EECS 649 Group Project 2: QuickDraw!

An A-grade project attempt

Rohan, Shuai, Max, Saharsh

Project Goal

Purpose: Get hands-on experience with Classification problems using a Deep Neural Network

Motivation: Enhance knowledge in the field of Artificial Intelligence and get hands-on practice

Using the QuickDraw! dataset from Google containing a broad variety of drawn objects done by real persons in a limited amount of time using an online interface

Goal: Classify the images

Project Overview

We developed three models:

1) k-NN
2) MLP Classifier
3) Convolutional Neural Network using Python and Tensorflow/Keras: Developed with Google Colab

Project Approach

1) Understanding the dataset
2) Clean and split the dataset for validation
3) Transform the data to fit a model
4) Train, compile and test

The Dataset

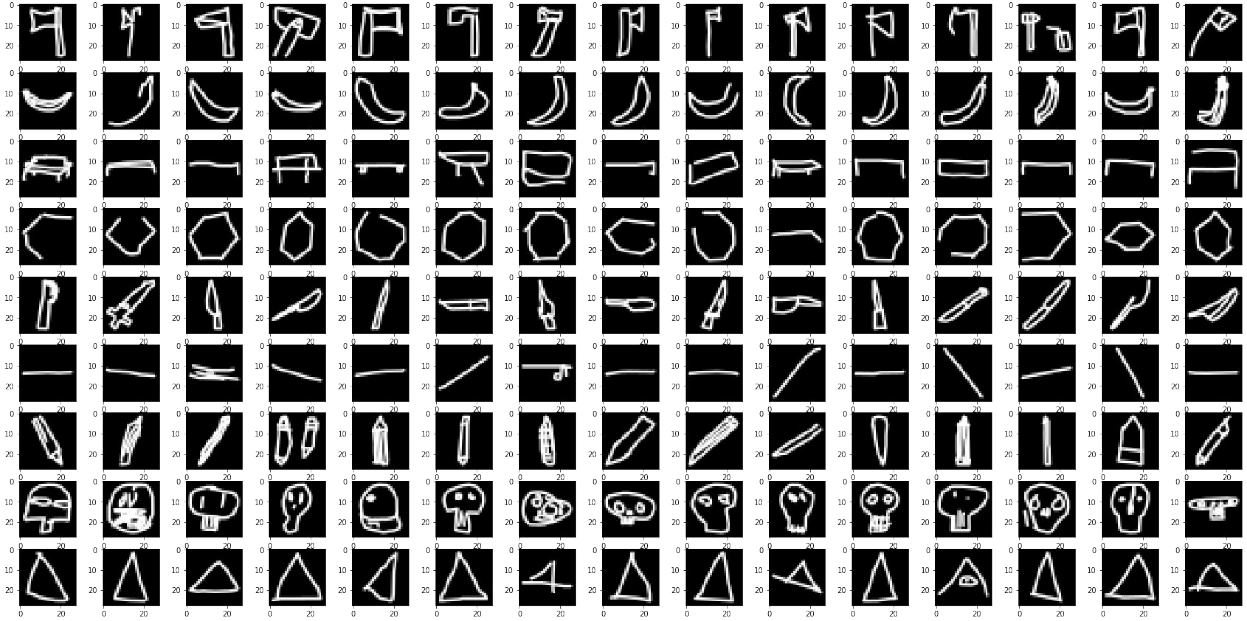
The dataset was obtained from https://quickdraw.withgoogle.com/

Used Categories: axe, banana, bench, hexagon, knife, line, pencil, skull, triangle

Output: Image Category

- 9 categories w/ 1.5 million drawings in total
- Clean Dataset

The Dataset



Train, Test and Validation data split

k-NN & MLP Classifier

- 80% trainset, 20% test set

Convolutional Neural Network

- 80% train set, 20% test set
- k-fold cross-validation w/ k=5

k-Nearest Neighbors (k-NN)

- K-Nearest Neighbors classifier, classifies the data based on its neighbors.

Examples:

- Searching for semantically similar documents - called Concept Search
- Recommender systems like the ones in Amazon, Yelp, Advertisement
- Recognizing drawings/patterns (our project)

Result:

- With k=5, our model achieved an accuracy of 89.6%

Multi-Layer Perceptron (MLP) Classifier

- MLP is a type of feedforward artificial neural network.

Examples:

- Autonomous Driving – ALVINN.
- Pattern Recognition/Computer Vision (Our Project).

Result:

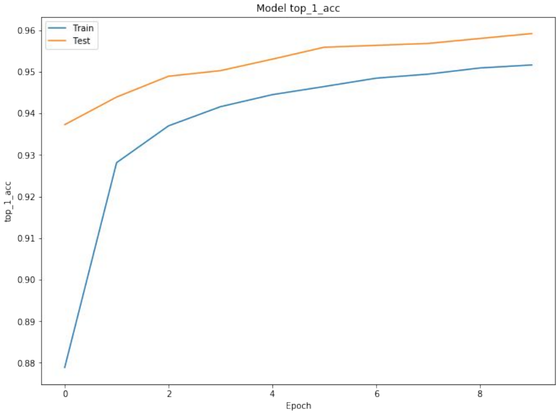
- MLP Classifier from sklearn achieved an accuracy of 82.4%.

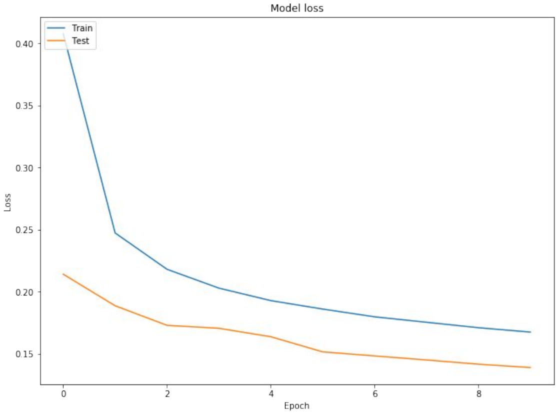
Convolutional Neural Network

- Transforming input
  - The images are available in grey scale (0-255, 8-bit) in a 756-dimensional vector
  - Resizing to a 28x28 matrix to input to the network
- CNN-architecture:
  - 4x(conv-layer + dropout) & 2x(dense-layer + dropout)
  - in total over 3,451,497 trained parameters
  - trained on 10 epochs w/ a batch size of 1024 for each cross-validation

Neural Network Results

- ~3.5 million trainable parameters
- 93% top-1 accuracy
- 98% top-3 accuracy





Deliverables & Requirements (A-grade project)

Dataset Requirements (kaggle, very large)	1.5 million drawings
Platform such as IBM, AWS, Google etc.	Google CoLab
Two ML algorithms developed in Python	done
Powerpoint & in-class Demo	done
Youtube Video	done
train/test data split	done
Explaining the theoretical background behind the learning algorithms	done

What did we learn?

Four key take-aways:

- We learned the theory behind convolutional neural networks and how to apply them
- We learned to develop different Machine Learning models
- We learned to validate them using a method of cross-validation
- We learned to use online platforms for collaborative working and code execution