Conversion of Convolutional Neural Networks (CNNs) into Logic Flows for Efficient Execution on RISC-V CPUs

When CNNs are deployed on edge devices, often a huge number of dedicated hardware multiply-accumulate (MAC) units are not available to process massive MAC operations in CNNs. Instead, CPUs exist in nearly all these devices. CPUs themselves are not good at executing such mathematical operations on a large scale, since they opt more to execute control flow logic. To execute CNNs on CPUs efficiently, it is critical to convert their MAC operations into logic flows. In this master thesis, the execution of a convolutional neural network (CNN) will be converted to logic flows, so that it can be executed with low latency and low energy on CPUs.

The concept above is illustrated in Figure 1, where a fully-connected neural network is used as an example. To convert the network execution into a logic flow, it is converted into an equivalent decision tree, where the input condition *I* making y_0 larger than 0 with ReLU as the activation is set as the *root* of the tree. To trigger y_3 and y_4, y_1 should be processed. The input condition making y_1 larger than 0 is then constructed as *decision nodes*, which are connected with the root via two branches representing the *true (T)* and *false (F)* decision of the root. Similarly, the remaining neurons can be processed to create more decision nodes. After redundant and invalid nodes are removed, the decision tree can be represented as a logic flow, as shown in Figure 1(c), and compiled for CPU execution.

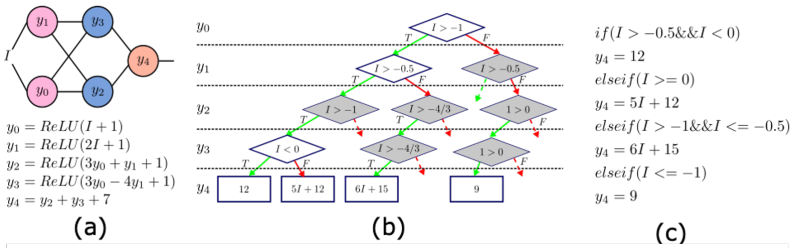

Figure 1: Converting a neural network into an equivalent decision tree. (a) Neural network with ReLU activation functions. (b) Equivalent decision tree of the neural network. (c) Converted logic flow derived from the decision tree.

For CNNs, consecutive convolutional layers will be fused into fully-connected layers, which can be converted into logic flows with the method above. Specially, a CNN first needs to be converted to a decision tree. To avoid exponential growth of the tree, training data is used to only convert the relevant paths. Then either a SAT/SMT or ILP solver is used to determine which ReLU decisions are redundant. For this step it might be necessary to use layer fusion and convert parts of the convolutional layers to fully connected layers. The most used paths of the Decision tree are then kept and combined with the original model to form the hybrid model. At the end, C-Code is generated to run the hybrid model on a RISC-V Simulator, as shown in Figure 2.

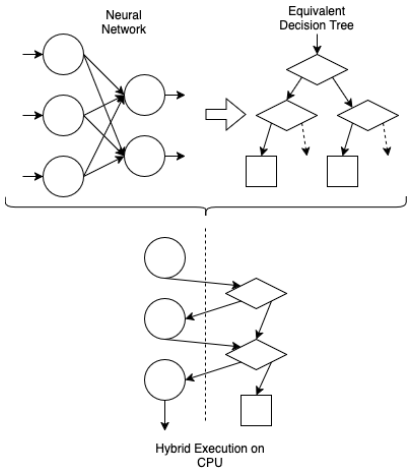

Figure 1: The concept of hybrid execution.

If you are interested in this topic for master thesis, please contact:

Prof. Dr.-Ing. Li Zhang (grace.zhang@tu-darmstadt.de) with your CV and transcripts.