A Study on Optimization of
Residual Binarized Neural Network

1910260 CHEN Yan

Convolutional Neural Networks show their high ability in image classification and are widely used in many fields in recent years. Modern CNNs models contain millions of parameters and require billions of floating-point operations to infer an image. Lightweight CNNs models such as MobileNet and ShuffleNet were proposed to enable CNNs to run on mobile devices. On the other hand, Binarized Neural Networks for hardware with higher speed and lower power consumption was also introduced. BNN is a technology that realizes speedup and weight reduction by converting the conventional CNN floating-point matrix operation to a binary (-1 and $+1$) bit XNOR operation at the expense of some accuracy. It is suitable for BNNs to run inference on specific hardware accelerators in, for example, FPGA devices. Residual Binarized Neural Network (ReBNet) greatly improves accuracy by introducing binarize factor and performing multi-level binarization. Since the binarize factors in ReBNet are decimals, it is necessary to multiply the fixed-point numbers by DSP in the FPGAs. DSPs are a scarce resource in today's FPGAs, so their degree of parallelism is limited.

In this thesis, we introduce the basic knowledge of CNNs and BNNs and recent related work. And we propose a state-of-the-art method to accelerate and optimize the ReBNet by replacing fixed-point number multiplication with logical shift operation. We designed an end-to-end framework for training Binarized Neural Networks, on which the conversion to logical-shift-based multiplication on software and hardware accelerators implemented on FPGA is performed. We propose Isometric Residual-Binarization, which reduces the elements of binarize factor from the number of levels to 1, and reuses the single element to express binarize factor vector of multiple levels. Like ReBNet, this binarize factor can be determined through training. Then, we show how to transform the parameters in the convolutional layer, fully connected layer, and batch normalization layer, so that the binarize factors become integers. Benefiting from this, no more DSPs are required to multiply the binarize factors, and a large quantity of hardware recourse can be saved. We redesign processing elements (PEs):

- only 1 DSP and 1 accumulator are required per PE,
- encoder becomes much simpler than that in ReBNet,
- it can compute multiple levels at the same time but requires more popcount modules.

We also apply throughput optimization which makes Initiation Interval from the number of levels to 1 in our design. However, this optimization causes the data stream to become wider and logic between layers to become larger. We propose Adaptive Bit Width to resolve these problems without any performance degradation.

We implement our design and compare it with baseline research in different settings. First, we train BNNs models with Isometric Residual-Binarization we proposed on multiple datasets in 3 neutral network architectures. Compared with the original work, ours has fewer parameters but reaches similar accuracy. Then, we transform parameters in models and evaluate the accuracy changes. The accuracy does not change obviously. Next, we make the testbenches and simulate the behavior of hardware accelerators, and start implementation . We try to find the possibly highest degree of parallelism for 2 levels and 3 levels on the resource-limited device for architectures of small datasets, and our design achieves 8 times higher throughput than ReBNet on average. We also implement the maximum parallelism of architectures of small datasets on the large device to find out the hardware resource usage in highly parallel. For the model of the large dataset which requires a huge amount of on-chip memory, we implement it on the Virtex UltraScale device. After that, we measure the scenario of the resource-limited device on the development kit. We design the software evaluation programs and measure the accuracy, throughput, and power usage. All of the results are the same with software models, and the throughput is very close to the theoretical values. We compare these with GPU implementations. At last, we analyze the effect of each optimization quantitatively. Hardware resource usage of each PE is much lower than ReBNet when input data width is less than or equal to 32. Our Data Stream with Adaptive Bit Width can reduce up to $\frac{7}{8}$ Block RAMs in the buffer of sampling modules of convolutional layers.

Finally, we conclude this thesis and describe the future works.