Programming with Symbols

Image Recognition for Drawings

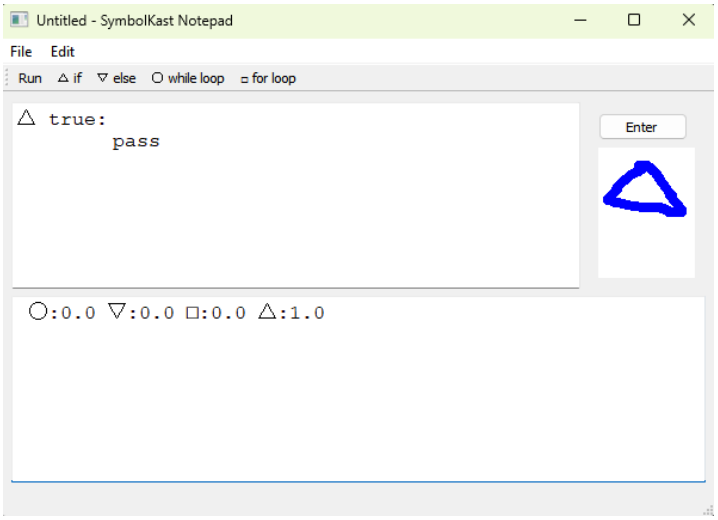
Sefunmi Ashiru *(Author)*
Digital Image Processing – ECE Department
Virginia Tech
Blacksburg, VA, United States

***Abstract*—This paper outlines my proposal for developing an image recognition application used for programing with symbols identified from drawings on a touchscreen. The application would consist of image preprocessing, ML image detection as well as a backend interpreter and front-end GUI for the user.**

I. INTRODUCTION

The goal of this project is to create a program that will allow the user to input a symbol drawing and then output the symbol drawing in a new window as code statements for programing. The program will help improve the user interface of programing by allowing the user to draw a symbol and then have the program output the code for the symbol. This will allow the user to write code faster and provides greater accessibility to programing. The full project consists of 3 parts. The first part is the user interface, the second part is the computer vision used to detect symbols and the third part is the code generation and execution.

Code example:
/// If num == 1:
/// Pass
///ifel num == 2:
/// Pass
///Else:
/// Pass
△ num == 1 ➡ Pass △
◆ num == 2 ➡ Pass ◆
▽ ➡ Pass

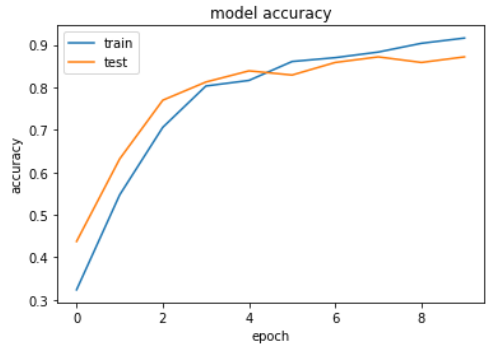


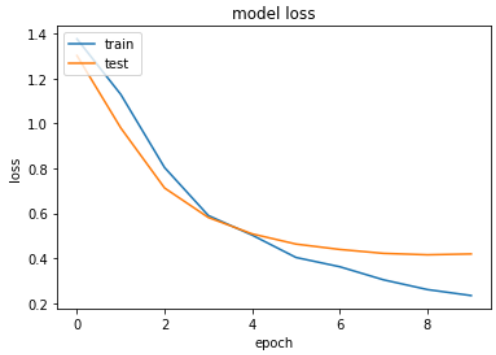
II. APPROACH

The first part of the project is the user interface. The user interface will be created using the Qtpy library. The user interface will have a canvas for the user to draw on, a button to enter the symbol drawing. It also contains a menu, toolbar, text editor and terminal. The menu will contain actions for handing files and editing the text editor.

The second part of the project is the computer vision. The computer vision will be created using the OpenCV library. The computer vision will be used to detect the symbol drawing and then output the code for the symbol drawing. The model uses a CNN and applies a gaussian blur to the image as a preprocessing step. The CNN is trained on a dataset of 500 images for each symbols class. the model has an accuracy of 84% on the test set. The model is then used to predict the symbol drawn by the user.

Identify applicable sponsor/s here. *(sponsors)*



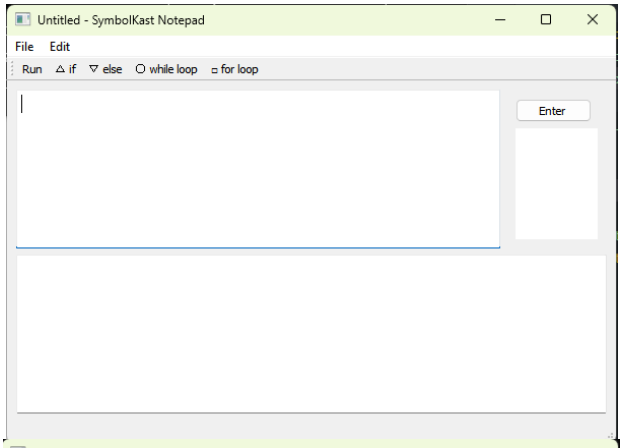


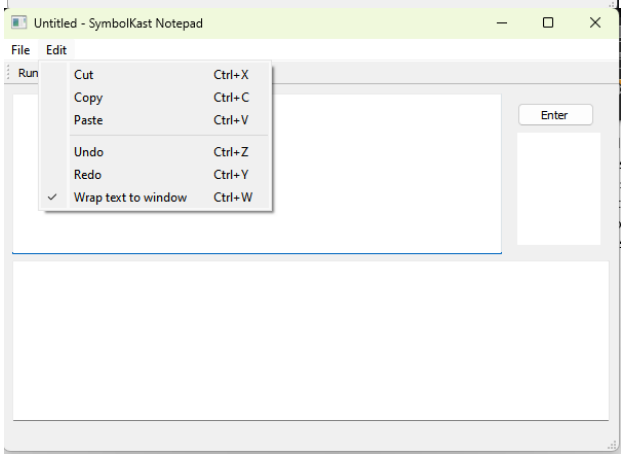
The third part of the project is the code generation and execution. The code generation and execution will be created using the Qtpy library. This will be done by using a dictionary to map the symbol to replace then symbols with their common python code. The code will then be executed using the exec with the output displayed through the terminal widget.

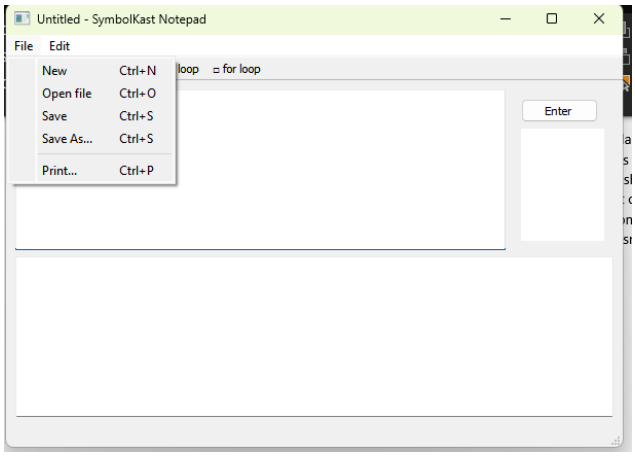
III. EXPECTED RESULTS

The way to run and use the application is to run the main.py file. The user will then be able to draw a symbol on the canvas and then press the enter button. The symbol will then be detected, and the code will be displayed in the text editor. The user can then press the run button to execute the code and see the output in the terminal.

Images of the Application in action have been provided below. The first image shows the blank user interface with the canvas, menu, toolbar, text editor and terminal. The second image shows the user drawing an if statement symbol and the correct value being outputted in the text editor. It should be noted that thought he model has an accuracy of 84% the predictions maybe wrong and out output the wrong statement The third image shows a script of commands and the expected output from the terminal. However due to a bug with using Unicode the data the terminal functionality doesn't run but the logic is present in the code.







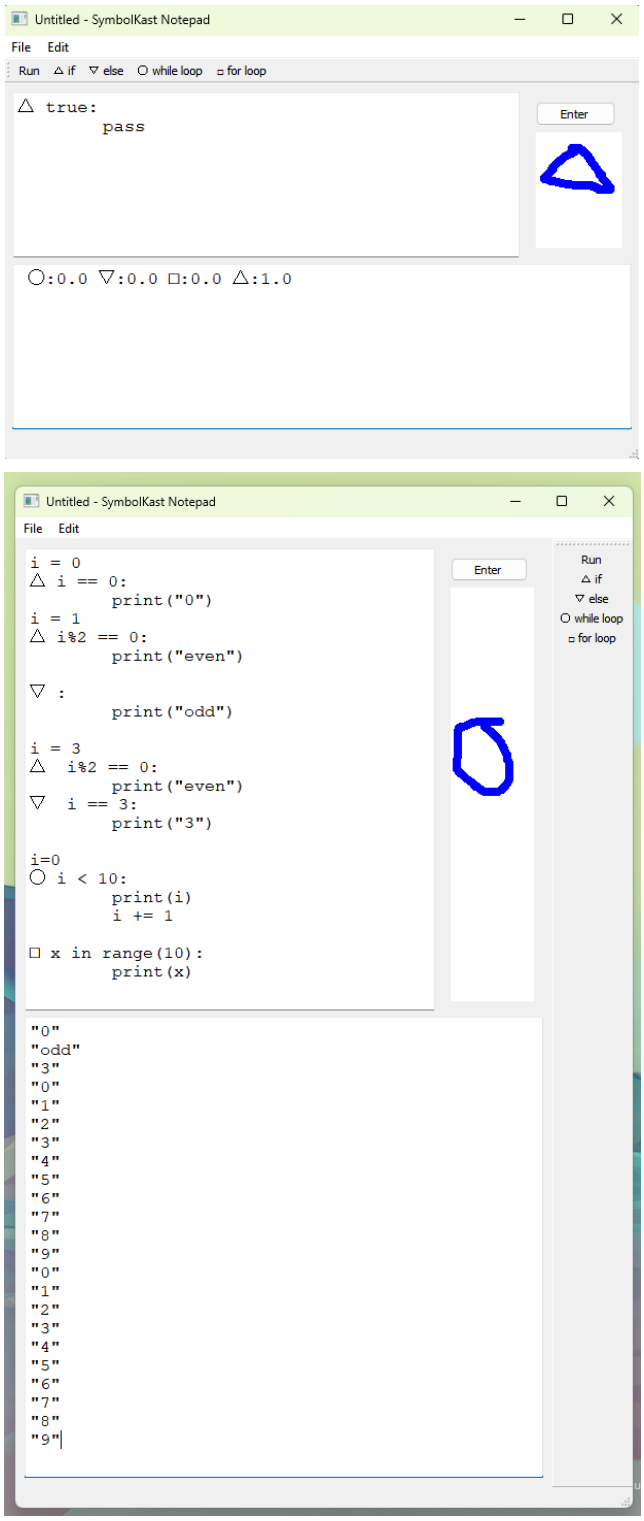


Figure Labels: Application Screenshots 1-5

IV. CONCLUSION

The limitations of the project are the accuracy of the model. The model has an accuracy of 84% on the test set. However, the model is not perfect and may output the wrong symbol. The model also has a hard time detecting the symbols when they have a thin brush stroke as well as when they are drawn in a small area. If I were to continue this project, I would try to improve the accuracy of the model by training it on more data and using a different model. I would also try to improve the model by using a different preprocessing step. Overall, i am happy with the results of the project and I think it is a good start to a larger project. I think the project is a good example of how computer vision can be used to improve the user interface of programing and help people in there day to day lives with communicating with computers.

REFERENCES

[1] T. Jain, "Basics of Machine Learning Image Classification Techniques," OpenGenus IQ: Computing Expertise & Legacy, 29-Aug-2019. [Online]. Available: https://iq.opengenus.org/basics-of-machine-learningimage-classification-techniques/. [Accessed: 09-Nov2022].

[2] "Quick, draw! Doodle Recognition Challenge," Kaggle. [Online]. Available: https://www.kaggle.com/competitions/quickdraw-doodlerecognition/data. [Accessed: 09-Nov-2022].

[3] "Python gui guide: Introduction to tkinter," Python GUI Guide: Introduction to Tkinter - SparkFun Learn. [Online]. Available: https://learn.sparkfun.com/tutorials/python-gui-guideintroduction-to-tkinter. [Accessed: 09-Nov-2022].

[4] (HOW TO WRITE A (LISP) interpreter (in python)). [Online]. Available: https://norvig.com/lispy.html. [Accessed: 09-Nov-2022].