Aplicação de uma Rede Neural Convolucional em Tecnologia FPGA

Ricardo Di Curzio Lera[1], Prof. Dr. Reinaldo Augusto da Costa Bianchi[2]
[1,2] Centro Universitário FEI
ricardodclera@mail.com / rbianchi@fei.edu.br

Resumo: Redes Neurais Convolucionais são uma das mais eficientes aplicações de Deep Learning para reconhecimento de imagens. Implementar esta tecnologia em um Field Programmable Gate Array (FPGA) traz muitas vantagens como processamento em paralelo, mas acarreta seus próprios desafios. Este projeto visa solucionar estes desafios e avaliar a viabilidade desta implementação. Para tal, diferentes redes implementadas em nível modelar foram testadas e eventuais dificuldades foram analisadas e resolvidas até a síntese final da rede em tecnologia FPGA. Concluímos que esta tecnologia é adequada para a implementação de CNNs.

1. Introdução

A necessidade de se implementar algoritmos complexos de análise de imagem em FPGAs se deu inicialmente a partir de um esforço internacional para o Phase-II upgrade do colisor de partículas LHC, no centro de pesquisa nuclear do CERN.

A tecnologia FPGA é utilizada nestes experimentos devido à sua capacidade de processamento em paralelo e sua natureza reprogramável. Sua natureza única, porém, traz várias dificuldades na sua implementação.

A FPGA possui uma arquitetura dinâmica que, ao invés de executar um programa nela definido, permite ao usuário descrever um circuito lógico qualquer e este será "escrito" fisicamente através de portões lógicos. Para isto, se utiliza uma linguagem não-sequencial de baixo nível denominada Hardware Description Language (HDL).

Redes Neurais, essencialmente, são algoritmos capazes de reconhecer padrões e se modificar apropriadamente com base na resposta desejada [1]. Elas se compõem de "camadas" que servem diferentes propósitos no algoritmo. Uma Rede Neural Convolucional (CNN) é uma rede neural que possui uma ou mais camadas que utilizam a operação de *convolução*. Esta operação descreve o modo como uma função é modificada por uma segunda, e é utilizada para descrever *filtros* para o reconhecimento de padrões em imagens [2].

O objetivo deste projeto foi implementar uma rede convolucional primeiramente em *nível modelar*, isto é, no mais alto nível no qual é possível trabalhar com suas camadas diretamente, e depois passar esta rede por uma série de processos nos quais resultarão em uma rede funcional descrita em HLD para ser implementada em tecnologia FPGA.

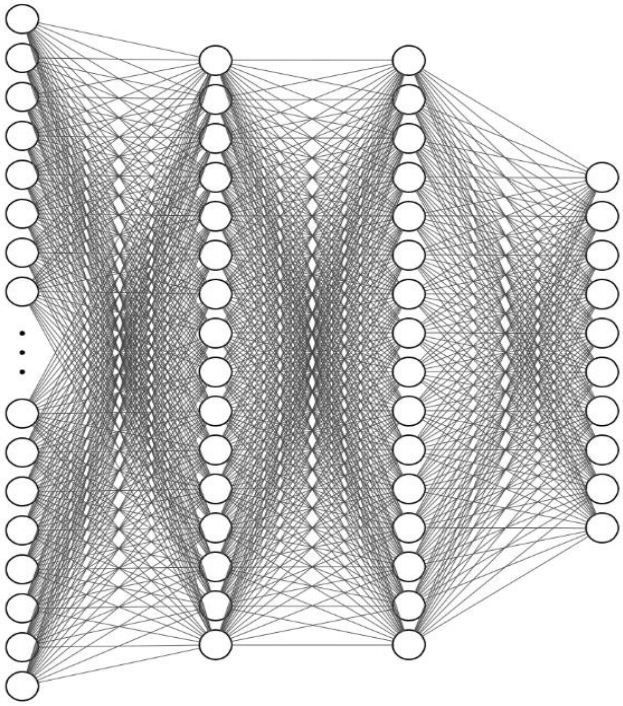

Figura 1 – Exemplo de uma rede neural do tipo *fully connected*, também chamada de *Multilayer Perceptron*

2. Metodologia

A rede inicial utilizada é uma rede convolucional simplificada de 3 camadas treinada para reconhecimento de dígitos escritos à mão em imagens 8x8. Esta rede foi escrita em Keras, um *framework* que utiliza linguagem Python para descrever redes neurais em nível modelar.

Desta rede foram retirados dois arquivos chave: seu modelo e seus parâmetros, para serem processados em um programa criado pelo CERN em parceria com outras instituições de ensino com o propósito de auxiliar nesta implementação, o *hls4ml* [3].

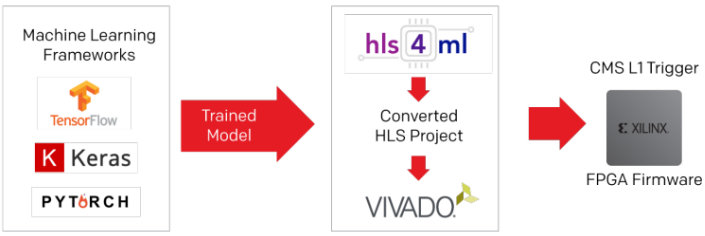

Figura 2 – Fluxograma de operação do hls4ml

Este programa reescreve a rede em linguagem C++ e cria um arquivo *build_prj.tcl* que pode ser executado pelo *Vivado High Level Synthesis*, um programa desenvolvido pela Xilinx capaz de sintetizar linguagem de descrição de software a partir de um programa em linguagem C++.

Com o HDL sintetizado, ele é interpretado pela FPGA e a rede é gerada em sua arquitetura. A imagem a ser analisada é inserida serialmente na FPGA e é tratada por um módulo separado que a transforma uma variável de 1024 bits para ser interpretada pela rede. Este número depende no tamanho da imagem e a precisão utilizada para cada um de seus pixels.

Depois do processamento a rede retorna uma *bitstream* que representa o quanto a rede associa a imagem de entrada com cada um dos dez dígitos de 0 a 9 e aquele com o maior valor é escolhido como resposta.

3. Testes e Resultados

Como o hls4ml ainda é um projeto em desenvolvimento, vários erros e outras dificuldades foram encontrados e tiveram de ser remediados. Em particular, o Vivado HLS possui funções de otimização que não se aplicam bem aos processos requeridos pela rede neural.

O processo de *Pipelining* por exemplo, tenta reduzir o tempo necessário para se processar uma dada função para um dado número de ciclos de *clock*. Quando isso é aplicado à função de multiplicação de convolução, que se caracteriza por 6 funções 'for' encavaladas, o programa resultante se tornaria grande demais e a síntese é abortada automaticamente.

Estas funções são *pragmas*, ou funções pragmáticas. Para evitar este problema, algumas destes pragmas de otimização tiveram de ser desabilitados, incluindo: *Dependence*, *Dataflow*, *Latency*, *Occurence*, *Pipeline* e *Unroll* em todos os headers e programas utilizados, sendo que estas funções interagem com *loops* extensos de acordo com o manual oficial do Vivado HLS [4].

Após remediados estes problemas, foi possível simular a rede previamente especificada convertida para VHDL.

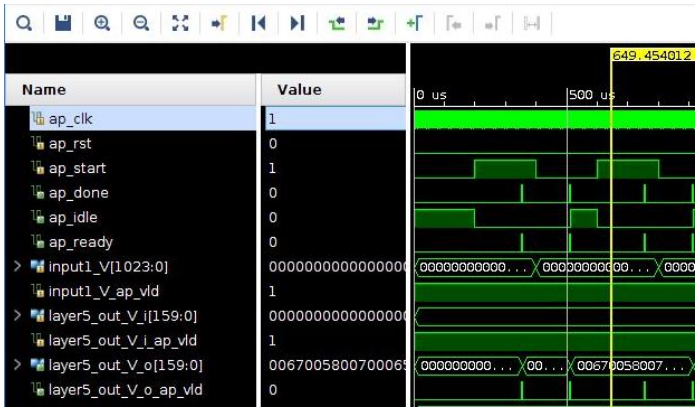

Figura 3 – Simulação da rede em Vidado

A figura acima demonstra o avanço dos estados da rede, resultando em uma saída diferente dependendo de sua entrada. A entrada representa a imagem 8x8, cada pixel com 16 bits, resultando num total de 1024 bits. A saída possui 10 diferentes respostas, uma para cada dígito reconhecível pela rede, também cada resposta com 16 bits representando o quanto a rede assimilou a entrada com cada dígito, e resultando num total de 160 bits.

4. Conclusões

Embora alguns dos programas utilizados ainda devem ser refinados por seus respectivos desenvolvedores, eles representam um avanço importante na facilitação de implementação de redes neurais como um todo e na utilização de FPGAs.

Este projeto contribui a este refinamento implementando pela primeira vez uma rede neural convolucional construída em nível modelar em tecnologia FPGA.

5. Referências

[1] HECHT-NIELSEN, Robert. III.3 - Theory of the Backpropagation Neural Network. *In*: HECHT-NIELSEN, ROBERT. **Neural Networks for Perception**. HNC, Inc. and University of California, San Diego: Elsevier Inc., 1992.

[2] BETTONI, Marco *et al.* **A Convolutional Neural Network Fully Implemented on FPGA for Embedded Platforms**. New Generation of CAS (NGCAS), 2017.

[3] DUARTE, Javier *et al.* **Fast inference of deep neural networks in FPGAs for particle physics.** JINST, FERMILAB, 2018.

[4] XILINX™, **Vivado Design Suite User Guide – High-Level Synthesis**, 2014. Disponível em: https://www.xilinx.com/support/documentation/sw_manuals/xilinx2014_1/ug902-vivado-high-level-synthesis.pdf. Acesso em: 14 fev. 2019.

[1] Aluno de IC do Centro Universitário FEI. Projeto com vigência de 09/18 a 08/19.