FPGA- Accelerated Sign Language Recognition using The Ztachip

Maiva Ndjiakou - hardware integration
Will Berling - software & testing
Daniel Lee - software design

09 December 2025

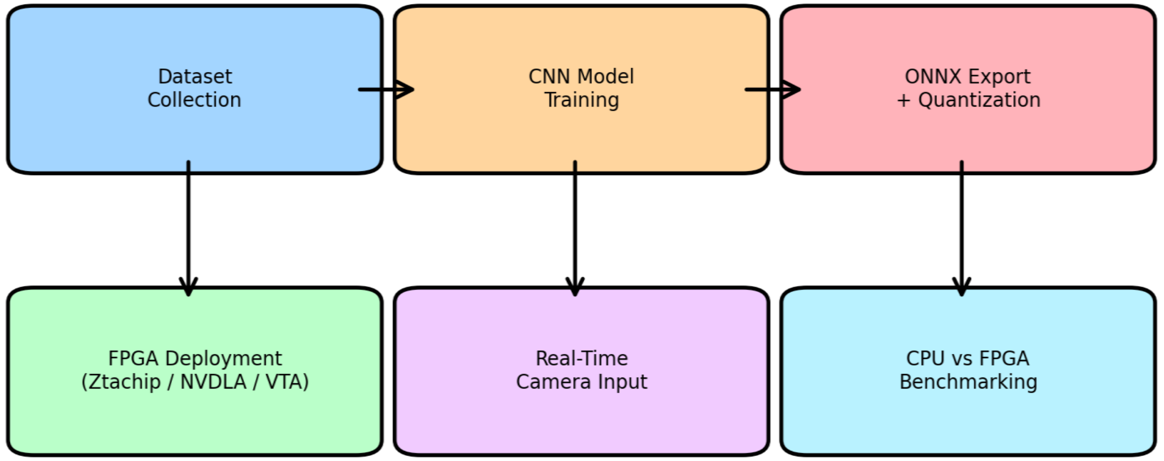
Problem : Efficient Real time ASL Recognition on FPGA hardware

Problem Overview

- Real-time American Sign Language (ASL) alphabet recognition requires fast and accurate inference.
- CNN-based sign recognition is computationally intensive, and CPU-only execution leads to high latency
- FPGA acceleration (Ztachip) leverages parallelism to improve speed while keeping power usage low.

Goal

Develop an ASL alphabet recognition pipeline that runs fast, accurately, and efficiently on Ztachip FPGA hardware.



Methodology Description

Workflow: Train → ONNX export → Ztachip compile → deploy.

Validation: Measure accuracy, FPS, and compare CPU vs FPGA latency.

Dataset | CNN model training | ONNX Export+ quantization

Dataset (Kaggle ASL Alphabet Dataset)

- Public dataset containing hand gesture images for all ASL alphabet letters ~87,000 labeled RGB images (200×200)
- Preprocessing steps:

✓ Resize to ***64***×***64*** for lightweight FPGA-optimized CNN
✓ Convert to NCHW format
✓ Normalize pixel values to [0,1]
✓ Split into train/validation sets

CNN Model Training

- Lightweight custom CNN designed to run efficiently on Ztachip
- Architecture includes:

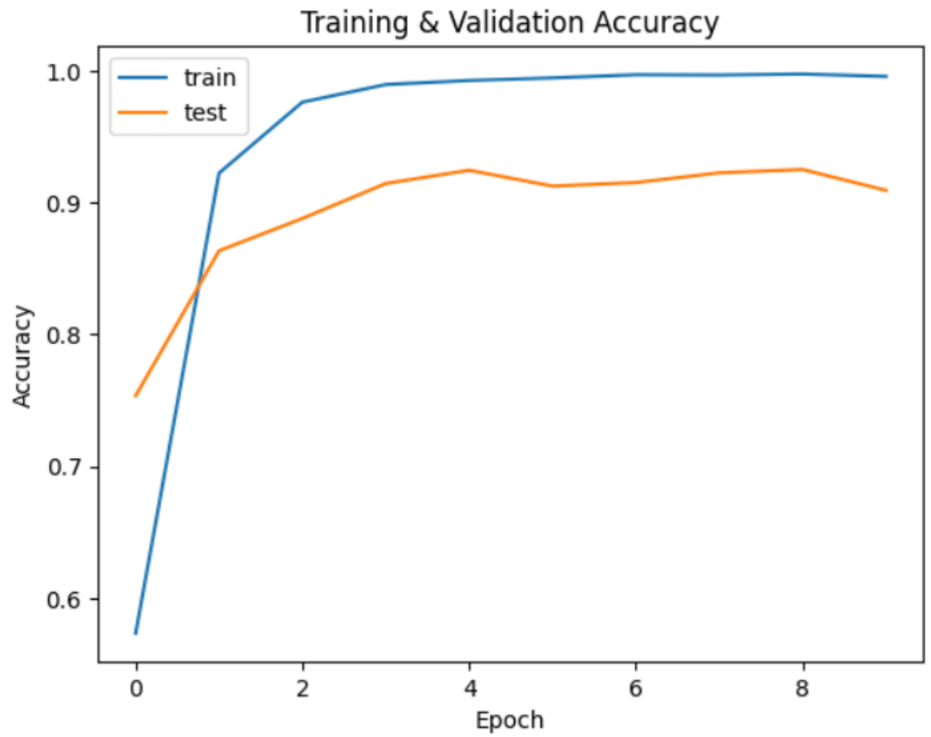
✓ 3 convolution blocks + ReLU
✓ Max-pooling layers
✓ Fully connected output with 29 classes (A–Z excluding J & Z, plus extra signs)

- Training pipeline:

Implemented in PyTorch
✓ Data augmentation (flip, rotate, brightness adjust)
✓ Cross-entropy loss + Adam optimizer
Achieved ≈ 90% accuracy on validation set

ONNX Export + Quantization for Ztachip : *Applied* ***INT8 quantization*** *model for deployment on Ztachip FPGA*

Image: Sign Language Alphabet



Board Set up | Future steps of the Project

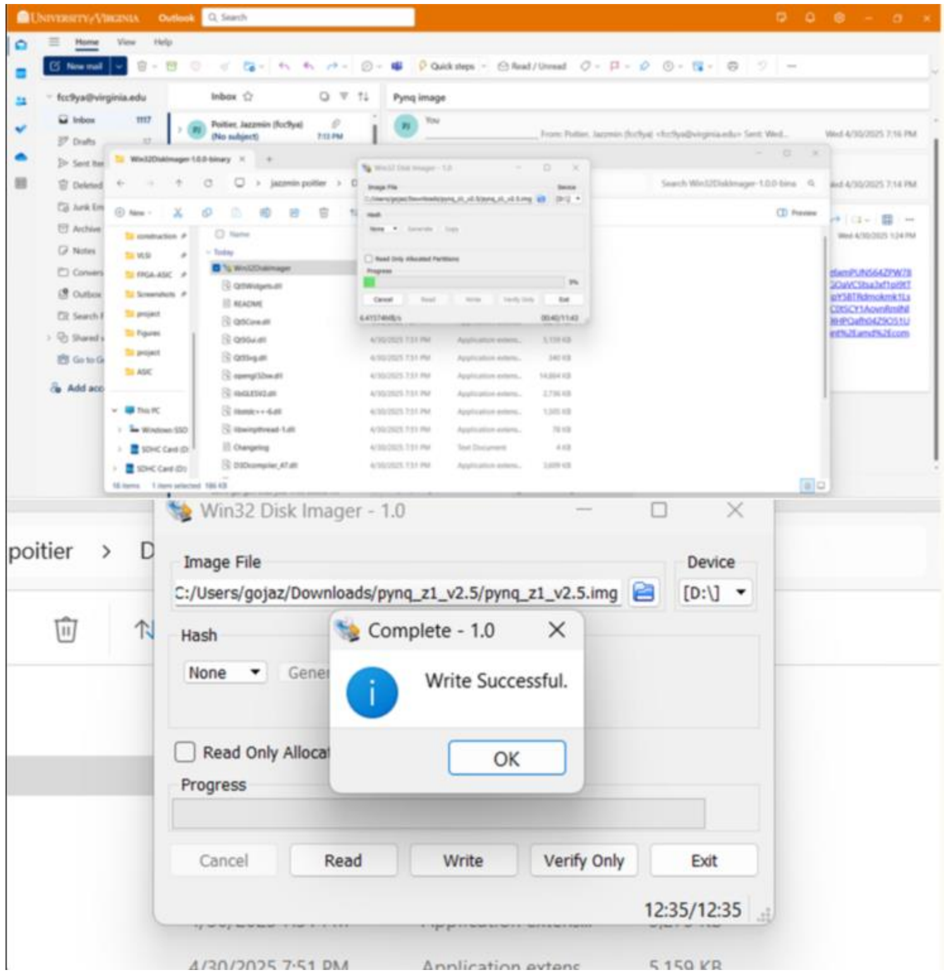


Image File: C:/Users/gojaz/Downloads/pynq_z1_v2.5/pynq_z1_v2.5.img

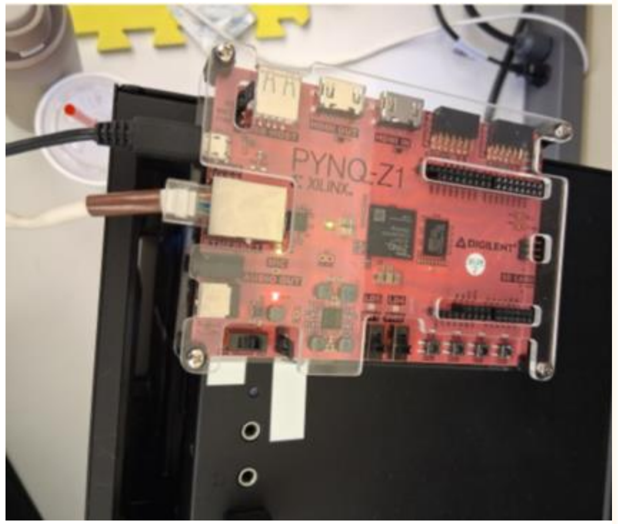
Device: [D:\]

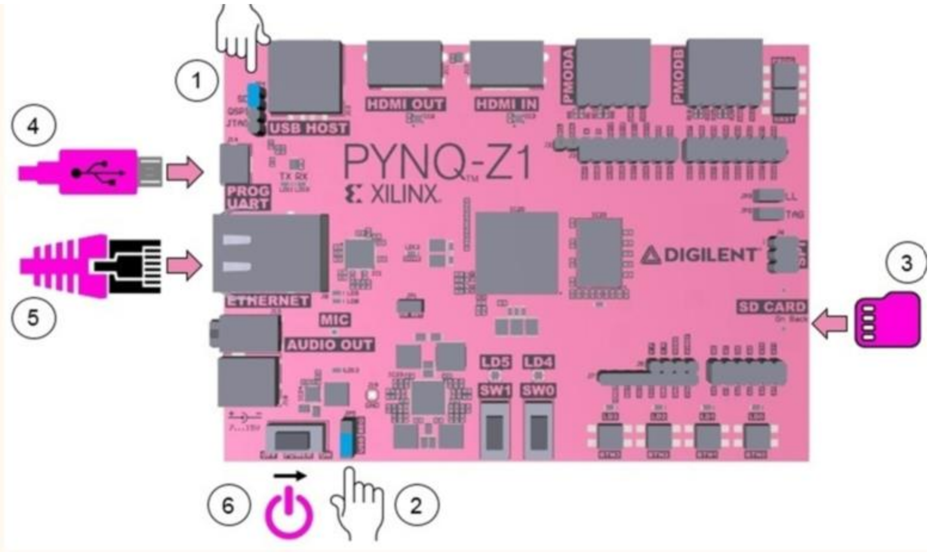
Hash: None

Write Successful.

12:35/12:35

- Installed PYNQ v2.5 image on the board's SD card





- Connected the board to PC, retrieved IP address, and accessed the Jupyter interface

✅ Future Steps of the Project

- Deploy the Quantized ONNX Model on the Ztachip FPGA
- Implement Real-Time Inference Pipeline
  - ✓ upload images through Jupyter Notebook
  - ✓ Capture live ASL hand gestures
  - ✓ Run inference through Ztachip and display predicted letters in real time
- Benchmark CPU vs FPGA Performance
- Final Report + Demo Video

Thank You