FALCULTY
OF MECHANICAL ENGINEERING

HCMUTE

HAND DRAW PREDICTION APPLICATION

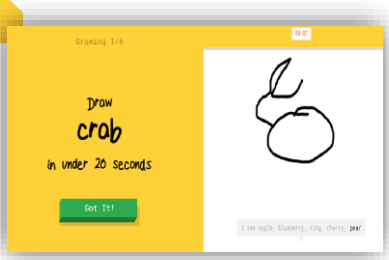
Lecturer:
Nguyen Truong Thinh
Made by:
Nguyen Dang Duy Tan

INTRODUCTION

I present a hand-drawn prediction application that is based on a neural network model trained on the Quick Draw dataset. The model is able to achieve high levels of accuracy on the Quick Draw dataset, and demonstrates good generalization to other datasets as well. In addition, I discuss the strengths and limitations of the model, as well as potential improvements that could be made. Overall, the results suggest that the neural network model trained on the Quick Draw dataset is a powerful tool for hand-drawn image recognition tasks.

RELATED WORK

One of the most popular and well-known examples of hand -drawn image recognition is the Quick Draw application developed by Google.



This application uses a similar approach to my project, where users can draw an image on a canvas and then receive a prediction of the object represented by the drawing. However, Quick Draw uses a different dataset and a different type of neural network, called a recurrent neural network (RNN), for its predictions. Despite the differences in implementation, the overall goal of Quick Draw and my project is the same: to predict the hand-draw image

DATASET

The dataset used for this project is the Quick, Draw! dataset, which was developed by Google. The dataset contains more than 50 million drawings from 345 categories, such as animals, vehicles, and everyday objects.

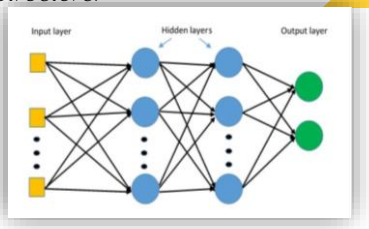
Each drawing is a 28x28 matrix in greyscale. The dataset is publicly available on the Google Cloud Platform and can be downloaded in a variety of formats, including .csv, .json and .npy.

Quick Draw dataset offers diverse training data, important for deep learning models to learn variations of objects. It is also pre-processed and ready for training.

MODEL

MLP (Multilayer Perceptron)

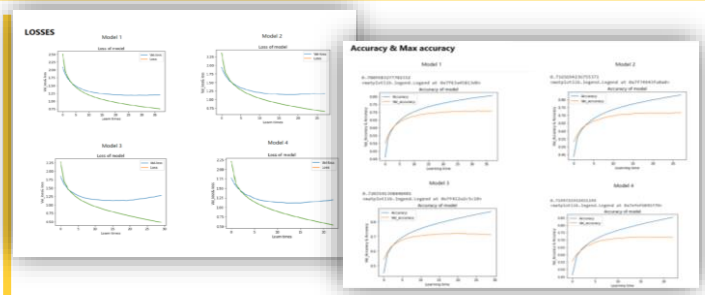
Structure:



I will evaluate several MLP architectures and find the best performance. Starting with a basic 3-layer model and increasing complexity to improve performance

Model	Model 1	Model 2	Model 3	Model4
Learning rate	0,0001	0,0001	0,0001	0,0001
Optimizer	Adam	Adam	Adam	Adam
n layers	3	4	5	6
n epochs	100	100	100	100
Layer 1	(784, 500)	(784,500)	(784,500)	(784,500)
Layer 2	(500, 256)	(500,500)	(500,500)	(500,500)
Layer 3	(256, 80)	(500,256)	(500,500)	(500,500)
Layer 4		(256,80)	(500,256)	(500,500)
Layer 5			(256,80)	(500,256)
Layer 6				(256,80)

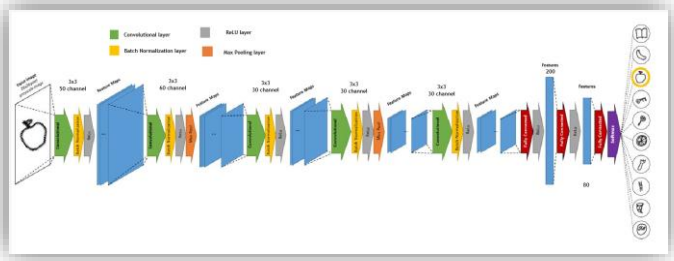
Collect all result for better comparision:6



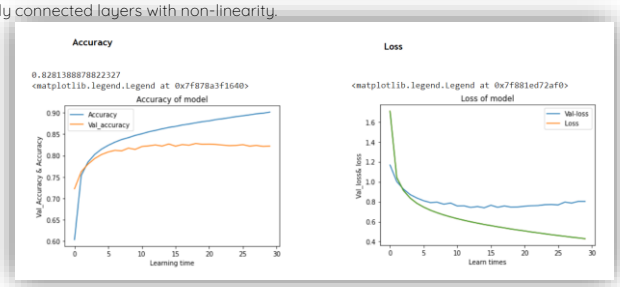
The highest accuracy achieved was 72.03% in Model 3.

CNN (Convolutional Neuron network)

Structure:



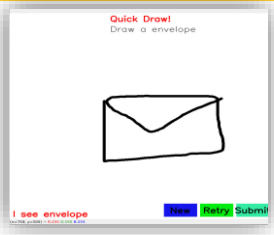
This architecture consists of 5 convolutional layers with non-linearity, 2 max-pooling layers, and 3 fully connected layers with non-linearity.



After implementing the CNN architecture, compared the results with previous used MLP model and found that CNNs significantly improved the accuracy. I obtained an accuracy of 82.81% on the test set using the above model.

APPLICATION & RESULT

I created an app that accepts user drawings based on prompts and processes them using basic image processing techniques before passing them through a trained model for evaluation. The app allows users to have fun while also checking the model's performance and making improvements if needed. It has three buttons: one for requesting a new drawing assignment, one for retrying a drawing task, and one for submitting the drawing for evaluation.



The result are pretty accurate, but there is a certain margin of error due to the limitations of the model and errors in the Quick Draw dataset. To improve accuracy, it would be necessary to clean and pre-process the data more thoroughly, and apply more complex architectures or techniques, such as transfer learning, to the model.