
EE712 - Embedded Systems Design

# Handwritten Digit Classification

Project Proposal

Hemant Hajare(20D070037),
Tejaswee Sulekh(20D070082)

May 4, 2023

# Contents

# 1 Team Members

| Team Member Name | Roll Number |
|---|---|
| Hemant Hajare | 20D070037 |
| Tejaswee Sulekh | 20D070082 |

Table 1: Member Table

# 2 Motivation

Embedded systems are computing systems with tightly coupled hardware and software integration that are designed to perform a dedicated function. A typical embedded system has a single application and has application dependant constraints on power, size, performance and cost. Historically, a large number of sensors like accelerometers, gyroscopes, displays etc. have been interfaced to achieve application specific goals.

Machine Learning has contributed significantly to automate a large number of things in various spaces. However, these come up with setting up spacious computers with high speed computing and memory requirements to train the models. Once trained, these investments become redundant. Embedded systems, which typically are small sized, can be used to implement these trained models in a practical setting.

We plan to demonstrate such an integration of Machine Learning with embedded system in our project. We will implement an Handwritten digit classification model on Raspberry Pi Pico and demonstrate the live working using a camera module. Our system finds applications for security systems and attendance systems.

# 3 Conceptual Design Review

## 3.1 Micro-control Unit

We will be using **Raspberry Pi Pico** as the microcontroller unit for our system. Raspberry Pi Pico comes with dual-core ARM Cortex M0+ processor with clock speed of 133 MHz. It has 26 multi-function GPIO pins which support 2 channels each for Serial Peripheral Interface (SPI), Universal Asynchronous Receiver Transmitter (UART) and I2C communication. It also had accelerated floating libraries on-board which might come handy in our application. The raspberry pi can be programmed using Micro Python, and C/C++ Software Development Kit (SDK).

## 3.2 Camera module

We plan to use OV2640 2MP SPI Camera in order to capture images for our Machine Learning model. This is 1600 $\times$ 1200 Color SPI Camera. Image can be processed on board using I2C to alter sharpness, noise, defect, saturation and special effects.

### 3.3 LCD module

The hand written digit classification model that we will be using will not be very complex and it becomes important that we ensure a good feedback of the orientation of the camera. Thus we also plan to use an LCD display module to display the real-time image captured by the camera. A standard SPI TFT LCD seems to work with Raspberry Pi Pico.

### 3.4 Machine Learning Model

We know convolutional neural networks are the gold standard for image classification using Machine Learning. However, neural networks are complex and require heavy computing and a good amount of memory to operate.
On the other hand Support Vector Machines is another ML methodology that is less accurate and less reliable but can be implemented on embedded systems. We plan to use "LinearSVM" implementation of this technique available in the sckikit-learn Python package.

Note that we will train our model off-board using MNIST handwritten digit dataset that comes with scikit-learn.

### 3.5 Goals

Our project aims to develop an embedded system that can classify digits from 0 to 9 using Linear SVM based Machine Learning. We plan to provide a camera feedback on an LCD display to make sure the input image is correctly oriented. This main goal is divided into three sub-goals. First, interfacing the camera module. Second, interfacing the LCD module. Last, hardware implementation of Machine Learning algorithm.

## 4 Description and block diagram

### 4.1 Description

The camera module will be interfaced with the SPI and some additional pins. In all total 9 pins will used for interfacing the camera module with the Raspberry Pi Pico. The LCD module will be interfaced using 4 pin SPI interface. We will test the Machine Learning algorithm first on software (Python) then convert it to microPython so that it can be implemented on-board the Raspberry Pi Pico.

### 4.2 Block diagram

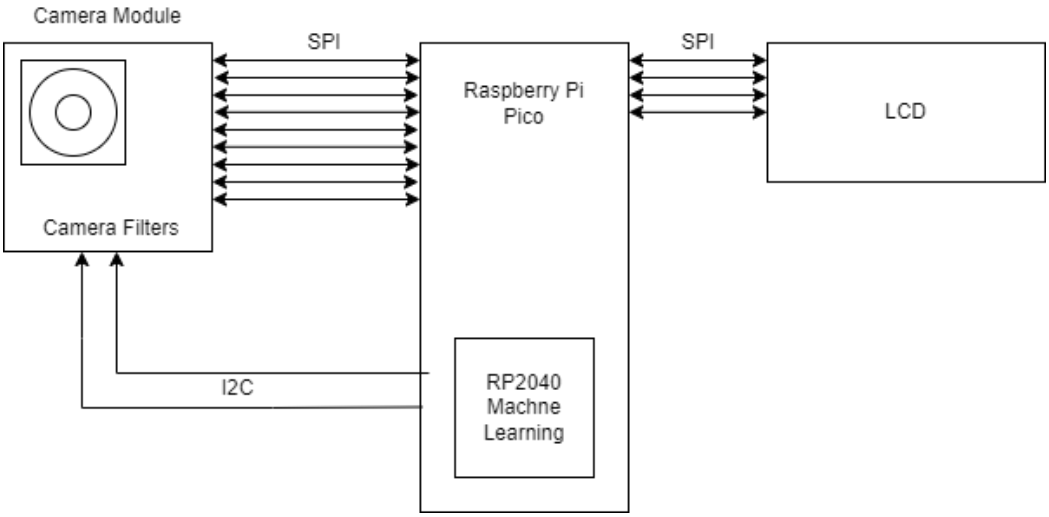


Figure 1: Block Diagram of The Project

## 5 Learning outcome and relevance

### 5.1 Memory-based optimisation

1. Handling the image data from camera sensor module with given memory constraints of Raspberry Pi Pico
2. Choosing just the right Linear SVM Machine Learning model that will give accuracy while at the same time meet the memory constraints
3. Identify and perform in-code optimizations based on the application environment

### 5.2 Sensor Interfacing

1. Interfacing the camera module with Raspberry Pi Pico
2. Interfacing the LCD with Raspberry Pi Pico
3. Interfacing the on-board camera functionality with Raspberry Pi Pico

### 5.3 Image Processing

1. Applying noise reduction techniques on images to extract maximum accuracy from the model
2. Identifying and tuning filters on-board the camera module for improving classification accuracy

# 6 BOM (as of Preliminary Design Report)

| Components | Code | Price (Rs) | Quantity | Website | Total |
|---|---|---|---|---|---|
| Camera Sensor | OV2640 | 650 | 3 | ifuturetch.org | 1950 |
| LCD display | Pico LCD | 700 | 3 | robu.in | 2100 |
| **Grand Total** | | | | | 3850 |

Note: We plan to discuss with the TA's to find other sensors with similar properties as both of them seem to be out of stock and our knowledge in the domain is little. These sensors are part of initial proposal and might be subject to change depending upon availability and cost.

# 7 Weekly Plan

| Week | Target |
|---|---|
| 1st March - 7th March | Software testing of Machine Learning model (Python) |
| 8th March - 14th March | Hardware testing of Machine Learning model |
| 15th March - 21st March | Debugging and optimization of Python code |
| 22nd March - 28th March | Interfacing camera and LCD modules with Raspberry Pi Pico |
| 29th March - 4th April | Performing preliminary testing of proposed Embedded System |
| 5th April - 11th April | Tuning and final testing |
| 12th April - 18th April | ENDSEM BREAK |
| 19th April - 25th April | ENDSEM BREAK |
| 26th April - 30th April | Presentation |

# 8 Division of Work

Hemant and Tejaswee will together work on the software testing of the Machine Learning algorithms till the sensors arrive after which Hemant will look after interfacing and Tejaswee will take up model tuning on hardware. Both of us will together work on the presentation, if any.

# 9 Implementation

## 9.1 Testing and Training

During implementation of this project we faced many problems, both in software and hardware concerning optimum model implementation and interfacing all the modules properly. For testing the project we followed the same set-up as mentioned in the block diagram above where, camera module is interfaced with a microcontroller and then its output is shown on an LCD screen.

### 9.2 Challenges in Model Deployment

Firstly we had to decide upon which ML model to train for image classification which we had to choose one while being mindful of the memory present in the microcontroller.

After some consideration we decided to go ahead with soft SVM model while downgrading the features of the image that is from 28X28 pixels to 12x12 pixels. This was done because if we went for higher features it would cause memory exhaustion. Still loosing on data caused the accuracy of the model to be hindered.

Another issue we had was the size of the image obtained from the camera module was different from the images we trained the model on to solve this issue we had to re-sample the image in code once more to obtain a 12x12 input image from what we obtained from the camera.

### 9.3 Hardware Interfacing

We found difficulty in interfacing the LCD module we ordered first because of lack of documentation thus we ordered another model of LCD display with better documentation.

Though we proposed to use a more sophisticated OV2640 for our system we shifted to a much cheaper OV7670 because of unavailability of the former one.

We used Circuit Python to interface OV7670 camera module and the TFT LCD to interface with the Raspberry Pi Pico. A number of resources were available online to carry out this interfacing along with the documentation.

Our system was very sensitive to motion of the object because of low frames/second and limited field of view (FoV).

It was not possible to handle these problems with the chosen sensor. However, these can be handled using better software to accommodate real time noise or upgrading the camera module with higher fps and FoV.

## 10 Results and Observation

We managed interface both the newly procured components together without any hiccups. The only issues which still persisted were not enough memory to allocate heavy model weights to predict without having any considerable classification error.

Definitely, the hardware that we had used was minimal and better hardware can be used. The goal of this project was however to optimise the performance ML model deployed on a simple controller like the Raspberry Pi Pico.

The real time system is different from the one tested on software as the real time image acquisition adds noise to the image which affects the classification error. This was evident in the experiments we performed.

A better image acquisition is thus a need of such a system which can considered in the future work.

We observed that our model had trouble classifying a few digits such as '8, 1'. While it worked for others given that there was enough lighting in the environment for the captured image to have enough information which could be processed.

## 11 Future Work

In future we could expand this idea on other form of classification such as object classification or person detector (i.e. determine who is present in the image) and deploy these models on heavier boards where memory is not an issue. We could also have facility of training the weights remotely rather than hard coding them onboard so that the model can adapt based on the working environment.

## 12 BOM

| **Components** | **Code** | **Price (Rs)** | **Quantity** | **Website** | **Total** |
|---|---|---|---|---|---|
| LCD display | Pico LCD | 700 | 1 | robu.in | 700 |
| SPI TFT LCD display | Arduino LCD | 362 | 1 | robu.in | 362 |
| **Grand Total** | | | | | 1062 |