LeNet-5 Models for Handwriting Recognition Based on Pynq-Z1

Team Member: Longhao Tan (pzp8vh), Mohamed Tajudeen Sunaideen (xcu7gp)

1 Description

This project focuses on training and quantizing the LeNet-5 CNN deep learning model and deploying it on the PYNQ-Z1 platform utilizing its FPGA capabilities to achieve efficient hardware-accelerated inference for edge AI applications. This project involves testing and comparing inference accuracy and execution time on the computer CPU, the processor of PYNQ-Z1, and the PYNQ-Z1 with FPGA acceleration.

2 Motivation

Currently, the research and development of embedded devices have attracted growing attention. With the rapid progress of AI technology, edge AI systems are increasingly deployed in embedded systems to tackle complex tasks such as autonomous driving. These applications require both high accuracy and fast inference. However, the resource constraints of edge devices limit the performance of deep learning models. Motivated by this challenge, we aim to investigate whether deep learning models deployed on edge devices can achieve both fast and accurate inference, thereby assessing their viability for deployment on resource-constrained systems.

LeNet-5 model is a foundational convolutional neural network (CNN), including both convolutional and fully connected layers, which is designed primarily for handwriting recognition tasks. Running the LeNet-5 model on edge devices can demonstrate the potential of deploying deep learning model on resource-limited devices. Leveraging the FPGA capabilities of the PYNQ-Z1 enables full utilization of hardware acceleration, enhancing performance and scalability for AI applications. This project will present a practical case for using FPGA to accelerate machine learning model on edge devices, integrating software and hardware to optimize embedded AI systems.

3 Objectives

- Model Optimization and Deployment: Train and quantize LeNet-5 model with MNIST dataset which contains several images of handwritten digits (0–9) to fit the constraints of the PYNQ-Z1 and deploy the optimized model to demonstrate that deep learning models have the potential to be deployed on resource-constrained devices.
- FPGA Acceleration Design: Utilizing Vivado HLS, a Xilinx tool to design digital circuits for FPGA, implement a hardware pipeline that accelerates the inference of LeNet-5.
- Verification: Evaluate the inference accuracy and execution time on the computer CPU, the processor of PYNQ-Z1 and the PYNQ-Z1 with FPGA acceleration.

4 Instruction

4.1 On Computer

1. Download and extract ai-hardware-project-6501-group1-main.zip. Enter /ai-hardware-project-6501-group1-main/model and run LeNet-5.py. This file includes the construction of LeNet-5 model,

loading and processing the MNIST dataset (training set only) using TensorFlow framework, training the model, and finally saving it as the HDF5 (.h5) format. This Python file will finally generate LeNet-5.h5 in the current folder.

2. Run LeNet-5-int8.py. This file uses the first 100 test data in MNIST to quantize the model to int 8, and save it as tflite (.tflite) format, which is the lightweight version of Tensorflow. We use tflite because deploying the full Tensorflow framework on PYNQ-Z1 will consume a large amount of power and computational resources, making it inefficient for the platform's capabilities. This Python file will finally generate LeNet-5-int8.tflite in the current folder.

3. Enter /ai-hardware-project-6501-group1-main/Inference-Time-on-CPU and run LeNet-5-inference-time-on-CPU.py. The output shows the inference time and accuracy of LeNet-5 on your computer CPU by testing 10000 MNIST test data in /../data/minist_test.csv.

4.2 On PYNQ-Z1 without FPGA Acceleration

The following steps are based on an offline environment of PYNQ-Z1.

1. Follow the steps on the official website to set up PYNQ-Z1: `https://pynq.readthedocs.io/en/v2.5.1/getting_started/pynq_z1_setup.html`

2. Connect to Jupyternotebooks. Download the corresponding version of the armv7l wheel of tflite_runtime based on the Python version on PYNQ-Z1. Check the Python version in the PYNQ-Z1's terminal using the command: python –version. Then upload the wheel from your computer. In the terminal, navigate to the directory of the uploaded wheel, using ls and cd command (if you directly upload it by the upload button, use the command: cd /home/Xilinx/jupyternotebooks), and install it using the command: pip install your_tflite_runtime_liabrary.wheel.

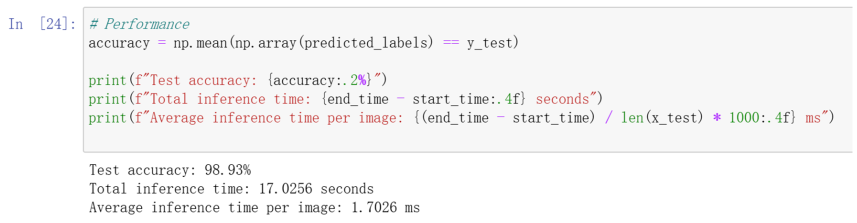
3. Enter /ai-hardware-project-6501-group1-main, and paste the code in Inference-time-on-PYNQ-Z1-without-acceleration.py to the jupyternotebooks. Run it and it will show the inference time and accuracy of LeNet-5-int8 model.

4.3 On PYNQ-Z1 with FPGA Acceleration

We referred to the work of Wang et al. [1], whose GitHub link is as follows: `https://github.com/awai54st/PYNQ-Classification/tree/master`. In their work, they designed an FPGA acceleration module for the LeNet-5 model. By following their instructions, we successfully used Vivado HLS to generate the bitstream file for FPGA configuration and uploaded it to the PYNQ-Z1 board. Alternatively, their pre-built image file can be used directly, which includes the FPGA module and pretrained Caffe network [2]. Their prepared code will output the inference time and accuracy of LeNet-5 with FPGA acceleration.

5 Expected Results

The expected inference accuracy of LeNet-5 on different platforms will remain 98% - 99%. The inference time on computer CPU, the processor of PYNQ-Z1, and the PYNQ-Z1 with FPGA acceleration are respectively expected to be less than 1 second, more than 10 seconds, and less than 2 seconds.

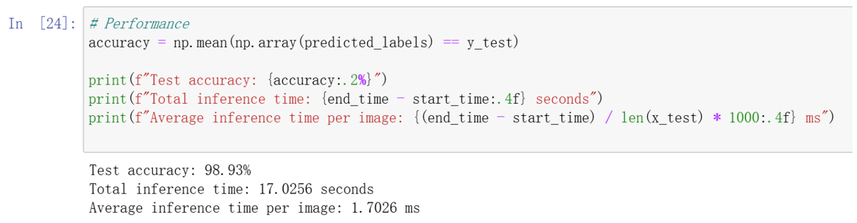


```
In  [24]: # Performance
          accuracy = np.mean(np.array(predicted_labels) == y_test)

          print(f"Test accuracy: {accuracy:.2%}")
          print(f"Total inference time: {end_time - start_time:.4f} seconds")
          print(f"Average inference time per image: {(end_time - start_time) / len(x_test) * 1000:.4f} ms")

          Test accuracy: 98.93%
          Total inference time: 17.0256 seconds
          Average inference time per image: 1.7026 ms
```

Figure 1: Model Performance on Computer's CPU

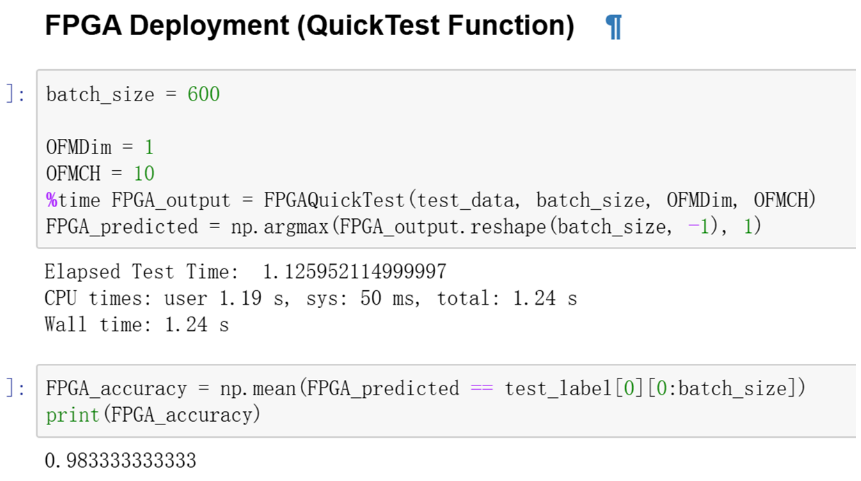


```
In  [24]: # Performance
          accuracy = np.mean(np.array(predicted_labels) == y_test)

          print(f"Test accuracy: {accuracy:.2%}")
          print(f"Total inference time: {end_time - start_time:.4f} seconds")
          print(f"Average inference time per image: {(end_time - start_time) / len(x_test) * 1000:.4f} ms")

          Test accuracy: 98.93%
          Total inference time: 17.0256 seconds
          Average inference time per image: 1.7026 ms
```

Figure 2: Model Performance on PYNQ-Z1 without FPGA Acceleration



FPGA Deployment (QuickTest Function) ¶

```
]: batch_size = 600

   OFMDim = 1
   OFMCH = 10
   %time FPGA_output = FPGAQuickTest(test_data, batch_size, OFMDim, OFMCH)
   FPGA_predicted = np.argmax(FPGA_output.reshape(batch_size, -1), 1)

   Elapsed Test Time:  1.125952114999997
   CPU times: user 1.19 s, sys: 50 ms, total: 1.24 s
   Wall time: 1.24 s

]: FPGA_accuracy = np.mean(FPGA_predicted == test_label[0][0:batch_size])
   print(FPGA_accuracy)

   0.983333333333
```

Figure 3: Model Performance on PYNQ-Z1 with FPGA Acceleration [1]

Figures 1 - 3 show our results. Running on the computer, the total reference time is 0.5324 seconds and the test accuracy is 98.94%. Only relying on the PYNQ-Z1 processor, the inference accuracy and time are 98.93% and 17.0256 seconds. With FPGA acceleration, the model's inference time is significantly reduced to 1.24 seconds, demonstrating substantial improvement, while maintaining a high accuracy of 98.33%. The reduction of inference time demonstrates the viability to deploy deep learning model LeNet-5 on resource-limited PYNQ-Z1 with FPGA acceleration for handwriting recognition.

References

[1] E. Wang, J. J. Davis, and P. Y. K. Cheung, "A PYNQ-based Framework for Rapid CNN Prototyping," in *IEEE Symposium on Field-programmable Custom Computing Machines (FCCM)*, 2018.

[2] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” *arXiv preprint arXiv:1408.5093*, 2014.