Marmara University

Faculty of Engineering

Electrical and Electronics Engineering

EE4065.1: INTRODUCTION TO EMBEDDED IMAGE PROCESSING#

Homework #4 Report & Results

Name Surname	Student ID
Yusuf Yiğit Söyleyici	150723524
Mehmet Açar	150719020

Introduction

The objective of this assignment was to implement embedded machine learning applications for handwritten digit recognition on the STM32 Nucleo-F446RE microcontroller. The work is based on Sections 10.9 and 11.8 of the course textbook (*Embedded Machine Learning with Microcontrollers*).

The assignment was divided into two main tasks:

1. **Binary Classification (Q1):** Distinguishing the digit "0" from "non-zero" digits using a Single Neuron model.
2. **Multi-Class Classification (Q2):** Recognizing all ten digits (0-9) using a Multi-Layer Perceptron (MLP) Neural Network.

Both implementations utilized a hybrid workflow: model training was performed offline using Python (TensorFlow/Keras) on the MNIST dataset, while inference—including the manual calculation of Hu Moments—was implemented in C on the STM32 microcontroller.

Q1 Single Neuron Application (Section 10.9)

2.1 Methodology

The goal was to classify images as "Zero" (Class 0) or "Not Zero" (Class 1).

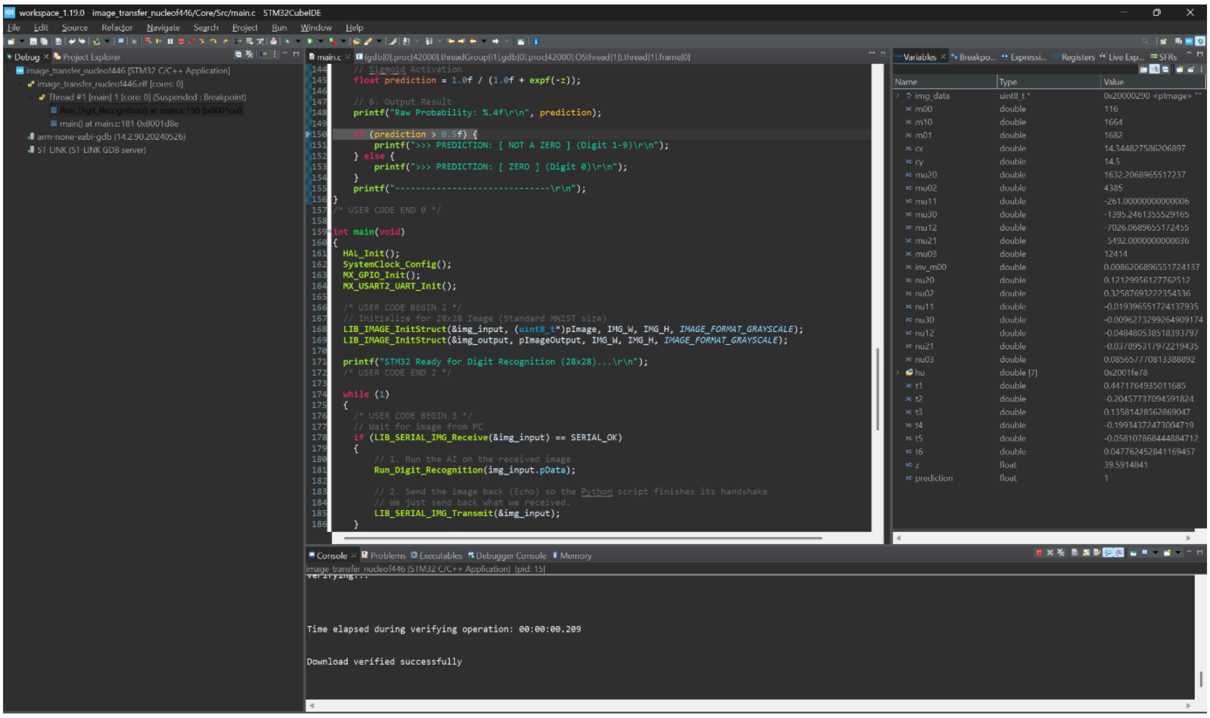
- Training (Python): A single neuron with a sigmoid activation function was trained on the MNIST dataset. The 784-pixel images were reduced to 7 Hu Moments before training. The learned weights and bias were exported to C.
- Embedded Implementation (STM32):
    - Input: 28x28 grayscale image received via UART.
    - Feature Extraction: A custom C function was implemented to calculate the 7 Hu Moments from raw pixels without using external libraries like OpenCV.
    - Inference: The neuron's output was calculated using the dot product of the normalized features and weights, followed by the sigmoid function:

$$P(y=1|x) = \frac{1}{1 + e^{-(W*x+b)}}$$

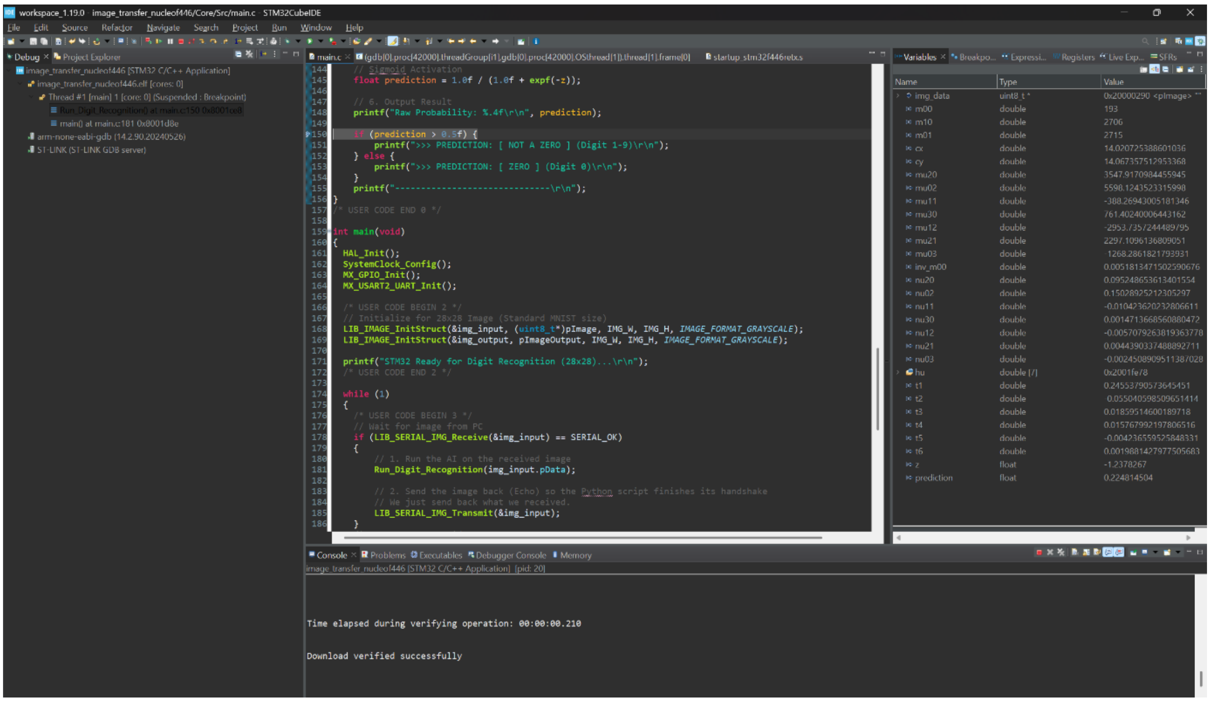
2.2 Results

The system was verified using sample images sent from a PC via UART.

- Test Case A (Digit '7'): The model predicted a probability of 1.0, correctly classifying the image as "NOT A ZERO".



- Test Case B (Digit '0'): The model predicted a probability of 0.22, correctly classifying the image as "ZERO".



Q2 Multi-Layer Neural Network (Section 11.8)

3.1 Methodology

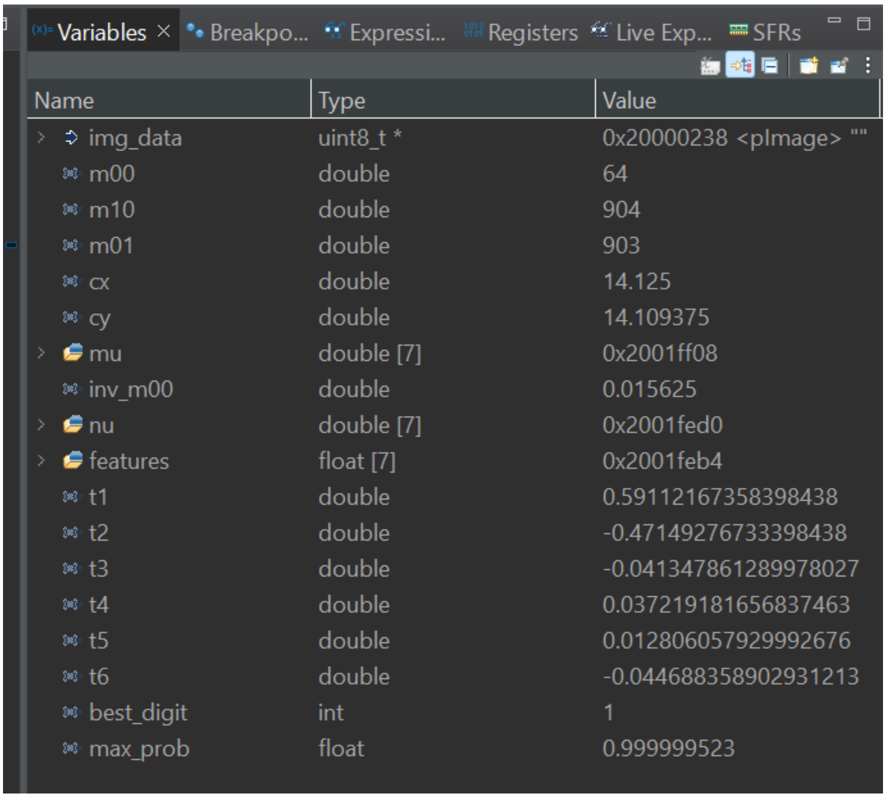
The goal was to classify digits 0-9 using a deep neural network.

- Architecture: A Multi-Layer Perceptron (MLP) was constructed with:
    - Input Layer: 7 neurons (Normalized Hu Moments).
    - Hidden Layer 1: 100 neurons (ReLU activation).
    - Hidden Layer 2: 100 neurons (ReLU activation).
    - Output Layer: 10 neurons (Softmax activation).
- Embedded Implementation: A general-purpose dense_layer function was implemented in C to handle matrix multiplication for layers of arbitrary size. The weights for all three layers were included via a generated header file (weights_data.h).

3.2 Results & Analysis

The model was tested with various digits. Distinct shapes were classified correctly, confirming the validity of the C implementation.

- Success Case: The digit '1' was correctly identified with high confidence.



Screenshots of test results for all of the digits can be found in the repository

3.3 Performance Analysis

While the code successfully executes the neural network logic, the model struggled to correctly distinguish certain digits. Specifically, digits '3' and '8' were frequently misclassified as '0', while the digit '5' was consistently misclassified as '7'.

Reasoning: These errors highlight the limitations of using Hu Moments as the sole feature extractor. Hu Moments describe global shape invariants—such as "roundness," "center of mass," and "skewness"—but they discard local topological details (like the number of holes or specific stroke angles).

- 3 & 8 identified as 0: Mathematically, '0', '3', and '8' share very similar mass distributions. They are all roughly symmetrical shapes where the pixel mass is distributed around a central empty space.
  - The Topology Problem: Hu Moments cannot "count" holes. They cannot distinguish between the single closed loop of a '0', the two stacked loops of an '8', or the open loops of a '3'. To the algorithm, all three appear as generic "round blobs" with similar rotational inertia.
  - Result: Since '0' is the simplest of these shapes, the network converges on '0' as the most likely prediction for all three.
- 5 identified as 7: This specific error arises from the "Top-Heavy" nature of both digits.
  - Mass Distribution: Both '5' and '7' are characterized by a strong horizontal bar at the top of the image.
  - Skewness: Unlike the symmetric '0' or '8', both '5' and '7' are asymmetrical. If the bottom loop of the handwritten '5' is open or angular, its overall center of gravity and skewness become statistically almost identical to a '7'.
  - Result: The feature vector for '5' overlaps significantly with '7', leading the model to predict the class it is more confident in (Class 7).

Conclusion

This homework successfully demonstrated the deployment of machine learning models onto a microcontroller.

1. Feature Extraction: Complex mathematical features (Hu Moments) were successfully implemented in pure C on the STM32F446RE.
2. Model Porting: Weights from a Python-trained model were successfully exported and used for inference in C.
3. Hardware-in-the-Loop: A robust testing framework was established using UART to transfer image data from a PC to the microcontroller for real-time verification.

The project highlights the trade-off in embedded ML: reducing dimensionality (using Hu Moments) lowers computational cost but introduces accuracy limitations for complex shapes.