Names: Oren Raz and Lam Kai YI Karen  
Assignment: REI603M Assignment 8 – Project Progress Report

**About the Project**

Táknmálstúlkur is a web-based application with the aim of making sign language translation easily accessible to everyday users. With a simple to use application interface and by harnessing the power of Machine Learning model, Táknmálstúlkur hopes to provide an instantaneous translation from word level American Sign Language to English. This would bridge the communication gap between people with hearing impairment and regular people with little knowledge of sign language.

**Current Progress**

1. Data Collection of 21,095 videos labelled by 2000 words: We have received the full Word Level American Sign Language dataset from the creators of the dataset. Hence, we have access to all the previously missing videos as well. However, due to the scope and timeline of this project we may not be exploring the full dataset.
2. Data exploration and cleaning – removed missing videos, checked video quality, and re-assigned train, validation and test sets.
3. Embedded Data – used mediapipe library to extract key landmark points (543 landmark points per frame in video, each with x, y, z coordinate)
4. Data Engineering – Creation of more train data by shifting hands landmark points in data and laterally flipping landmark points.
5. Model Engineering - Attempted using mean value across all frames, followed by Logistic Regression, Support Vector Machine and Random Forest
6. Model Engineering - Attempted Dynamic Time Wrap (DTW) followed by k-means.
7. Model Engineering - Attempted Recurrent Neural Network – LSTM on subset of train data set

**This Week’s Progress**

1. Model Engineering - Attempted Recurrent Neural Network – LSTM on subset of train data set: We trained the model on a small subset of 200 videos. Each video was embedded to form landmarks via the Mediapipe library. To solve the issue of varying length in videos, 24 frames (evenly spaced out) were selected from each video. The 24 frames of landmark points were fed through 2 LSTM layers in the model during training.
2. Model Deployment – The frontend and backend deployment of the model on a web application was completed this week. The frontend uses HTML and CSS to create a simple to use user interface. Users can submit a short clip on the interface and receive an output of the predicted word and the model’s confidence level regarding the prediction. The backend is a uvicorn server with FasAPI implemented inside a Docker container to ensure the model could be run on different operating systems.

**The Demo**

The “Demo” video attached in the folder showcases the intended user-application interaction when the user uses the web application.

**Challenges Faced and Plans for Future Work**

1. Issue of unclean data: Mentioned in the presentation last week, there are words in the list of 2000 words, which can be signed in multiple ways to convey slightly different meanings. For example, the word “short” has two different ways of signing to convey the message of “short (height)” and “short (time)”. This problem is challenging to solve as it would take time, effort and knowledge of sign language to run through all the data to spot discrepancies in the data. As we are limited by time constraints, we intend to perform the data cleaning on the current small subset only.
2. Challenges with the use of Mediapipe library: The Mediapipe library was not compatible with the use on computers running on the latest Mac Operating Systems. This made the use of the library in the extraction of landmark points challenging. In order to get around this, we had to adopt the use of dockers.
3. Frontend and Backend linkage: The current python code in main.py file is still facing issues with the sharing of information between the frontend interface and backend server. This is one area we intend to work on this week.
4. Exploration of other model implementation – To try to improve the model, we would like to experiment with the use of SigLip this week. We intend to select 9 or 12 frames from each video, arrange the frames in an image and run the new images through SigLip. The output will then be passed through a simple classification model. As SigLip is an established embedding model, we would like to explore the use of SigLip to try to improve the model performance.