HARDWARE ACCELERATORS FOR NEURAL NETWORKS

SEMINAR REPORT

Submitted in partial fulfillment of the requirements
for the degree of

Master of Technology

by
GOGIREDDY SASI KIRAN REDDY
(Roll No. 213079025)

Under the guidance of
Prof. Virendra Singh

Department of Electrical Engineering
Indian Institute of Technology Bombay
October 2022

Acknowledgement

I express my gratitude to my guide Prof. Virendra Singh for providing me the opportunity to work on this topic.

Gogireddy Sasi Kiran Reddy
Electrical Engineering
IIT Bombay

Abstract

Nowadays deep convolutional neural networks are most widely used in the field of machine learning. To classify an image we need to perform millions of MAC operations. So in order to reduce the computation,several hardware accelerators have been proposed.

Contents

List of Figures

Chapter 1

1. Introduction

Deep convolutional neural networks are among the most used in the machine learning field.CNNs require an ample number of computations to label one image i.e. to which class it belongs. This paper introduces a novel technique that cuts the computation based on certain parameters. Each convolution operation is commonly followed by an activation function called a Rectifying Linear Unit (ReLU) that returns zero for negative inputs and yields the input itself for the positive ones. We observed that almost 42% to 68% are zero which comes from the ReLU outputs.

Chapter 2

Literature Survey

Snapea provides an insight that the amount of computation in CNNs can be significantly reduced by using a combination of runtime information along with the algorithmic structure of CNNs, which feeds many negative inputs to the activation function. In addition, SnaPEA comes with a the predictive mode that speculates on the outcome of sign-check and terminates the computation even earlier, trading off accuracy for less computation. They developed a multi-variable optimization algorithm that changes the threshold value based on the classification accuracy achieved by the snapea architecture. The threshold becomes a knob for controlling the accuracy-computation tradeoff. In the exact mode, which has no effect on the classification accuracy, SnaPEA, on average, delivers 28% (maximum of 74%) speedup and 16% (maximum of 51%) energy reduction over the EYERISS [2].With a 3% loss in classification accuracy, on average, 67.8% of the convolutional layers can operate in the predictive mode. The average speedup and energy saving of the layers in the predictive mode over EYERISS are 2.02× and 1.89×, respectively. GoogLeNet sees the maximum benefit of 3.59× speedup and 3.14× energy reduction. In the exact mode, SqueezeNet achieves 30% speedup and 15% energy reductions with no loss of accuracy.

Chapter 3

Review

3.1 SnaPEA: Predictive Early Activation for Reducing Computation in Deep Convolutional Neural Networks

3.1.1 Summary

SNAPEA HARDWARE-SOFTWARE SOLUTION

Snapea provides a hardware-software solution to reduce the computation of CNN. The software part consists of two distinct phases i.e. exact mode and the predictive mode.This is illustrated in Figure [1]. In the exact mode, they extracted kernels from the architecture and weights are re-ordered according to the sign-based in such a way that positive weights are first followed by negative weights for all the kernels in the CNN architecture. In the predictive mode, snapea will further reduce the computation before starting the convolution with the negative weights. The intuition is that after performing a predetermined number of operations, it checks with the threshold and if the partial output is less than the threshold value then the output is negative but it predictive mode there might be miss speculation. There are two speculation parameters. They are:
1. Threshold value.
2. Number of a pre-determined number of MAC operations.
These two speculation parameters need to be determined for as many layers as possible to maximize the benefits. The algorithm is run by the software part on the Optimization Dataset through the following three passes. First is kernel profiling which assigns speculation parameters for each kernel and then the local optimization pass which picks a set of speculation parameters for each layer and then the final pass is the global optimization pass which picks a set of speculation parameters for the whole network. This pass reorders the kernel weights by placing the ones determined by the speculation parameters ahead of the others. Then, the remaining weights are reordered based on the same procedure used for the Sign-Based Weight Reordering pass, which puts the negative weights after the positive ones. Finally, these reordered weights determine the execution of the CNN on the SnaPEA hardware.

Kernel Profiling

In kernel profiling, for each layer in CNN and in each layer, they took each kernel for a set of threshold values and a pre-determined number of MAC operations randomly and then simulate the CNN for that kernel, and if the error is less than the acceptable loss

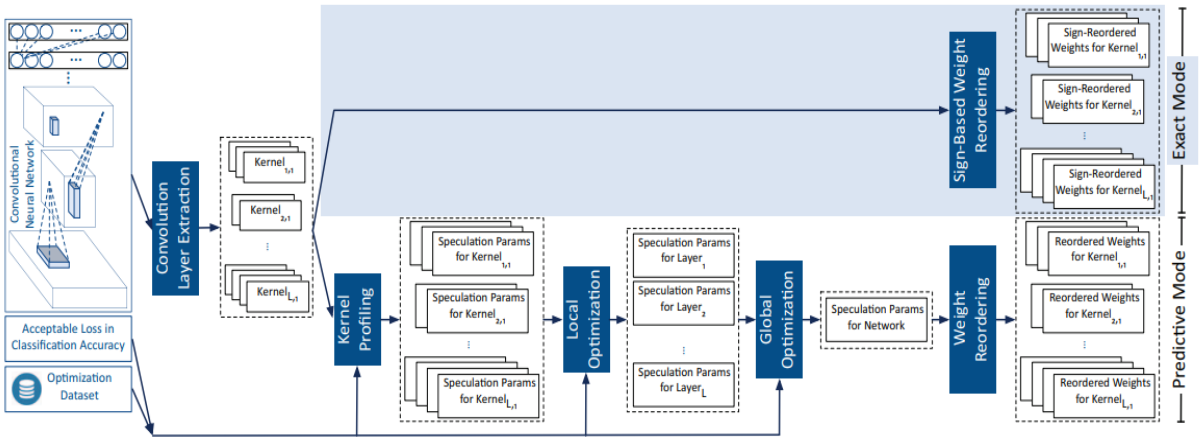

Figure 3.1: Software Workflow.

in the classification accuracy then append those speculation parameters in the ParamK list. So in this way, they got for each kernel there will be multiple speculation parameters.

Local optimization pass and Global optimization pass

The next pass is the local optimization pass in which they further filtered speculation parameters for each layer and next is the global optimization pass in which they further filtered and then find speculation parameters for the whole CNN network. If the obtained error is more than the acceptable loss in the classification accuracy then adjust the parameters accordingly and append the best suitable parameters for the network.

Snapea Hardware architecture

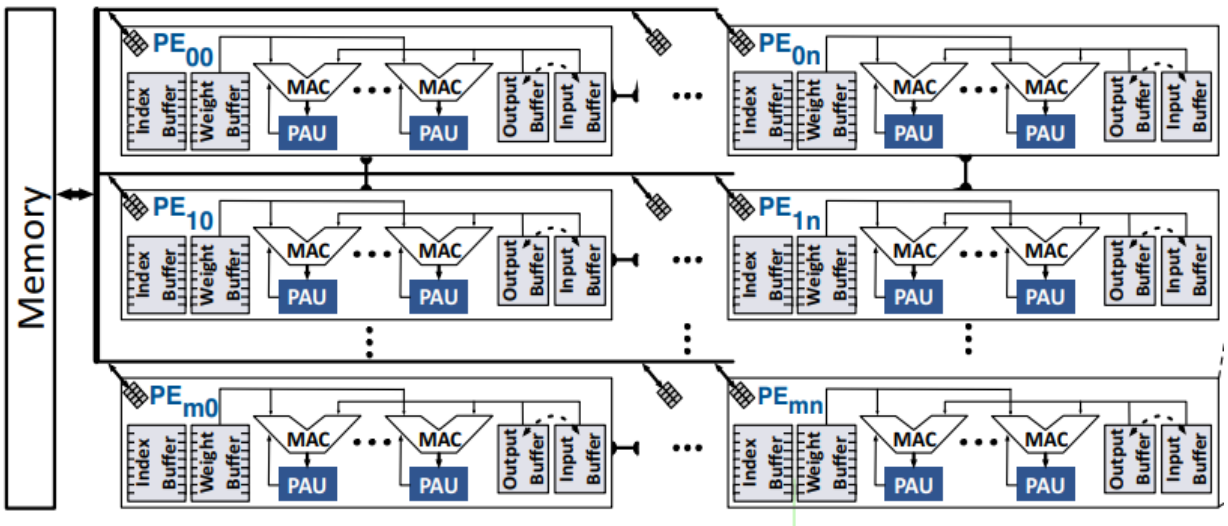

Figure 3.2: Snapea architecture.

Snapea provides a hardware architecture in an efficient way in such a way that to efficiently execute the CNN with the transformed convolution operations. Modern CNNs consist of several back-to-back layers including convolution, ReLU activation, pooling,

and fully-connected layers. Snapea architecture consists of an 8*8 array of processing elements in which each PE is equipped with an input and output buffer that communicates with the off-chip memory.

Each processing element comprises an index buffer, weight buffer, and input and output buffer and the number of computing lanes are four for each compute lane there is a predictive activation unit(PAU) that controls based on the threshold value and the number of pre-determined operations. That is, each MAC unit performs the multiplication of one input and weight for each convolution window and sends the results to the accumulation register. The accumulation register accumulates the partial sums for each convolution window.

Weight and Index Buffer

The weight buffer contains the values of the weights in a pre-determined order where in exact mode the weights are arranged according to the sign of the weights i.e. positive weights are first and followed by the negative weights. Whereas in predictive mode the speculation weights are kept first and then the positive weights are followed by the negative weights. The index buffer is used to load the index values of the weights because of the re-arrangement of weights in the weight buffer.

Input and Output Buffers

The input buffer holds the portion of the input for each convolution layer and based upon the index values with respect to weights the inputs are fetched from the input buffer and then given to the MAC units. The resulting corresponding output is stored in the output buffer and is given as input to the next convolution layer.

Predictive Activation Unit

One PAU is attached with one compute lane to support the convolution operation in the exact mode and in the predictive mode. In exact mode, the PAU only needs to check the sign of the partial sum and if the partial sum is negative, the PAU directly terminates the operation otherwise it will continue the operation. In the predictive mode, the PAU checks with the threshold value after a pre-determined number of operations and if the partial sum is less than the threshold value then the PAU automatically terminates the operation otherwise it will continue the operation.

Experimentation Results

SnaPEA, on average, delivers 1.3× speedup and 1.16× energy reductions over EYERISS, respectively. Even for SqueezeNet [6]—a statically pruned convolutional neural network—SnaPEA yields 1.3× and 1.14×. In predictive mode, on average, 67.8% of the convolutional layers operate in the predictive mode, and the average speedup and energy saving across these layers are 2.02× and 1.89×, respectively.

3.1.2 Strengths

The strengths of snapea are the two modes i.e. exact mode and the predictive mode. In the exact mode if it detects a negative partial output then it cuts the computation then and there and the output is directly zero. So in this way the maximum number of computations in the exact mode are decreased compared to the unaltered mode number of computations.

In the predictive mode the number of computations are even further decreased than the exact mode due to the speculation parameters are selected across the network due to greedy software algorithm.Due to this the speedup is increased compared to the eyeriss architecture and the exact mode. And also the energy consumption is also reduced compared to the eyeriss architecture and to the exact mode.If we keep relaxation in loss in the classification then the speedup is increasing in the predictive mode.

3.1.3 Weakness

The weakness of the snapea architecture is in the software algorithm that to in predictive mode i.e in the predictive mode if the partial output is predicted as negative due to the speculation parameters then there is a loss in the classification accuracy. And also due to rearranging the order of weights, they need to keep a buffer to store the indices of the weights, this increases the on-chip size.

3.1.4 Opportunities

The software algorithm of snapea can be optimized still further.

Chapter 4

Conclusion

Due to an increase in the convolutional layers and the number of computations to classify an image, different accelerators have proposed different architectures to reduce the number of computations. In this paper, Snapea outperformed very well in comparison with the eyeriss architecture.

References

[1] V. Akhlaghi, A. Yazdanbakhsh, K. Samadi, R. K. Gupta, and H. Esmaeilzadeh, "Snapea: Predictive early activation for reducing computation in deep convolutional neural networks," in *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*, pp. 662–673, IEEE, 2018.