EECE 4632: Spring 2022
Hardware-Software Codesign for FPGA-Based Systems
Project Proposal

Oscar Kellner

Convolution neural networks (CNNs) are a type of neural network that excel in processing multidimensional data and have significant applications in computer vision. A CNN is composed of multiple layers, including convolutional layers, pooling layers, and fully-connected layers. With each layer, more complex and larger features of the input data are recognized, until at the end of the network where the pattern would be predicted. This project intends to design a hardware implementation of part of a convolution neural network, specifically the computations within a convolutional layer (known as a "convolution accelerator").

Because CNNs are ideal for processing images, this project will use images as its input data to evaluate the functionality of the project. The output of a convolutional layer is known as a feature map, a multidimensional set of data that is often the input for a pooling layer before being processed through another convolution layer. We can determine if the hardware design is working correctly if an image input outputs the same exact feature map in both the software and hardware implementations, given the arguments within the algorithms are exactly the same. The output registers from the hardware design can be read from and evaluated using the interactive python hardware overlay features.

In a convolution layer, a "convolution operation" is applied to the input data, with the output feature map being passed into the next layer (normally a pooling layer). This operation involves applying a series of smaller image filters (convolution kernels) in order to extract data from the image, such as edges or color patterns. Multiple filters can be used to produce multiple feature maps. Each filter consists of a smaller matrix that is multiplied element-wise with the input matrix (the image) and summed to produce an output, and many of these outputs are what produces the final feature map for the given convolution layer. There are a couple of hyperparameters (kernel size, padding, number of filters, etc) that can adjust how the CNN 'learns,' which will need to be chosen beforehand when evaluating the performance of the design.

The project will base its code on Darknet. Darknet is a machine learning framework written in C, which makes it ideal for a hardware implementation as C is a suitable language for High Level Synthesis (HLS). The convolution layer algorithm will be isolated and implemented in a hardware overlay that can be used on a Pynq FPGA board.

Given the potential complexity of this project, the primary goal is to first implement only a single convolution layer with grayscale images as inputs to reduce the complexity. This will be done by spending extensive amounts of time analyzing the source code of the Darknet framework to see what portions of the code can be isolated for within the scope of the project, and what hardware optimizations can be made in order to get the best performance using Vivado HLS. If time allows it, the plan is to add additional convolution layers to the model and observe the difference in performance as more layers in a full CNN are converted to their hardware counterpart.

Ideally, by the March 3 deadline a software implementation of a convolution layer from either the Darknet API or an alternative machine learning framework (e.g. PyTorch) would be demonstrated on the Pynq FPGA board. This software implementation would be later used as a point of comparison when evaluating the difference in performance between the software and hardware implementations of the convolution layer algorithm.

References:

https://www.ibm.com/cloud/learn/convolutional-neural-networks
https://pjreddie.com/darknet/
https://medium.com/analytics-vidhya/image-classification-with-convolutional-neural-networks-ac14a978f0fa