The Sparsity and Activation Analysis of Compressed CNN Networks in a HW CNN Accelerator Model

*Mi-young Lee, Joo-hyun Lee, Jin-kyu Kim, Byung-jo Kim, Ju-yeob Kim
Intelligent SoC Research Division, ICT Materials & Components Research Laboratory
Electronics and Telecommunications Research Institute, ETRI
Daejeon, Republic of Korea
sharav@etri.re.kr

Abstract—**In this paper, we present the result of sparsity increase by CNN compression on 6 representative CNN networks including a famous localization CNN network VGG16-SSD-300. We will also show activation analysis result by applying the compressed CNN networks to a CNN HW accelerator model. Finally, this paper will be ended up with processing time estimation result of the CNN HW accelerator model which reflects the reduced transmission time of sparse weights.**

Keywords; CNN compression

I. INTRODUCTION

CNN (Convolutional Neural Network) is a well-known neural network that has been proven to be successfully used in various fields (image recognition / localization, speech recognition, language translation, etc.). Because CNN processing is based on the kernel multiplications of 3D weights and 3D input features, it has characteristic requiring big computing power and high bandwidth. While in servers without power limitation GPUs are used to implement CNN, power efficient Machine Learning dedicated CPUs or ASICs have been developed for edge mobile platforms. For edge mobile platform, Han. et al. [1] developed a pruning algorithm to increase sparsity and reduce the amount of computations itself. Recent CNN accelerator researches [2] proposed efficient structures using sparsity by skipping zero-valued weights or activations.

II. CNN COMPRESSION AND SPARSITY

A. The Amount of CNN Weights and Activations

TABLE I. THE NUBER OF WEIGHTS AND COMPUTATIONS OF CNN

CNN	*VGG16 [5]*	*SSD [6]*	*ResNet 101[7]*	*GoogLe Net_bn [8]*	*Inceptio n_v3[9]*	*Squeeze Net[10]*
#WGT	138 M	26.3 M	44.4 M	11.2 M	23.8 M	1.2 M
#MAC	15.5 B	31.4 B	7.57 B	2.02 B	5.71 B	0.86 B

Table 1 shows the amount of computation and weights of 6 commonly used CNN networks. For VGG16, the number of weights is over 138 M and it also needs huge computational power of 15.5 B MAC operations. VGG16-SSD-300 (SSD) network has fewer weights than VGG16 of 26.3 M, but input feature size of that network is 300 larger than 224 of VGG16, so that network needs much more computational power of 31.4 B than VGG16. Although there is a difference depending on a neural network, it can be seen that the amount of computation is enormous over 1 billion MACs.

B. Network Compression Results

In order to reduce the computational complexity, CNN compression was applied. We used a pruning algorithm from previous research [1]. First, the sensitivity analysis step was performed to find a pruning threshold of each individual layer of a neural network. It is a process of finding the threshold that a layer can be pruned without degradation of performance while changing only the pruning threshold of the layer. Pruning process was performed with the pruning thresholds and neural network was re-trained.

The quantization process is then performed to further increase compression rate. We used a quantization algorithm which proposed by Gysel. Et al. [3] named "Dynamic Fixed Point Quantization". For all target networks, quantization is applied from 32-bit floating weights to 8 bit fixed weights. We used the Caffe framework [4] to compress CNN networks by adding sensitivity analysis, pruning, quantization methods.

TABLE II. CNN COMPRESSION RESULTS

CNN	*BTop1*	*BTop5*	*Top1*	*Top5*	*%W_Z*	*CR*
VGG16	68.5	88.7	64.7	86.7	7.22	27.2
SSD	77.6 (mAP)		72.5 (mAP)		26.64	7.5
ResNet101	74.4	91.9	68.5	88.6	39.08	5.1
GoogLeNet_ bn	68.9	89.1	64.5	86.8	29.75	6.7
Inception_v3	78.0	94.1	66.3	87.1	36.32	5.5
SqueezeNet	57.5	80.3	56.6	79.6	64.06	3.1

Table 2 shows compression results of 6 CNN networks. BTop1, BTop5 are baseline top1 and top5 accuracy and Top1, Top5 are top1, top5 accuracy of compressed networks. %W_Z is the percentage value of the final survived weights. We use an 8-bit sparse index that represents the position in the original weight 3D data. Zeros are added if a sparse index exceeds the max value (255) that can be represented by 8 bits. %W_Z

means final sparse weight percent including added these zeros. CR means compression rate and used equation is as (1). (WBW: weight bit-width (8), SIBW: sparse index bit-width (8))

$$CR = 100 / (\%W\_Z * (WBW + SIBW) / 32)) \quad (1)$$

The performance degradation is 0.7 ~ 7% compared to the Top5 baseline accuracy, which is due to short learning time of less than 10 epochs. If re-training is performed for a sufficiently long period of time, accuracy will reach baseline. In all networks except SqueezeNet, %W_Z is 7.22 ~39.08 which means that more than half of weights were eliminated. For SqueezeNet, because it is developed to minimize redundant weights, compression rate is not high like others. Finally, the compression rate CR is 3 to 27 times compared to original CNN networks despite the addition of sparse indexes.

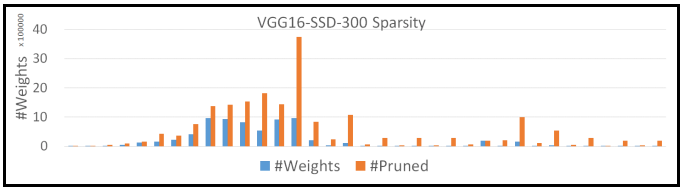

Figure 1 VGG16-SSD-300 Sparsity

Figure 1 shows sparsity of each layer in VGG16-SSD-300. In the layer having big kernel size in the middle of Figure 1, sparsity is high and this can lead to a large reduction of computational load.

C. *Activation Analysis of Compressed Network*

Table 3 shows the activation results obtained by applying compressed networks to a HW CNN accelerator model. The model is a behavioral model of a HW CNN accelerator written in C++. It includes network file parsing, HW configuration, weight transmission, In/Out feature transmission, HW CNN accelerator behavioral model which is performed according to tiling schedule. A tile is a part of layer processing. We put monitoring functions to summarize activation percentage while processing compressed CNNs. Activation is defined the percentage of MAC operations which output non-zero valued results among total MAC operations.

TABLE III. ACTIVATIONS ANSYSIS RESULTS OF COMPRESSED NETWORKS

CNN	*VGG16*	*SSD*	*ResNet 101*	*GoogLe Net_bn*	*Inceptio n_v3*	*Squeeze Net*
%ACT	17.2	12.1	14.7	22.9	21.7	28.3

Table 3 shows the activation results are all under 30%. Especially for SSD which has a huge computational load of over 30 B MAC, activation percentage is 12.1 % which means 87.9 % MAC operations can be skipped if efficient HW scheme filter out unnecessary operations using sparsity and activation properties.

Figure 2 is the activation percentage of each layer in VGG16-SSD-300. Layers of big computation load result low activation under 30% which can be exploited to speed up and save energy by gating redundant MAC operations.

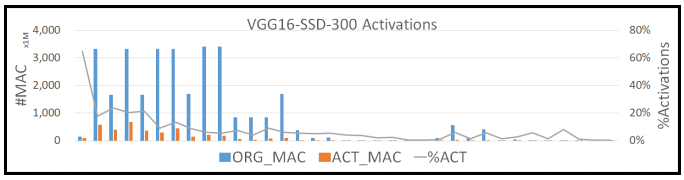

Figure 2. VGG16-SSD-300 Activations

III. Conclusion

Table 4 shows the processing time estimation result of the HW CNN accelerator model. In the model we estimate processing time based transmission data size (weights, I/O features) and DMAC transmission data rate (2.5 G bps) because we expect transmission time is dominant and kernel processing can be pipelined with transmission time. In transmission time in model, HW configurations, weight (dense/sparse) transmission, I/O feature transmission are all included. In all networks, processing time saving of sparse cases can be confirmed. But for some CNNs, processing time saving is not much as expected, it's because the model only exploits sparse property not for zero activations, this means processing time result can be further improved by eliminating unnecessary zero-valued activation transmissions.

TABLE IV. THE PROCESSING TIME ESTIMATION RESULTS

CNN	*VGG16*	*SSD*	*ResNet 101*	*GoogLe Net_bn*	*Inceptio n_v3*	*Squeeze Net*
Dense (ms)	152.8	177.6	133.8	37.8	94.0	19.4
Sparse (ms)	64.0	136.2	126.4	34.5	83.1	19.7

Acknowledgment

This work was supported by the ICT R&D program of MSIT/IITP [2016-0-00098, Brain-Inspired Neuromorphic Perception and Learning Processor].

References

[1] Han, Song, et al. "Learning both weights and connections for efficient neural network." Advances in neural information processing systems. 2015.

[2] Parashar, Angshuman, et al. "Scnn: An accelerator for compressed-sparse convolutional neural networks." 2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA). IEEE, 2017.

[3] Gysel, Philipp, Mohammad Motamedi, and Soheil Ghiasi. "Hardware-oriented approximation of convolutional neural networks." arXiv preprint arXiv:1604.03168 (2016).

[4] https://caffe.berkeleyvision.org/

[5] https://github.com/BVLC/caffe/wiki/Model-Zoo

[6] https://github.com/weiliu89/caffe/tree/ssd

[7] https://github.com/KaimingHe/deep-residual-networks

[8] https://github.com/lim0606/caffe-googlenet-bn

[9] https://github.com/soeaver/caffe-mode

[10] https://github.com/DeepScale/SqueezeNe