SIGN LANGUAGE DIGITS DATASET

DEEP LEARNING CAPSTONE

INTRODUCTION TO SIGN LANGUAGE

- **Sign languages** (also known as **signed languages**) are languages that use the visual-manual modality to convey meaning

- This can include simultaneously employing hand gestures, movement, orientation of the fingers, arms or body, and facial expressions to convey a speaker's ideas.

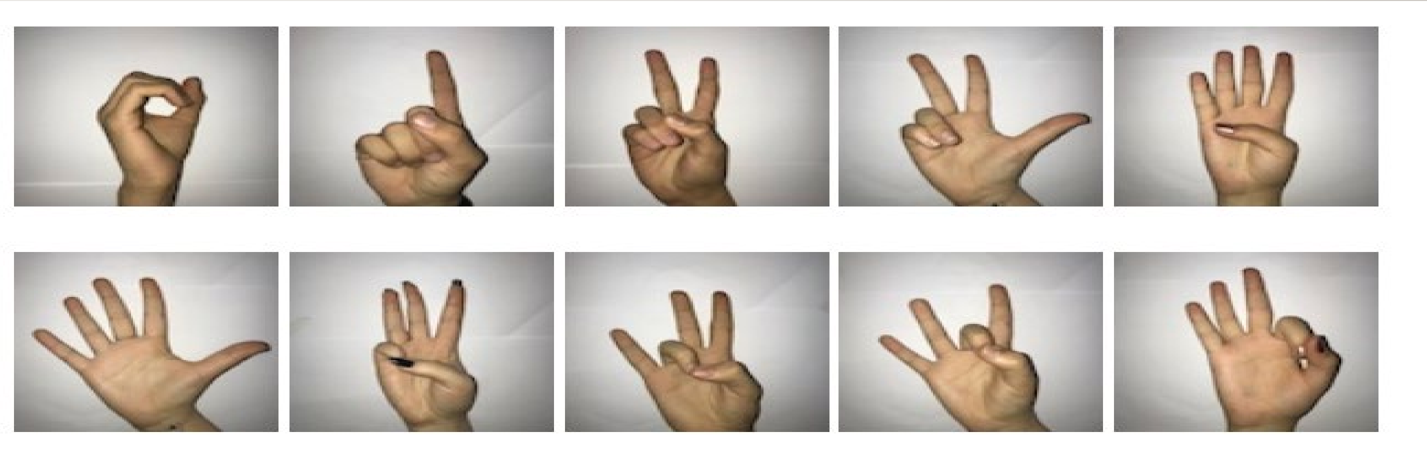
- The aim is to create a model that can predict Sign language digits (0 - 9) from the input images.

- Deep Learning is founded on novel algorithms and architectures for artificial neural networks together with the recent availability of very fast computers and massive datasets enables multi-tasking thereby learning highly informative features.

- Since we were dealing with images, it seemed intuitive to use CNN as it greatly reduces the number of parameters of the model to learn.

SIGN LANGUAGE DIGITS DATASET

- **Sign Language Digits Dataset** dataset prepared by **Turkey Ankara Ayrancı Anadolu High School** students.
- **Details of datasets:**
- Image size: 64x64
- Color space: Grayscale
- Number of classes: 10 (Digits: 0-9)
- Number of participant students: 218
- Number of samples per student: 10
- Below is a Kaggle link to the dataset

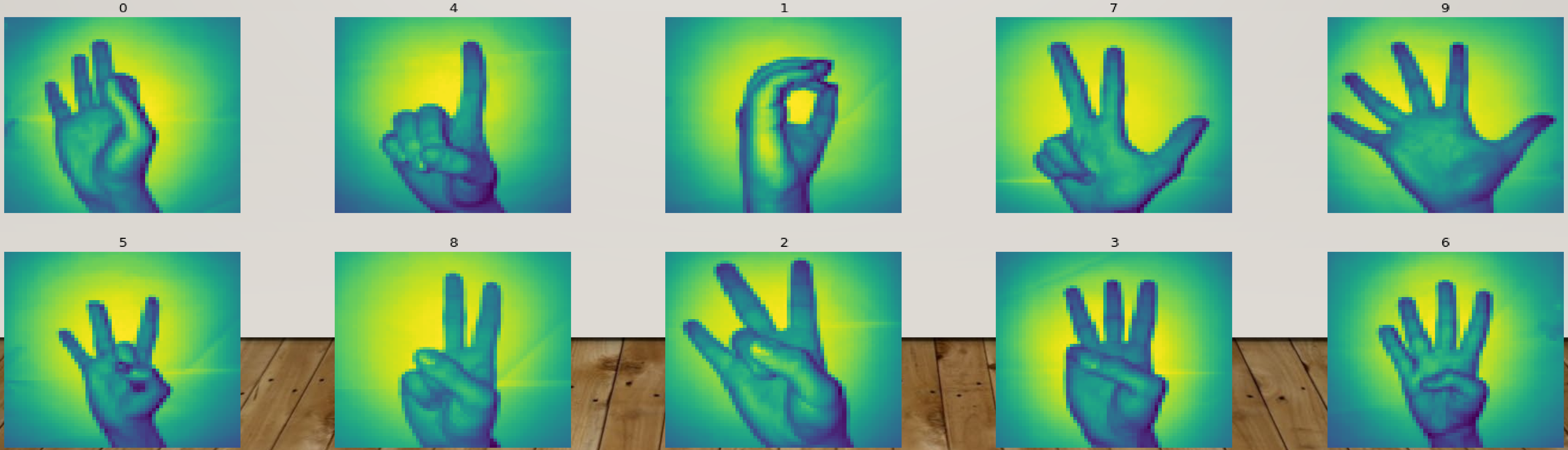


https://www.kaggle.com/ardamavi/sign-language-digits-dataset

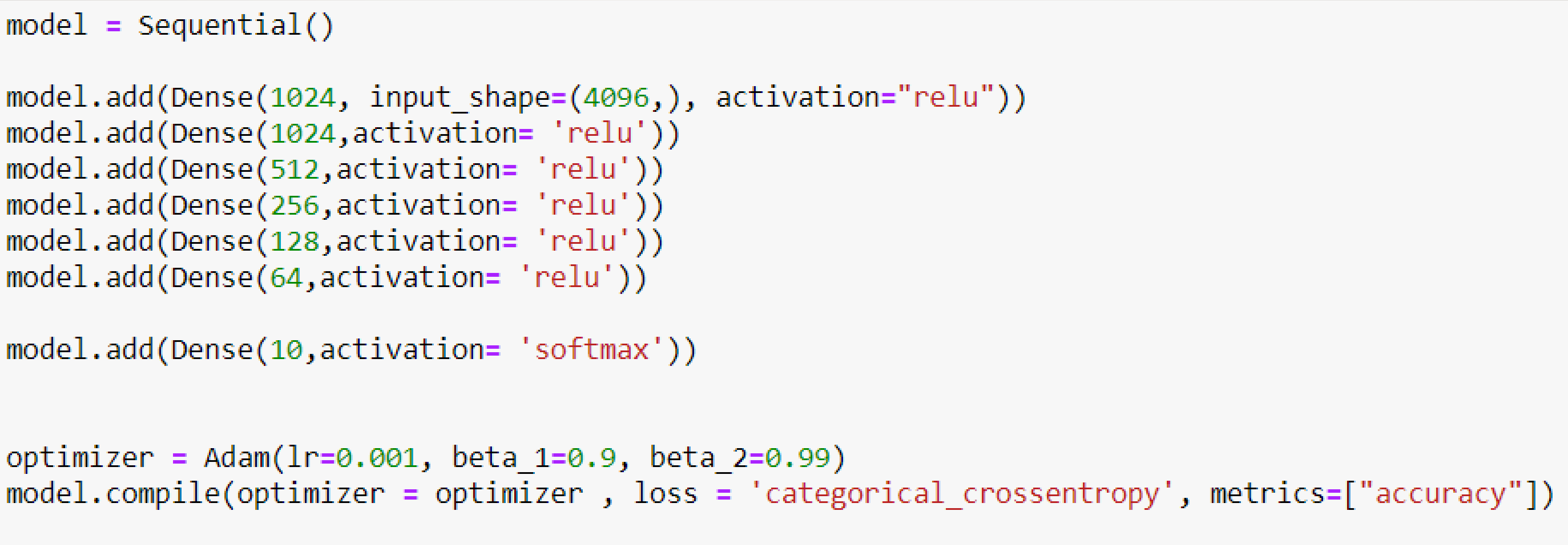
DATA EXPLORATION

- The dataset was presented as separate X and Y files
- The shape of X data is (2062, 64, 64, 1)
- The shape of Y data is (2062, 10)
- The Y is already one – hot – encoded in the file and readily available for analysis
- The data was split into test – train (20 : 80)

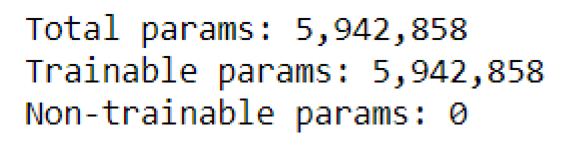
Example of each sign



MODEL PIPELINE



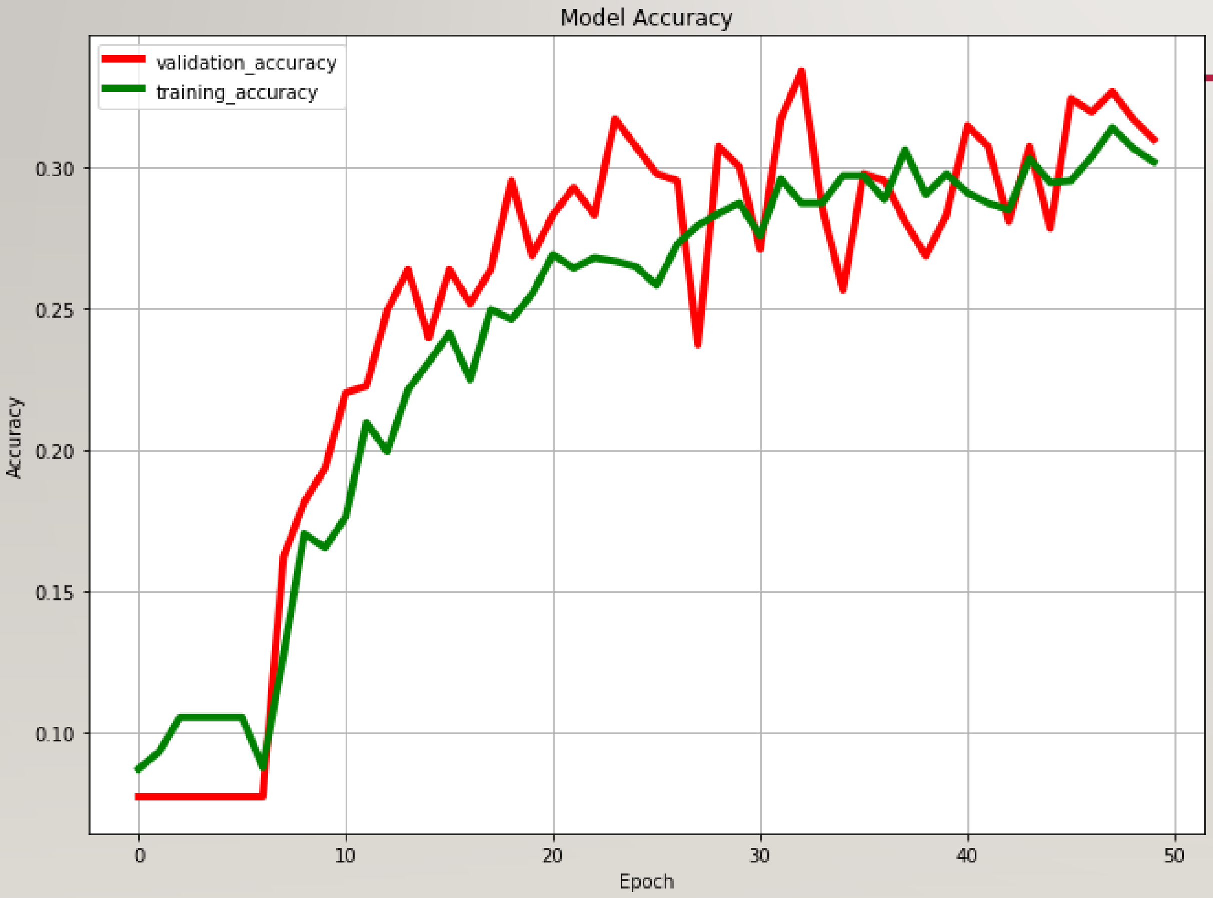
```
model = Sequential()

model.add(Dense(1024, input_shape=(4096,), activation="relu"))
model.add(Dense(1024,activation= 'relu'))
model.add(Dense(512,activation= 'relu'))
model.add(Dense(256,activation= 'relu'))
model.add(Dense(128,activation= 'relu'))
model.add(Dense(64,activation= 'relu'))

model.add(Dense(10,activation= 'softmax'))


optimizer = Adam(lr=0.001, beta_1=0.9, beta_2=0.99)
model.compile(optimizer = optimizer , loss = 'categorical_crossentropy', metrics=["accuracy"])
```



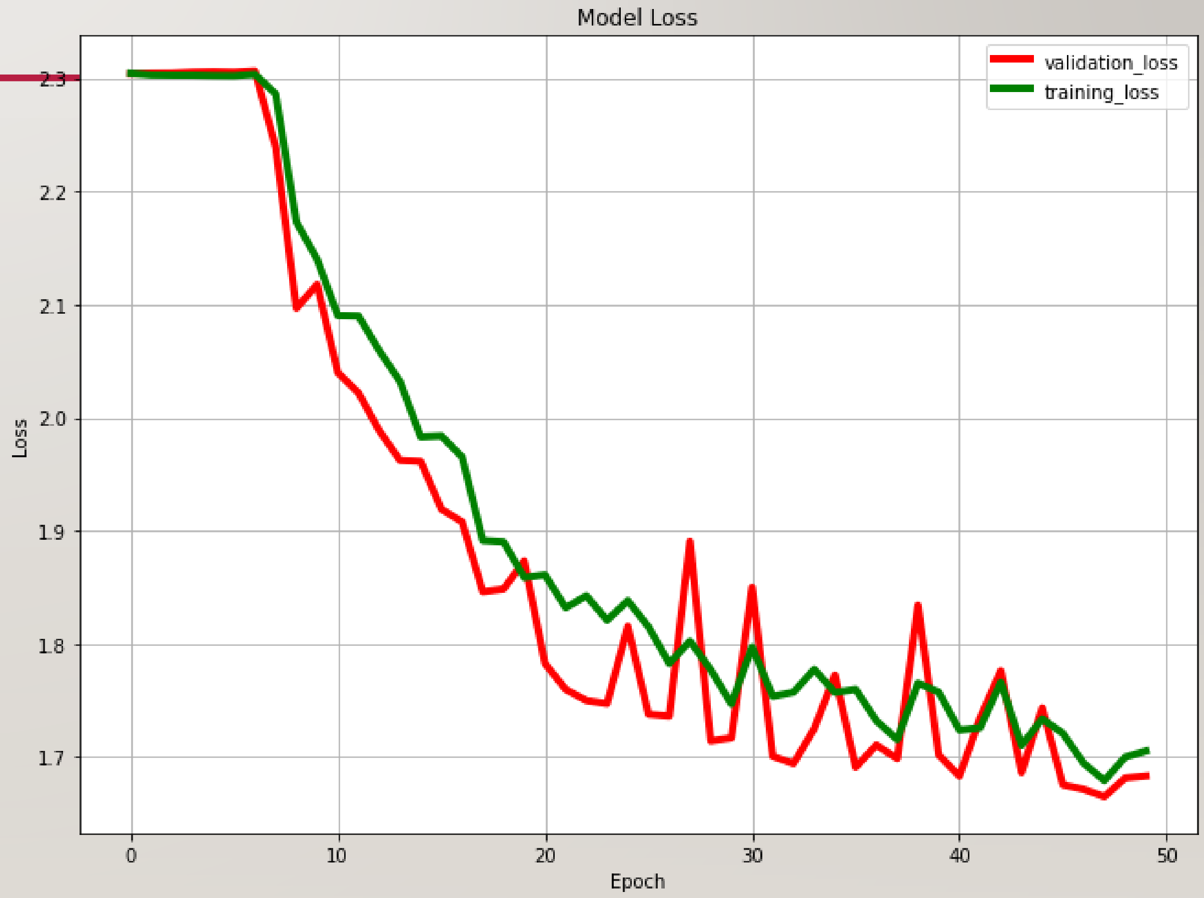
```
Total params: 5,942,858
Trainable params: 5,942,858
Non-trainable params: 0
```

VISUALIZATION OF MODEL RESULTS
ADAM OPTIMIZER

ACC / LOSS



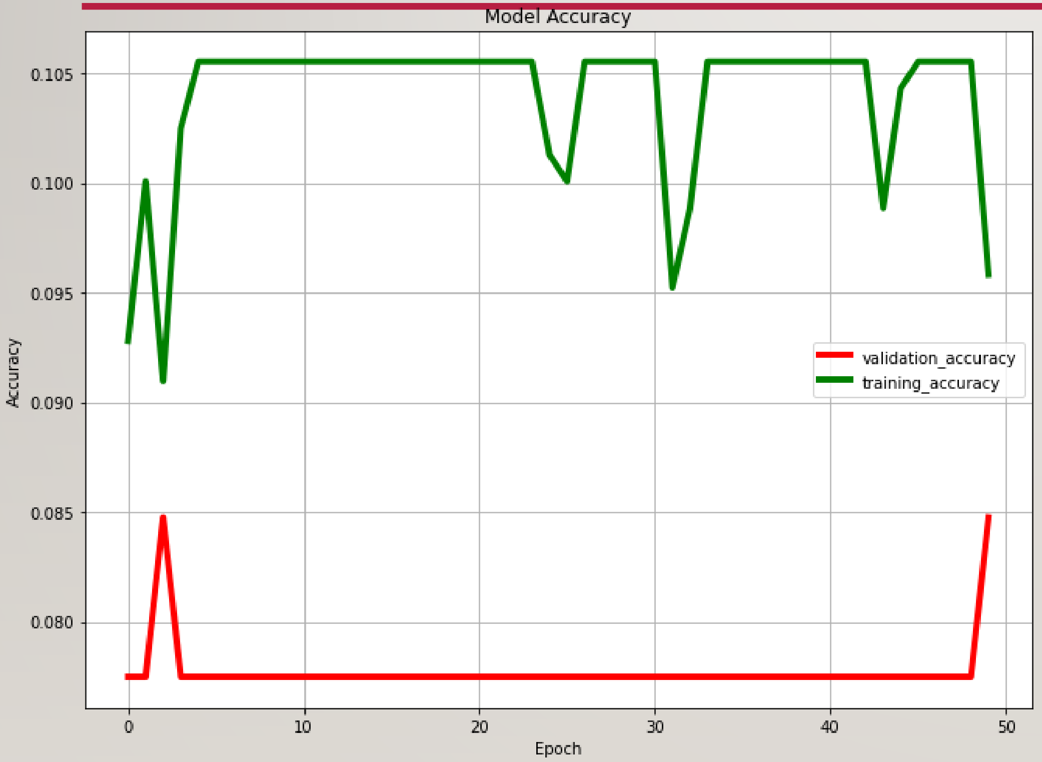
Model Accuracy

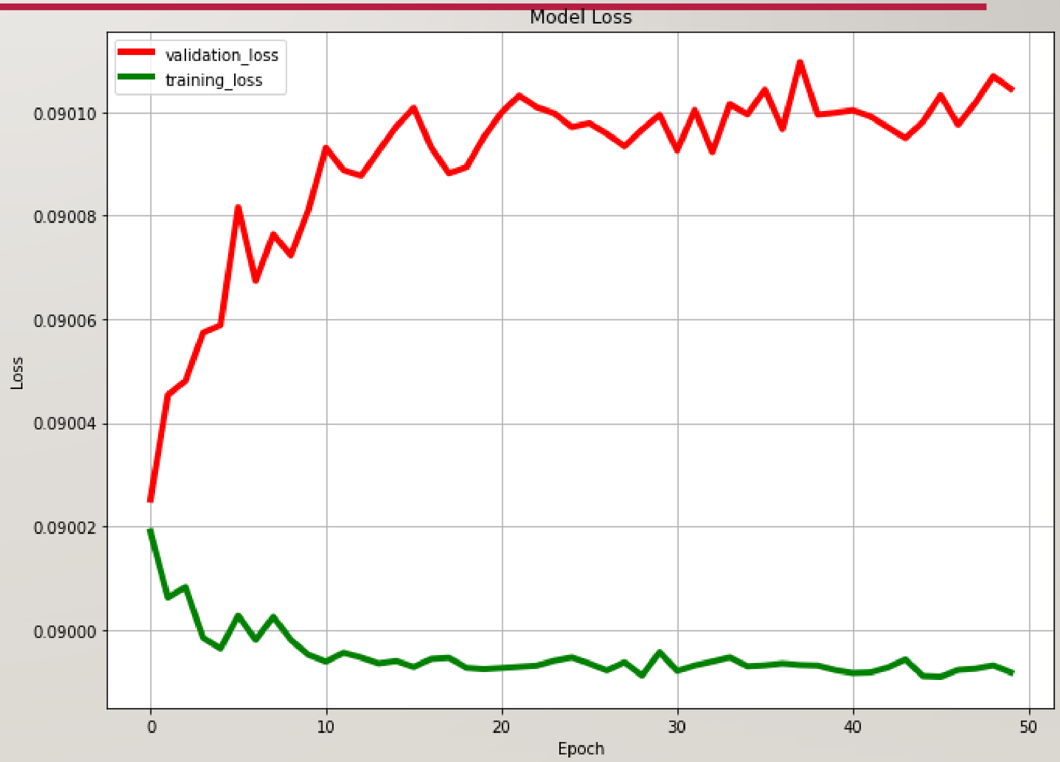


Model Loss

VISUALIZATION OF MODEL RESULTS ADAM OPTIMIZER

ACC / LOSS





OPTIMIZER COMPARISON

ITERATION NUMBER	ADAM	SGD
1	36.08%	7.75%
2	25.91%	7.75%
3	36.56%	7.75%
4	7.75%	7.75%
5	7.75%	7.75%
6	7.75%	7.75%
7	40.68%	7.75%
8	35.11%	7.75%
9	34.14%	7.75%
10	43.34%	7.75%

RECOMMENDATIONS AND CONCLUSIONS

- In this work we created a deep learning model pipeline to study the sign language dataset of digits
- The model was tuned for various loss and activation functions for accuracy
- We chose two different optimizers **Stochastic Descent and Adam optimizers** and obtained results iteratively for statistical variability
- While **Adam optimizer** resulted in statistically rich results for accuracies in the iteration **SGD optimizer** resulted in a more uniform set of values
- Moreover, accuracies with **SGD optimizer** suffered as it got stuck in local minimums at the stage of minimizing the error with frequent updates and performed lower than other optimizers at equal epoch values.
- **Adam Optimizer** gave an accuracy of about **45%** and it is better for the current dataset
- A **more sophisticated modeling pipeline** with convolution, pooling, flatten will have better accuracy than the one currently proposed
- The novel developments in **computer vision** and other deep learning techniques make the development in image classification calls for interesting improvements in the model but it is **beyond the scope** of the **current study** as it is just an **introductory level study in deep learning.**

THANK YOU!