Sign Language Translation with Machine Learning

Group 7: Frank Feng (A0074169X), Lee Cheng Shu (A0262750H), Yao Xinyi (A0131172W), Yin Kefei (A0262779L), Zhang Yuqi (A0164665W)

GitHub link: https://github.com/vonVehstolovski/BT5153-Final-Project-Group-7-

Abstract

To break the communication barrier that adversely affects the lives and social relationship of the growing number of deaf communities, this project aimed to develop a sign language translation tool that can recognize American Sign Language (ASL) gestures and sentences. Google - Isolated Sign Language Recognition (GISLR) dataset is obtained from Kaggle, and it is pre-processed to classify ASL through training Neural Network models on labelled landmark data extracted. Upon evaluation, the Transformer model is identified as the best-performing model.

1. Introduction

1.1 Background

Communication is the foundation of human relationships, and it is a tool that enables us to share thoughts and connect with others mainly through listening and speaking. However, a substantial portion of the world's population lacks this ability. The latest report from WHO reveals that over 5% of the world's population – 430 million people, will suffer from disabling hearing loss in 2023. By 2050, this number is expected to jump to 700 million people (WHO, 2023).

The deaf-mute community depends on sign language as the primary means of communication, and it is the most effective and potent way to bridge the communication gap and social interactions with the able people. However, learning sign language is not a walk in the park. It takes on average 2-3 years of regular study and practice to achieve intermediate proficiency (Rose, 2023).

1.2 Problem Statement

Even though sign language interpreters could help to minimize the communication barrier by translating the sign languages into spoken words, it is cost prohibitive in the long run (The Singapore Association for the Deaf). Hence, the deaf community continues to face a lot of difficulties and challenges in their daily interactions.

For instance, research has shown that children who are deaf are more likely to be neglected and face greater loneliness than their hearing peers (Most et.al., 2012) (Xie et.al, 2014), while the adults who are deaf continue to face comparatively higher rates of unemployment in the workplace (Dammeyer et.al., 2019). Furthermore, studies have also shown deafness has prevented individuals from accessing proper health care services (Naseribooriabadi et.al, 2017).

To break the communication barrier that adversely affects the lives and social relationship of the growing number of deaf communities, our team endeavors to create a sign language translator tool. Through machine learning, we hope to ultimately create a near real time system that could recognize individual American Sign Language (ASL) gestures and sentences, thereby radically improving the accessibility for the Deaf and hard-of hearing communities.

2. Dataset Description

2.1 Data Source

The main data source is obtained from Kaggle's dataset, "Google - Isolated Sign Language Recognition" (Google, 2023). The dataset was originally created by Google to identify signs made in processed videos to support mobile apps development.

2.1.1 Google – Isolated Sign Language Recognition (GISLR)

There are several action recognition models available as skeleton-based action recognition has achieved great success recently (Yang, Sakti, Wu, & Nakamura, 2019). Double-feature Double-motion Network (DD-Net) was evaluated and compared against GISLR model. This project has decided to go with GISLR model as it requires less storage, provides better interpretability (Google, Hand landmarks detection guide, 2023) and more robust in performing under different conditions.

2.2 Data Overview

There are 3 files that are obtained from the dataset:

1. train.csv
2. sign_to_prediction_index_map.json
3. train_landmark_files.parquet

2.3 Data from train.csv

Each row of data links to a landmark file for training purpose. It consists of unique labels, corresponding to the label for the landmark sequence. This is a multi-class classification training dataset where each landmark file is trained to be classified into one of the labels.

Table 1. train.csv variables and data description

VARIABLE	DESCRIPTION
path	Path to the landmark file
participant_id	Unique identifier for the data contributor
sequence_id	Unique identifier for the landmark sequence
sign	Label for the landmark sequence

2.4 Data from sign_to_prediction_index_map.json

The purpose of the data is to map 250 unique labels for the landmark sequence to numerical indices. It is used to encode categorical target from text into numerical format.

2.5 MediaPipe Holistic Solution

The MediaPipe Holistic pipeline is developed by Google to enable live perception of simultaneous pose made by human, face landmarks and hand tracking in real-time for different life applications (Google, 2023). It can be used in applications such as detecting and classifying images, recognizing hand gesture and tracking hand landmark/poses, image embedding/segmentation, classifying and embedding texts and audio files.

Figure 1. Example of MediaPipe Holistic (Google, MediaPipe Holistic, 2023)

It generates a total of 543 landmarks, which consist of 33 pose landmarks, 468 face landmarks and 21 hand landmarks per hand (Google, 2023). An example of hand landmarks utilized is shown in Figure 2 below.

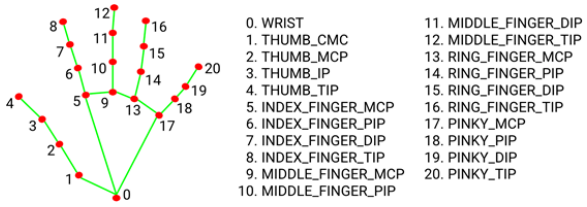

Figure 2. Hand landmark model showing 21 hand-knuckle coordinates (Google, Hand landmarks detection guide, 2023)

2.6 Data from train_landmark_files.parquet

This data source consists of all the landmark data, which were extracted from raw videos with the MediaPipe Holistic model. Each parquet file consists of the variables to illustrate a word. This dataset is used to classify isolated ASL signs with normalized spatial coordinates of the landmark.

Table 2. train_landmark_files.parquet variables and data description

VARIABLE	DESCRIPTION
frame	Frame number in the raw video
row_id	Unique identifier for the row
type	Type of landmark (eg: 'face', 'left_hand', 'pose', 'right_hand')
landmark_index	Landmark index number
[x/y/z]	Normalized spatial coordinates of the landmark. They are the columns that will be provided to chosen model for inference.

3. Exploratory Data Analysis

Exploratory data analysis was performed on the dataset. There are 21702 rows of data in train.csv, which consist of 250 unique signs, corresponding to the label for the landmark sequence.

Among the 250 unique signs, the most frequent label is "duck", which appeared for 105 times. The top 10 labels by row counts are listed in Figure 3.

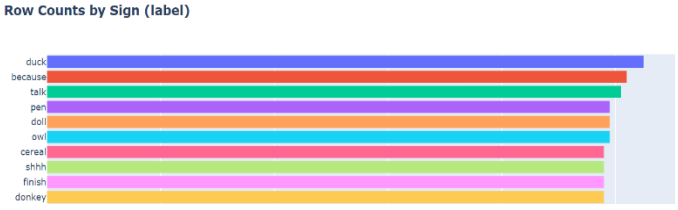

Figure 3. Top 10 labels with highest frequency

The dataset is examined for missing data. Log-frequency of the null values from different types of landmarks are shown in Figure 4.

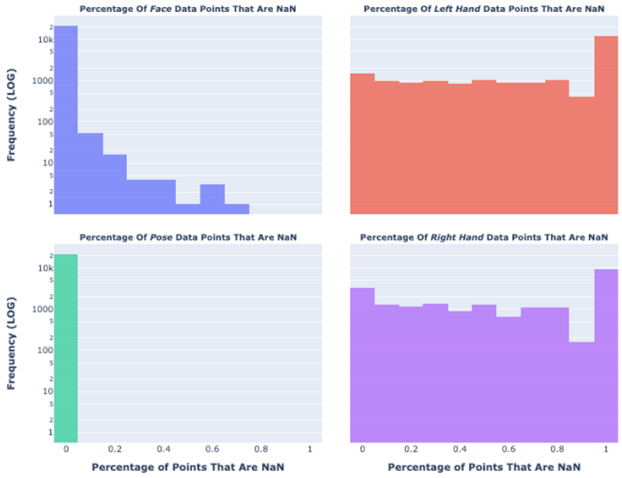

Figure 4. Log-frequency of percentage of null values for different types of landmarks

It is observed that there is no missing data in Pose, majority of Face data is available, while right- and left-hand data has similar probability of being unavailable.

Based on a random sample check from one of the parquet files, Figure 5 shows that there are 468 face landmarks, 33 pose landmarks and 21 hand landmarks for each hand. Left hand sometimes has 0 key points captured in a frame, suggesting that it is not a dominant hand.

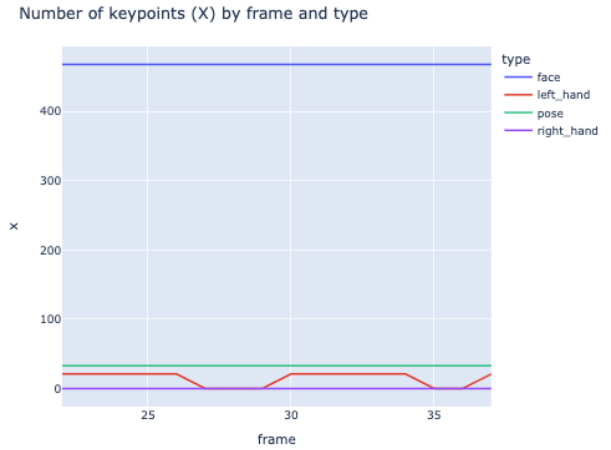

Figure 5. Number of key points by frame and landmark type

4. Convolutional Neural Network (CNN) Model

4.1 Data Pre-processing

Four landmark types are identified from the parquet file, i.e., the left hand, right hand, pose and face. Face has more than 500 key points, and only lip key points are kept as features because other facial features are deemed as irrelevant in sign language recognition. Selected features are listed in Table 3.

Table 3. Feature name with its key points index and count

Feature Name	Key Points Index	Count
Left hand	468 - 489	21
Right hand	522 - 543	21
Pose	502 - 511	10
Lip	0, 13-14, 17, 37-40, 78-82, 84, 87-88, 91, 95, 146, 178-181, 185, 191, 267-270, 291, 308-312, 314, 317-318, 321, 324, 375, 402, 405, 409, 415	40

Since hand gestures are the most important element of sign language, the constructed model will consist of only the hand movements. x and y coordinates of the hand motions were extracted to calculate the variables below and z coordinate is ignored first because it is believed that the depth will not affect sign language recognition. There is a total of 42 input key points from both right and left hand.

Table 4. Processed variables from the parquet file and data description

Variable	Description
n_frame	Number of frames in the data
Keypoints_per_frame	Number of keypoints in a frame
[x,y]	2D coordinates of the keypoint

Using the landmark indices in the original data as reference, the calculated key points are categorized to be either a left-handed or right-handed action. Using Tensorflow, the next step is to determine the dominant hand. In sign language, the dominant hand produces the one-handed signs and 'leads' in the two-handed signs, which is critical for sign language comprehension. Through the comparison of the summed absolute coordinates of the left and right hand, the hand with a greater summed absolute coordinate was deemed to be the dominant hand.

The number of non-empty values in each frame of the dominant hand was calculated. After which, the frames without the dominant hand were filtered out and the indices of the remaining frames was then normalized to start with zero. Lastly, the data was either padded or repeated to fit into a specified fixed length before it is fed into subsequent layers of neural network.

In overall, the pre-processing steps performs several operations to ensure that the input video frames are normalized, filtered and adjusted to a fixed length before these processed data flows into the neural network.

Input: (batch_size, total number of keypoints, [x,y] coordinates)

Output: (input_size, dominant side keypoints, [x,y] coordinates)

4.2 CNN Model Architecture

```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_2 (Conv2D)            (None, 62, 19, 32)        608

max_pooling2d (MaxPooling2D  (None, 31, 9, 32)         0
)

conv2d_3 (Conv2D)            (None, 29, 7, 64)         18496

max_pooling2d_1 (MaxPooling  (None, 14, 3, 64)         0
2D)

flatten_3 (Flatten)          (None, 2688)              0

dense_7 (Dense)              (None, 128)               344192

dropout_4 (Dropout)          (None, 128)               0

dense_8 (Dense)              (None, 250)               32250

=================================================================
Total params: 395,546
Trainable params: 395,546
Non-trainable params: 0
_________________________________________________________________
```

The first model chosen was a convolutional neural network (CNN) constructed with the TensorFlow library. Notably, the softmax function was used in the output layer of the classification neural network to normalizes the output of the last layer into a probability distribution over the various classes. As the probability distribution sum up to 1, the softmax functions helps the neural network to provide a clear and interpretable prediction of the most likely class label for the input.

Also, categorical cross-entropy loss was determined as the loss function. This loss function was designed for multi-classification task and has been known to be effective in training classification neural network, providing a smooth gradient for optimization (Cruttwell et.al., 2022). Finally, the Adam optimization algorithm was selected as it is capable of fast convergence, is well-suited for large datasets and high-dimensional parameter spaces. Moreover, Adam also uses adaptive learning rates and momentum to improve the convergence rate and stability of the training process (Cruttwell et.al., 2022).

4.3 Model Evaluation

After training for 30 epochs, the CNN model has accuracy of 15%, which is approximately 40 times better than random guessing among 250 targets. However, the model prediction accuracy is still low, which could be attributed to the model being unable to capture spatial transformations and also insufficient inputs being fed into the CNN model.

Firstly, since the CNN model was not trained to capture spatial transformation of the input image, it might misclassify an objected after the input image was rotated or translated, thereby significantly impacting the accuracy of the model. Also, only hand gestures were used as inputs in the CNN model, while facial expressions and poses were ignored. Facial expressions and body poses are important non-manual features of sign language that can convey crucial information about the meaning and context of the signs (Pfau & Quer, 2010). For example, facial expressions can indicate emotions while body postures can convey spatial and temporal relationships between signs (Mukushev et.al., 2020). Hence, ignoring these features may have resulted in misinterpretation of signs, leading to poor model performance.

With that, the team explored more complex models to increase the accuracy of the model prediction.

5. The Transformer Model

5.1 Data Pre-processing

The Transformer model is chosen as the second model to be explored and referenced (Wijkhuizen, 2023). In addition to the hand landmarks, pose and lip landmarks are extracted with three-dimensional coordinates. The remaining pre-processing steps are like the previous CNN model.

Table 5. Optimal value for each hyperparameter obtained from hyperparameter tuning.

	CNN Model	Transformer Model
Landmark	Dominant hand	Dominant hand, pose, lip
Key points	21	66
Coordinates	(x, y)	(x, y, z)

5.2 Transformer Model Architecture

5.2.1 Transformer

In the Transformer model, an Attention Head is defined when the Attention module computations are repeated multiple times in parallel. Different parts of input sequence are split and passes independently through a separate Head. Multi-Head Attention model is formed when all these similar Attention calculations are combined to produce a final score (Doshi, 2021). It takes the input sequence as an input and returns an output sequence of the same length.

Multi-head attention allows the model to control mixing of information between input sequences which results in creating richer representations, increasing machine learning task performance (Storrs, 2021).

5.2.2 Embedding Layer

Landmarks are embedded with Landmark Embedding Class. It takes in the number of embedding units and landmark name. The embedded landmarks are then used in the main Embedding Class. The dense layer is performed with two fully connected layers with a GELU activation function, and the weights of the layers are initialized using Glorot Uniform and HE Uniform initialization. If a landmark is missing in frame, the model will apply an empty embedding for the landmark.

With the output from Landmark Embedding as its inputs, Embedding Class takes in embedded instances for lips, left hand, and pose landmarks, and transformed them into embeddings, which are then combined into a weighted mean using landmark weights. It is then passed through two fully connected layers with a GELU activation function in between. The result is then combined with a learnt positional embedding for the frame to form the output embedding of the video input.

5.2.3 Main Model

The main model takes in two tensors as inputs, which are frames and non_empty_frame_idxs. The frames tensor is a 4-dimensional tensor of shape [batch_size, INPUT_SIZE, N_COLS, N_DIMS], while the non_empty_frame_idxs tensor is a 2-dimensional tensor of shape [batch_size, INPUT_SIZE].

Padding is first applied to the frame using a mask which indicates empty frames as 0 and non-empty frames as 1. It then applies random frame masking to the padding while the final mask will be passed to transformers later. The frame tensors are then sliced to extract different parts of the input data (lips, left_hand, and pose), normalized, flattened, and passed them through an embedding layer.

The embedded data is passed through a series of transformer blocks, followed by pooling and treatment with a classifier dropout, before going through the classifier. The output of the classifier is a SoftMax probability distribution over the number of classes. The model is compiled using Sparse Categorical Cross Entropy With Label Smoothing as the loss function, AdamW as the optimizer, and three metrics: Sparse Categorical Accuracy, Sparse Top-5 Categorical Accuracy and Sparse Top-10 Categorical Accuracy. The full structure of model architecture is attached under Appendix.

Input	Tenors: frames, non_empty_frame_idxs
Modelling Steps	- Padding + Random Frame Masks
	- Slicing Frames to obtain landmark data
	- Embedding of landmark dataset
	- Transformer using embedding and mask
	- Average Pooling using mask
	- Classifier Dropout
	- Classifier
Output	Model with Cross Entropy loss function and AdamW Optimizer

Figure 6. Summary of Transformer Model Architecture

5.2.4 Hyperparameter Tuning

Many hyperparameters listed in the model are arbitrarily chosen to get the model running. There are many trade-offs that need to be made in the choice of the hyperparameters.

5.2.4.1 Trade-off between test and validation accuracy

Classifier_dropout, MLP_Dropout (Perception Layer Dropout) and Label Smoothing are hyperparameters that belong to this category. Classifier_dropout is the dropout rate of some target classes in the main model neural network, while Layer dropouts determines the dropouts between layers in Transformer neural network. Label Smoothing is applied to the cross entropy (loss function) to adjust the true label to be a weighted average between the one-hot encoded true labels and a uniform distribution. All of these, if set to 0, can result in the model overfitting on training model.

5.2.4.2 Tradeoff between accuracy and computing capacity (time and cost)

Max learning rate (LR_Max) is used in conjunction with a Learning Rate Scheduler in this neural network training. Higher max learning rate will allow the model to have faster convergence, however, it might miss out the optimal parameter values for accuracy. With limited computing resources, we need to choose an optimal max learning rate, within the limited number of Epochs for training.

By applying fmin() from hyperopt, the optimal values for the four hyperparameters are obtained as shown in Table 6. Due to limited computing resources, hyperparameter tuning needs to be done in two stages, each time with 2 hyperparameters. It is performed with three epochs and a smaller subset of data to avoid prolonged computing time. Optimal values are then used in the final training, with 30 epochs and larger dataset.

Table 6. Optimal value for each hyperparameter obtained from hyperparameter tuning.

HyperParameter	Optimal Value
Classifier_Dropout	0.144
MLP_Dropout	0.196
Label_Smoothing	0.0788
LR_MAX	0.00183

5.3 Model Evaluation

5.3.1 Validation Accuracy

Train accuracy increases quickly to >80%, with increasing epochs within 10 epochs and only sees some slow down around Epoch=30. Validation accuracy seems to plateau at around 70%, after Epoch=10. This implies th
in

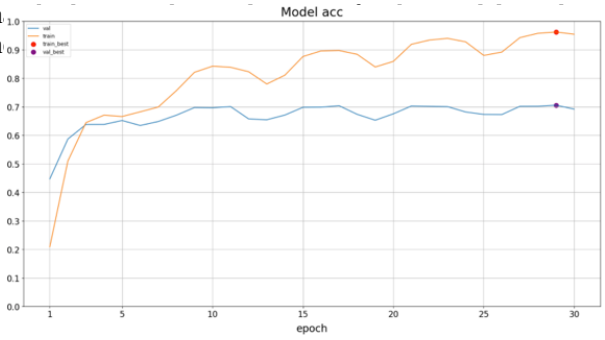

Figure 7. Model Accuracy of Transformer Model with optimized hyperparameters

For the Top 5 training and validation accuracy, it is observed that ~90% accuracy is achieved after 5 epochs. It indicates that the model's top 5 predictions have ~90% probability of inlcluding the true target.

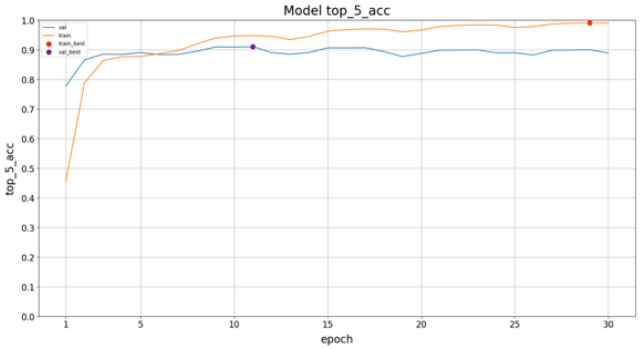

Figure 8. Model Top 5 Accuracy of Transformer Model with optimized hyperparameters

When compared against the previous CNN model's accuracy (15%) and random guess on 250 possible classes accuracy (0.4%), model validation accuracy of 70% seems to be more promising.

Table 7. Comparison of models' accuracy

Model	Model Accuracy
Random Guess	0.4 %
CNN	15 %
Transformer	70 %

5.3.2 Embedding Weights

The final Transformer model has the highest embedding weights with Dominant Hand (left hand). It shows that hand gestures are more important when predicting the correct label of sign language, although both lips and pose play some part in prediction as well.

Table 8. Embedding weights for different type of landmarks

Landmark	Embedding Weights
Lips	29.4%
Dominant Hand	42.5%
Pose	28.1%

6. Limitations and Future Works

In this section, a few limitations in this project will be discussed and potential opportunities are identified to be explored for further studies.

6.1 Data Limitation

There are several limitations from the perspective of the dataset itself. The dataset is very huge (30Gb+). Due to the limited computing resources, only a subset of the original data is utilized in this project for training, even with the use of GPU. For future works, full dataset can be utilized with better computing resources to further improve the existing model architecture. As the existing predicted labels are only 250 for full training dataset, more computing power is required to improve current model and increase the training vocabulary.

6.2 Model Complexity

Given the complexity of the transformer model, it takes a significant amount of time to train the model (~100 minutes). This makes it difficult to iterate quickly and to experiment with different architectures and parameters. Also, with thousands of parameters, the hyperparameter tuning also becomes very time-consuming and require significant computational resources. In the future, automated hyperparameter tuning techniques such as Bayesian optimization can be integrated to solve this problem.

Furthermore, interpretability becomes an issue with such complex models, making it challenging for users to understand how the model are making prediction. The sign language translator tools were intended to aid the communication between people who are deaf and those who are not. Hence, it is important for the model to be interpretable so that users can understand the working and trust the output of the model. On a more practical note, having an interpretable model also helps with debugging and troubleshooting, as potential problems can be easily identified and fixed.

6.3 Potential Future Works

In this project, our final Transformer model can achieve validation accuracy around 70% for all labels and 90% for Top 5. It shows that there are still room for improvement to increase validation accuracy for all labels. Different layers within the neural network can be explored. Pre-trained models such as SignBERT (Hu, Zhao, Zhou, Wang, & Li, 2021) can be utilized to enhance the model accuracy.

As our existing model is trained on ASL, future works can be done to enhance the model training with other public benchmark dataset such as Non-Manual-Feature-Aware Isolated Chinese Sign Language (NMFs-CSL).

In this project, word labels are predicted instead of full sentences. In real life application, Continuous Sign Language Recognition (CSLR) (Sharma, Gupta, & Kumar, 2021) would be more practical as it would be able to understand the context behind the sign language and translate them into complete sentences. Hence, continuous model improvement is required to improve user experience.

7. Conclusion

In this project, the team aimed to develop a sign language translation tool with machine learning, deep learning models. By extracting information from images extracted from raw videos with Google MediaPipe Holistic model, data is classified into 250 labels.

There are 468 face landmarks, 33 pose landmarks and 21 hand landmarks for each hand. These landmarks are then identified with different embedding weights that play a part in the classification predicted labels.

As the raw dataset is complicated to start with, pre-processing is performed prior to building the CNN model for prediction. Model evaluation on CNN model shows that even after training for 30 epochs, it only has an accuracy of around 15%. Hence, the Transformer model is explored subsequently, targeted to improve model's validation accuracy.

Transformer model utilized landmarks such as lips, dominant hand and pose and consist of 66 key points while the previous CNN model only utilized hand landmarks, which consist of 21 key points. Hyperparameter tuning is also utilized for Transformer model to find the most optimal value for hyperparameters on the model. As a result, Transformer model has achieved >90% training accuracy and ~70% validation accuracy after 30 epochs.

It is also identified that there are certain limitations within this project. Data limitation, model complexity and interpretability are challenges that the team faced while working on this project. However, the team believed that this project has laid a strong foundation for future model development. Pre-trained models such as SignBERT can be explored to further improve on model accuracy and increase the vocabulary trained on model by increasing the trained dataset language, as well as developing Continuous Sign Language Recognition (CSLR) for real-life application.

References

Cruttwell, SH, G., Gavranovic, B., Ghani, N., Wilson, P., & Zanasi, F. (2022). Categorical foundations of gradient-based learning. *Programming Languages and Systems: 31st European Symposium on Programming.* Munich, Germany: European Joint Conferences on Theory and Practice of Software ETAPS 2022.

Dammeyer, J., Crowe, K., Marschark, M., & Rosica, M. (2019). Work and Employment Characteristics of Deaf and Hard-of-Hearing Adults. *Journal of Deaf Studies and Deaf Education*, 386-395.

Doshi, K. (2021). *Transformers Explained Visually (Part 3): Multi-head Attention, deep dive*. Retrieved from https://towardsdatascience.com/transformers-explained-visually-part-3-multi-head-attention-deep-dive-1c1ff1024853

Google - Isolated Sign Language Recognition. (2023). Retrieved from Kaggle: https://www.kaggle.com/competitions/asl-signs/overview/description

Google. (2023). *Hand landmarks detection guide*. Retrieved from https://developers.google.com/mediapipe/solutions/vision/hand_landmarker

Google. (2023). *MediaPipe Holistic*. Retrieved from Github: https://github.com/google/mediapipe/blob/master/docs/solutions/holistic.md

Hu, H., Zhao, W., Zhou, W., Wang, Y., & Li, H. (2021). SignBERT: Pre-Training of Hand-Model-Aware Representation for Sign. *eprint arXiv:2110.05382*.

Mukushev, M. S., Imashev, A., Koishybay, K., Kimmelman, V., & Sandygulova, A. (2020). Evaluation of Manual and Non-manual Components. *Proceedings of the 12th Conference on Language Resources and Evaluation* (pp. 6075-6080). European Language Resources Association (ELRA).

Pfau, R., & Quer, J. (2010). Nonmanuals: their grammatical and prosodic roles. *R.Pfau, & J. Quer, Sign languages*, 381-402.

Rose, L. (29 March, 2023). *How Long Does It Take to Learn Sign Language (ASL & BSL)*. Retrieved from Prosperity for All: https://www.prosperityforamerica.org/how-long-does-it-take-to-learn-sign-language/

Sharma, S., Gupta, R., & Kumar, A. (2021). Continuous sign language recognition using isolated signs data and deep transfer learning. *Journal of Ambient Intelligence and Humanized Computing*.

Storrs, E. (2021). *Explained: Multi-head Attention (Part 1)*. Retrieved from https://storrs.io/attention/#:~:text=Multi%2Dhead%20attention%20allows%20for,performance%20on%20machine%20learning%20tasks.

The Singapore Associations For The Deaf, *"Sign Language Interpretation,"* 2023. [Online]. Available: https://sadeaf.org.sg/service/interpreting/. [Accessed 2023].

T. Most, S. Ingber and E. Heled-Ariam, *"Social competence, sense of loneliness, and speech intelligibility of young children with hearing loss*

in individual inclusion and group inclusion," The Journal of Deaf Studies and Deaf Education, vol. 17, no. 2, pp. 259-272, 2012.

T. Naseribooriabadi, F. Sadoughi and A. Sheikhtaheri, "*Barriers and Facilitators of Health Literacy among D/deaf Individuals: A Review Article*," Iranian Journal of Public Health, vol. 11, no. 1465–1474, p. 46, 2017.

Vaughan, A. E. (2023). Deafness and hearing loss. *World Health Organization*.

Wijkhuizen, M. (2023). *GISLR TF Data Processing & Transformer Training*. Retrieved from Kaggle: https://www.kaggle.com/code/markwijkhuizen/gislr-tf-data-processing-transformer-training

Yang, F., Sakti, S., Wu, Y., & Nakamura, S. (2019). Make Skeleton-based Action Recognition Model Smaller, Faster and Better. *In Proceedings of the ACM multimedia asia* , 1-6.

Y.-H. Xie , M. Potměšil and B. Peters, *"Children Who Are Deaf or Hard of Hearing in Inclusive Educational Settings: A Literature Review on Interactions With Peers,"* Journal of Deaf Studies and Deaf Education, vol. 19, no. 4, pp. 423-437, 2014.

Appendix – Transformer Model Architecture