to name just a few. At the same time, the accuracy of deep learning models are increasingly improved with various large training data sets and fine-tuned sophisticated network architectures. However the need for significant computation to complete the training and interference task is too high thereby leading to poor performance efficiency. Given this idea, we analyze execute time of each layer of convolution neural network(CNN) model [1], [2]. Furthermore, we focus on the output which updated by $ReLu$ layer and analyze the rate of $Zero$ value from convolution layer. From these analysis, we research redundancy calculation and infer computational volume reduction of CNN.

CNN is one method of Deep learning , generally using convolution layers and pooling layers for extracting features and fully connect layer for judgment. The feature thing in CNN is raised the kernels is multiplied corresponding pixels and the operation is repeated depending on the width of the stride and the size of the image. The experimental condition is under LeNet for the CNN model and MNIST for the data set. For the MNIST data set, 100 images of each number from 0 to 9 is used for the analysis [3].

II. LeNet and the motivation

A. Introduction of LeNet

LeNet was developed to recognize handwritten character in 1990s. The model can classify handwritten number with more than 90% accuracy, and some state-of-the-art CNN model is extended based on the model. Fig. 1 shows the structure which consists of five layers.

First convolution layer ($conv1$) and second convolution layer ($conv2$) have 6 and 16 filters, respectively. There are two max-pooling layer ($pool1$, $pool2$) with 2×2 sized filter at both of layers and followed by convolution layer. Behind $pool2$ layer, there are two fully connected layer, first one ($fc120$) has 120 connects, second one ($fc84$) has 84 connects. $ReLu$ activation is applied as activation function and follows each layer except max-pooling layers.

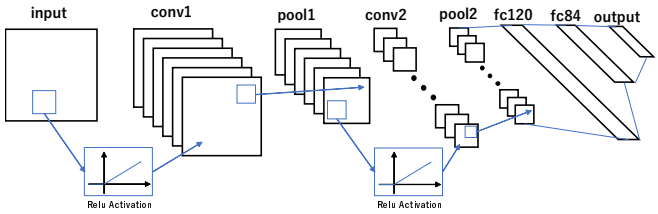

Fig. 1. The structure of LeNet-5

B. Motivation

In state-of-the-art CNN model, $ReLu$ activation is applied and this activation updates all negative results generated by previous layer to $Zero$. Hence, it is inferred redundant computation is involved. Therefore, we analyze the rate of $Zero$ value after $ReLu$ activation and update the results. From this analysis, we can figure out how much redundancy results generated by each layer are involved in CNN. The execution of CNN requires significant computation and large amount of time is consumed. We analyze the execution time of each layer and reveal that which layer has impact for the computation. From the analyses, we examine the possibility to remove the computation which has relatively high impact for execution time and redundancy computation.

III. Experimental Results

A. Analysis of negative value in CNN

Fig. 2 and Fig. 3 show the rate of $Zero$ value created by each filter in each layer. Horizontal-axis indicate the number of images given as input data. The values of vertical-axis indicate the rate of $Zero$ value created by each filter.

Fig. 3 shows the rates of $Zero$ value are different from each filter and especially, the rates are over 50% in filter0 and 3, indicating over half of convolutional computations are updated by $ReLu$ activation. Fig.2 shows the tendency of the rate is same in some of the points. It is different point from $conv1$ layer that the average outputs is over 55% with respect to 35% of $conv1$. Regarding fully connect layer, the average rates of fc120 and fc84 are 47% and 33%.

Focusing on ‘1’ on x-axis, the rates have deviations compared with other numbers. This results are assumed that the similarity of array relate to results of convolutional computation. In the normalized pixel, when we define the part as

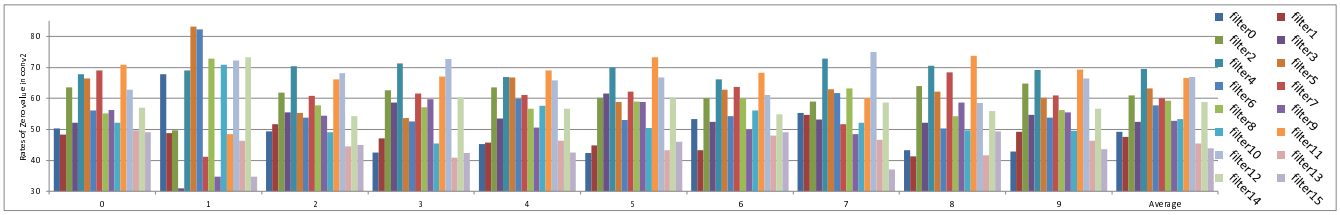

Fig. 2. Rate of Zero value created by each filter in conv2 layer

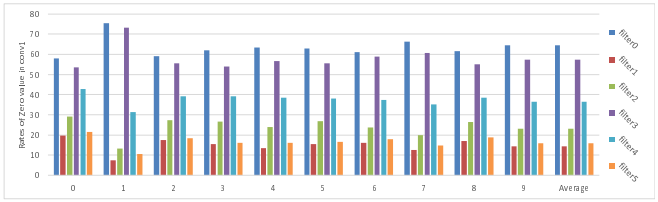

Fig. 3. Rate of Zero value created by each filter in conv1 layer

a value less than or equal to -0.9, and '1' as input data, it indicate that the the part of the percentage is 88% compared to an average of 80% in all of the test data. Given this idea and the weights of kernel are constant in trained network so the results of covolutional calculation have deviations.

In the point of classify accuracy, average outputs in previous layers account for 35% and 55% of in negative value and these outputs have a very low impact on network layers due to updating feature value by the activation.

B. Time consumption of each layer

TableI shows execution times which average 1000 test images. TableI shows highest time consumption is $conv1$ layer, second one is $fc120$ layer and third one is $conv2$ layer.

From these results, it is observed time consumption relate to the of numbers of calculations in a layer. Actually, total numbers of multiplications per one image of $conv1$, $conv2$, $fc120$ are 86,400, 30,720 and 25,600 respectively with 28 pixel of test image. The Equation(1) shows the total numbers of multiplications per input in a covolution layer and that of Equation (2) shows in the case of fully connect layer. 'TM' means Total numbers of multiplications. 'F' means The number of a filter. 'K' means The size of a Kernel. 'W' and 'H' means a height and width of the output result. 'PCN' means The number of connection in previous layer. 'CN' means The number of connection in fully connect layer. Based on these, Time consumption in CNN directory relates to total number of multiplications in a layer. By combining the computation time with the negative values in each layer, it is proved which layer has low impact for classify accuracy so we can infer the computational cost reduction in CNN.

TABLE I
AVERAGE EXECUTE TIME

Layer	execution time(sec)
$conv1$, $conv2$	1.08167E-03; 1.14701E-04
$pool1$, $pool2$	9.82654E-05; 8.97329E-05
$fc120$, $fc84$, $fc10$	3.46992E-04; 9.96735E-05; 8.67207E-05

$$TM = W \times H \times K^2 \times F \quad (1)$$
$$TM = PCN \times CN \quad (2)$$

C. Discuss on calculation reduction

By analyzing time consumption and negative value in CNN, removing the waste of multiple operation in $conv1$ layer leads to increase in performance efficiency because $conv1$ layer is account for largest time consumption in overall run time. In order to achieve this idea, A method that removes the redundant calculations of filter 0 and filter 3, which are relatively large in the rate to the other results, would be more effective. In this case, the idea may remove about 23% toward total multiplication in $conv1$ and overall run time is assumed to decrease by 9.7%. Based on this hypothesis, when we reduce convolutional computation in CNN, rather than reducing the computational complexity of the layers, it is more effective to reduce the computational complexity according to the filters that make up the layers, which is assumed to result in a smaller loss of accuracy.

IV. CONCLUSION

Analyses of rate of $Zero$ value and time consumption reveal redundancy covolutional computation and are assumed computational volume reduction of CNN. Average rates of $Zero$ value in convolution layers are 35% and 55%, For fully connection layers, the average rates are 47% and 33%. it is proved by redundancy computation. Largest time consumption is $conv1$ layer, and this time is about 3 times larger than second one for the reason of the total number of multiplications. By reducing the redundancy, with faster execution speed, it can be led performance efficiency of CNN can be improved.

REFERENCES

[1] V. Akhlaghi, A. Yazdanbakhsh, K. Samad, and R. K. Gupta Hadi, "SnaPEA: Predictive Early Activation for Reducing Computation in Deep Convolutional Neural Networks", ISCA'18, 2018.
[2] B. Akin, Z. A. Chishti, and A. R. Alameldeen,"ZCOMP: Reducing DNN Cross-Layer Memory Footprint Using Vector Extensions", MICRO'52, 2019
[3] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition", PROC OF THE IEEE, 1998