Progaming with Symbols

Image Recognition for Drawings

Sefunmi Ashiru *(Author)*
Digital Image Processing – ECE Dept.
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 problem will be tackling for my final project is symbol recognition for touch screen programing. The goal is to create an interpreter that recognizes symbols being draw on a monitor and relates it to functionality or procedures that can be executed by the interpreter. The symbols that the model will be trained on include the alphabet as well as numbers 0-9, and basic polygon shapes (e.g., upwards, and downwards triangle, diamond, circle, and square).

Code example:
/// If num == 1:
/// Pass
///Ifel num == 2:
/// Pass
///Else:
/// Pass
▲ num == 1 ➔ Pass ▲
◇ num == 2 ➔ Pass ◇
▼ ➔ Pass ▼

II. APPROACH

Model Training

The first step in my approach is to develop a trained model that uses computer vision and digital image processing to identify the symbols. I will collect data from Kaggle that will cover the alphabet, whole numbers, and the close shapes mentioned earlier. I plan to use affine transformations and gaussian blur to create more noisy data for training. After cleaning the data, I will use it to train three different algorithms for image recognition and measure their performance. The three options are Naïve Bayes, SVM's, and CNN's. I believe CNN's will have the best performance, but it is worth comparing it to simpler cheaper models to replicate. For each model I will train and test its hype parameters and select the model's performance with the with the highest score and least over fitting bias.

Interpreter and GUI Implementation

For the interpreter I plan to copy a past project to create a lisp interpreter in python. The GUI will also be simple and consist of 5 buttons for clearing updating and recognizing characters, a canvas to draw, and 2 text fields to input commands and display results respectively.

Component Integration

When it comes to integrating the model into the program, I plan to use a HashMap to map recognized symbols from the model to character to add to current word or new word.

III. EXPECTED OUTCOMES

The end goal for this project is to create a GUI application where a user can draw a symbol and have it recognized and added to their program script. After completing the script, the program will be evaluated with its unique symbols being mapped to procedures. The hope for this project is to make programming more accessible. I would also hop to later modify this program to work in VR /AR spaces.

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-learning-image-classification-techniques/. [Accessed: 09-Nov-2022].

[2] "Quick, draw! Doodle Recognition Challenge," *Kaggle*. [Online]. Available: https://www.kaggle.com/competitions/quickdraw-doodle-recognition/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-guide-introduction-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].