Handwritten Digit Classification using Raspberry Pi Pico

Team Members

Ala Manas Royal
CSE 2021 - 2025
Amrita Vishwa Vidyapeetham
Coimbatore, India

Deeban Kumar M
CSE 2021 - 2025
Amrita Vishwa Vidyapeetham
Coimbatore, India

Mithil S
CSE 2021 - 2025
Amrita Vishwa Vidyapeetham
Coimbatore, India

Vignesh S
CSE 2021 – 2025
Amrita Vishwa Vidyapeetham
Coimbatore, India

***Abstract*—** This report details the creation of a system for classifying handwritten digits, which utilizes a Raspberry Pi Pico, an OV7670 camera module, a 128x160 TFT LCD display, and machine learning methodologies. The primary objective of this project is to run a machine learning model on the Raspberry Pi Pico that can analyze images taken by the camera and determine the handwritten digit depicted in the image. The report encompasses the hardware and software prerequisites, the wiring connections, the process of image processing, the training of the machine learning model, and the procedure for converting the model into a format that the Raspberry Pi Pico can interpret. It also discusses the challenges encountered, potential future enhancements, and provides guidance for troubleshooting.

***Keywords*—** Pico Board, TFT LCD Display, OV7670 CAM Module

I. INTRODUCTION

A. Problem Statement

The main goal of this project is to create a reliable system for classifying handwritten digits, specifically tailored for deployment on the Raspberry Pi Pico microcontroller. This involves the development and implementation of a machine learning model that achieves a balance between accuracy and computational efficiency, emphasizing algorithms such as Convolutional Neural Networks or Support Vector Machines. Integration into the Raspberry Pi Pico environment, utilizing CircuitPython, will be a crucial element. Additionally, the project aims to build a user-friendly interface to enable easy interaction, allowing users to input handwritten digits, obtain predictions, and visualize the results.

B. Objective

This project's primary aim is to implement a self-contained system for classifying handwritten digits on the Raspberry Pi Pico using machine learning techniques. The objectives encompass developing an efficient machine learning model tailored to the resource constraints of the Raspberry Pi Pico, ensuring accurate handwritten digit classification. Additionally, the project strives to integrate a user-friendly interface, potentially incorporating a TFT LCD display, to dynamically visualize and interpret digit classification results. By accomplishing these goals, the project aims to showcase the viability of deploying machine learning applications on edge devices like the Raspberry Pi Pico. Furthermore, it seeks to serve as an educational and accessible platform, fostering understanding at the intersection of machine learning and embedded systems.

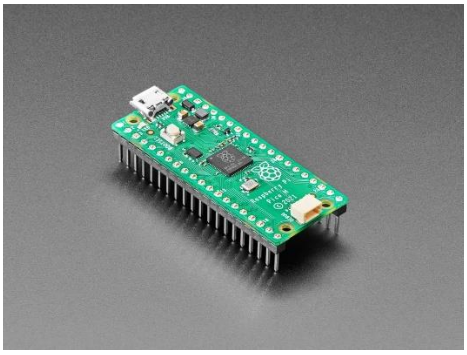
C. Scope of the Project

By conducting thorough testing and comprehensive documentation, this project aims to offer valuable insights to the community keen on deploying machine learning models on microcontrollers. The overarching objective is to demonstrate the practical application of embedded machine learning, with a specific focus on handwritten digit recognition using the Raspberry Pi Pico.

II. COMPONENTS USED

A. *Raspberry Pi Pico:*

The Raspberry Pi Pico is a microcontroller board based on the RP2040 chip. It's designed for low-cost and high-performance microcontroller applications. In this project, the Raspberry Pi Pico serves as the central processing unit, running the machine learning model and controlling the interactions between other components.



B. OV7670 Camera Module:

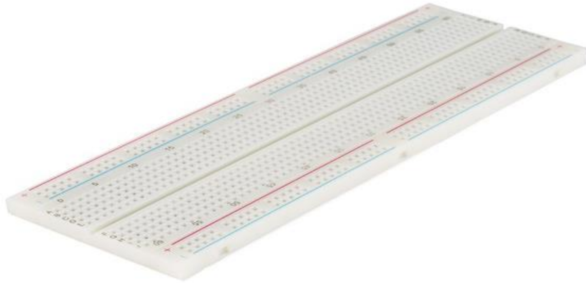
The OV7670 is a low-cost, small-sized camera module capable of capturing images. It plays a crucial role in this project by capturing handwritten digits. The images taken by the camera are processed and analyzed by the machine learning model running on the Raspberry Pi Pico.

C. 128x160 TFT LCD Display:

The 128x160 TFT LCD display provides a visual output for the project. It allows users to see the captured images, processed results, and the predicted handwritten digit.
The display enhances the user experience by providing real-time feedback.

D. Full-sized Breadboard:

A full-sized breadboard is used for prototyping and creating temporary connections between various components. It simplifies the wiring process and allows for a more organized setup during the development phase.



E. Jumper Wires for Peripheral Connections:

Jumper wires play a crucial role in establishing reliable connections between the Raspberry Pi and various peripherals, including components such as the ST7735 TFT SPI display, USB microphone, and other hardware relevant to language translation applications. These wires are flexible connectors with male or female ends, enabling seamless linking of different elements within a project. Jumper wires are highly versatile, facilitating the creation of custom wiring configurations. Their flexibility allows users to adapt and customize the connections based on the specific requirements of the language translation system. This versatility is particularly advantageous in projects where space constraints or unique

III.START UP PROTOCOL FOR RASPBERRY PI PICO OS

A. About the OS:

Tailored for the Raspberry Pi Pico microcontroller, the Raspberry Pi Pico OS is engineered to maximize computational capabilities. Employing a customized approach, it prioritizes performance enhancement, specifically addressing the distinctive features of the Pico platform. This operating system is meticulously designed to deliver an efficient and smooth experience for developers and enthusiasts engaged in projects utilizing the Raspberry Pi Pico.

B. Empowering Users with Microcontroller Computing:

The OS for the Raspberry Pi Pico provides users with the means to fully unleash the capabilities of the microcontroller. Specifically designed for the Pico's architecture, this OS ensures an optimized computing experience, allowing users to make the most of the microcontroller's potential. Striking a balance between innovation and compatibility within the broader Raspberry Pi ecosystem, the Raspberry Pi Pico OS highlights the adaptability and versatility inherent in its design.

C. Booting Process:

The Raspberry Pi Pico OS provides versatile booting options, allowing users to initiate the system from different sources like Flash, UF2, or a custom bootloader. The startup sequence involves the following steps:

1. Media Insertion: Insert the Flash, UF2, or custom bootloader into the Raspberry Pi Pico.

2. Power Connection: Connect the Raspberry Pi Pico to a power source.

3. Automatic Boot: The Raspberry Pi Pico will autonomously boot from the inserted media.

4. Initialization: Following the completion of the boot process, the system initializes, granting users access to the Raspberry Pi Pico OS environment.

This streamlined startup protocol ensures a simple and efficient process for initializing the Raspberry Pi Pico OS, allowing users to swiftly commence their microcontroller projects.

IV. CONNECTION SETUP

A. Connection of Pi and ST7735 TFI SPI Display:

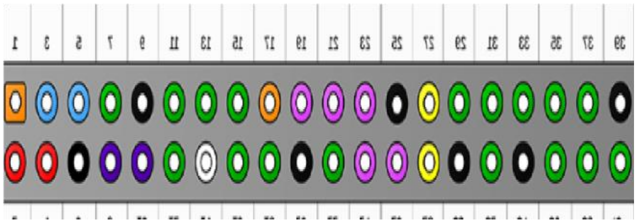


Fig. 7. Pin diagram of Raspberry pi 3

Fig. 8. Pin diagram of ST7735 TFI SPI Display

TABLE I

Raspberry Pi Pico	ST7735 TFI SPI	Purpose	Connection
3.3V Power (36st Pin) *	LED (Light Emitting Diode)	Provides power to the LED backlight of the display.	Linked to the 3.3V power source on the Raspberry Pi (1^{st} pin).
SPI Clock (Serial Peripheral Interface) (14th Pin)	SCL (Serial Clock)	Transmit clock signals for Aligned data transfer.	Wired to GPIO (General Purpose Input/Output) on the Raspberry Pi (23^{rd} pin)
SPI MOSI (Master Out Slave In) (15th Pin)	SDA (Serial Data Line)	Transmits data from the Raspberry Pi to the display	Connected to GPIO (MOSI) on The Raspberry Pi (19^{th} pin)
GP1018 (21st Pin)	AO (Analog 0)	Selects between command and data for the display.	Linked to GPIO 18 on the Raspberry Pi (12^{th} pin).

Raspberry Pi	ST7735 TFI SPI	Purpose	Connection
GP1023 (22th Pin)	RESET	Resets the display to its default state.	Wired to GPIO 23 on the Raspberry Pi (16th pin).
GP1024 (24th Pin)	CS (Chip Select)	Enables or disables the display for data communication.	Connected to GPIO 24 on the Raspberry Pi (18th pin).
Ground (38th Pin)	GND (Ground)	Serves as the common ground reference.	Connected to the ground (GND) pin on the Raspberry Pi (39th pin).
5V Power (39th Pin):	VCC (Voltage at the Common Collector)	Provides power to the display.	Wired to the 5V power source on the Raspberry Pi (2nd pin).

Connecting the OV7670 camera module to the Raspberry Pi requires both hardware and software setup. After attaching a pin header to the camera board, connect the power pins (VCC and GND) to matching GPIO pins on the Raspberry Pi. Configure the Raspberry Pi camera interface to recognize the OV7670 module. Test the connection to ensure it functions properly. Adjust script parameters to optimize image quality, resolution, and other settings.

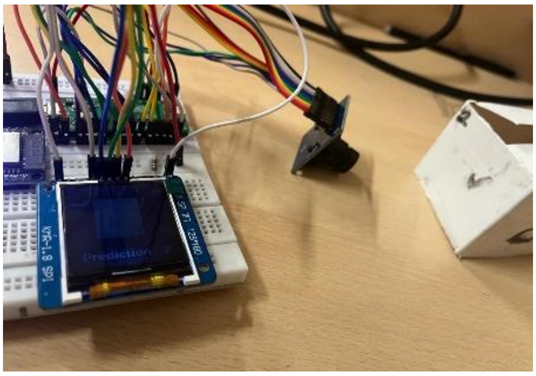


Fig. 10. Connection of Pi, ST7735 TFI SPI Display and OV7670

V. METHODLOGY

A. User Input:

1. Users will write a number and it has to be shown OV7670 camera.
2. Detecting processes is initiated upon detecting user input.

B. Target Detection:

1. Machine Learning Model identifies the input language.
2. The detected Number will be displayed on the TFT LCD.

C. Target Prediction:

1. The number is predicted using SVM Machine Algorithm.
2. System processing prepares for Prediction

D. Console Display:

The Predicted Number will be displayed on the console allowing for easy verification.


```
[0.546875, 0.625, 0.554688, 0.617188, 0.625, 0.625, 0.625, 0.554688, 0.554688, 0.5625, 0.5625, 0.5625, 0.539063, 0.625, 0.55
4688, 0.617188, 0.625, 0.5625, 0.625, 0.5625, 0.5625, 0.625, 0.625, 0.625, 0.53125, 0.617188, 0.546875, 0.554688, 0.625, 0.6
25, 0.625, 0.5625, 0.625, 0.625, 0.625, 0.632813, 0, 0.539063, 0.523438, 0.617188, 0.632813, 0.632813, 0.632813, 0.625, 0.62
5, 0.625, 0.632813, 0.640625, 0, 0.507813, 0, 0.640625, 0.640625, 0.640625, 0.648438, 0.648438, 0.734375, 0.828125, 0.84375,
0.84375, 0, 0, 0, 0, 0.835938, 0.835938, 0.835938, 0.835938, 0.84375, 0.84375, 0.84375, 0.835938, 0, 0, 0, 0.828125, 0.83593
8, 0.835938, 0.828125, 0.828125, 0.828125, 0.828125, 0.757813, 0.742188, 0.570313, 0.828125, 0.734375, 0.625, 0, 0.648438, 0
```


output provides comprehensive information.

1. It Provides the image of that particular number that user has shown.
2. It will show the Prediction of that particular number.

The Predicted Model will not always give the expected output as the model is still learning.

E. ST7735 SPI Display:

Simultaneously, the translated output is sent to the ST7735 TFI SPI Display, providing a visual representation of the Predicted Number.

Fig. 13. ST7735 SPI Display output

VI. CONCLUSION

In summary, the Handwritten Digit Classification project, which utilized the Raspberry Pi Pico and Machine Learning, represents a significant exploration into the merging of microcontroller technology and artificial intelligence. Our main objective was to create a standalone system capable of analyzing handwritten digits captured by an OV7670 camera module and providing real-time predictions using a machine learning model. The incorporation of a compact and cost-effective microcontroller like the Raspberry Pi Pico, along with an OV7670 camera module and a 128x160 TFT LCD display, demonstrated the feasibility of implementing machine learning on resource-constrained devices. The successful execution of a machine learning model entirely on the Raspberry Pi Pico highlights its adaptability and paves the way for similar applications in edge computing. The careful wiring and hardware components, including the TFT LCD and OV7670 camera, were essential for the system's functionality. The post-processing of camera images, training of the machine learning model using the scikit-learn library, and subsequent export of the model to a Pico-friendly format added layers of complexity, showcasing the interdisciplinary nature of the project. While the project delivers a functional handwritten digit classification system, there are opportunities for future improvements. Exploring larger image sizes, using data directly from the camera for training, and investigating different datasets (shapes, signs, patterns) are exciting avenues for further development. Additionally, the potential conversion of the project to C/C++ using the Pi Pico SDK and TF Lite micro presents opportunities for optimization and broader application. Ultimately, the Handwritten Digit Classification project stands as evidence of the evolving landscape of embedded systems and the integration of machine learning. As technology progresses, the synergy between microcontrollers and artificial intelligence opens up new possibilities for compact and intelligent edge devices. This project contributes to this narrative, showcasing the capabilities of the Raspberry Pi Pico and its potential to drive innovation at the intersection of hardware and machine learning.

VII. FUTURE DEVELOPMENTS

Looking ahead, the Handwritten Digit Classification project on the Raspberry Pi Pico holds promise for several future developments. These include the potential for real-time training from camera data, memory optimization for larger image sizes, and the exploration of diverse datasets to expand its classification capabilities. Additionally, transitioning to C/C++ using the Pi Pico SDK and TensorFlow Lite, improving the user interface, and fostering community collaboration are all part of the envisioned future for the project. These efforts aim to enhance the project's functionality, efficiency, and collaborative potential. Overall, the project represents an innovative fusion of machine learning and microcontroller technology, inviting further exploration and advancements in the field. The provided documentation and troubleshooting guidance ensure accessibility for enthusiasts and developers interested in similar pursuits.

[1] Programming the Raspberrypi Pico in CircuitPython by SteveCope

[2] RaspBerrypi Pico DIY Workshop by Sai Yamanoor (Author)

[3] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[4] https://ashishware.com/2022/09/03/pipico_digit_classification/#Training-ML-model-1