Isolated Sign Language Recognition

Chunhao Li
University of Rochester
cli79@u.rochester.edu

Junfei Liu
University of Rochester
jliu137@u.rochester.edu

Abstract

Every day, 33 babies are born with permanent hearing loss in the U.S. Around 90% of them are born to hearing parents, many of which may not know American Sign Language. Without sign language, deaf babies are at risk of Language Deprivation Syndrome. To help connect deaf children and their parents, the Isolated Sign Language competition is launched to promote a feasible and accessible model to provide feedback for sign language practices. We explore deep learning solutions including transformer, recurrent neural networks(RNN), and ensemble models using TensorFlow Lite with constraints on running time and space for potential application on mobile devices to recognize American sign language(ASL) in videos.

1 Introduction

1.1 Background

According to the American Society for Deaf Children, 33 babies born in the United States every day have permanent hearing loss, and 90% of these children's parents do not have any background knowledge about American Sign Language. Without access to Sign language, these children with hearing loss might end in Language Deprivation Syndrome(LDS), which means they may not develop the necessary skills to assimilate into the academic learning environment successfully[9].

To help children avoid LDS, Google created a smartphone game app called Popsign and launched the Isolated Sign Language Recognition competition on Kaggle to help them improve Popsign. Participants will build machine-learning models to recognize users' body movements and match them with specific signs in the American sign language with time and space limitations with respect to mobile devices' performance. This model will let Popsign detect users' body movements and sign the corresponding sign language vocabulary, which will help both deaf children and their parents to practice sign language[9]. We propose three approaches to training this task-specific model under strict constraints of time and space in this research project.

1.2 Information about American Sign Language(ASL)

Sign languages are natural languages used by deaf and hard-of-hearing people to communicate with each other and with hearing people who know the language. Sign languages have their own unique grammatical rules, vocabulary, and syntax and are not just a visual representations of spoken languages. Like verbal languages, sign languages are complete and complex languages in their own right, with their own grammar and structure. There are various sign languages that exist in the world. Among several sign languages used in the United States, American Sign Language (ASL) is the primary sign language used by the deaf and hard-of-hearing community in the United States and Canada[9].

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

ASL is a visual language that uses a combination of hand gestures, facial expressions, and body language to convey meaning. Signs in American sign language and some other sign languages are more like Chinese characters than alphabets; they represent concepts instead of letters, which makes each concept in these sign languages unique. Its signs are organized into five basic parameters: handshape, movement, location, orientation, and non-manual markers (facial expressions and body language). In standard ASL, a dominant hand will be used for fingerspelling and also for all "one-handed signs."[1] Figure 1 shows the body movements representing the word abandon in American sign language.

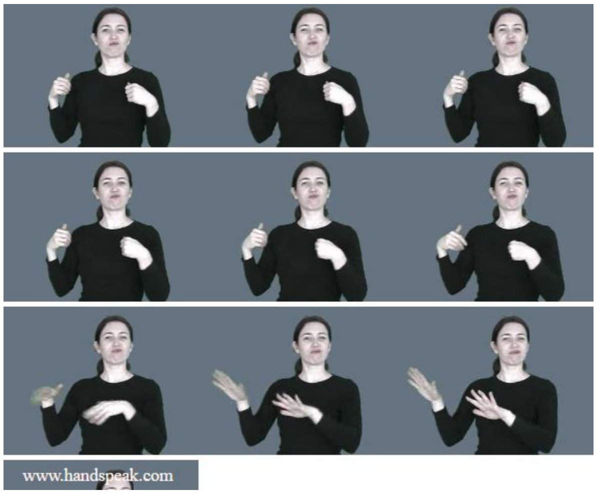


Figure 1: the word Abandon in American Sign Language[20]

ASL is a complex and nuanced language, and learning it requires time, dedication, and practice. Therefore, learning American Sign Language is difficult for parents who are eager to communicate with their deaf children. They want to learn sign language, but it's hard when they are working long hours just to make ends meet. It forms the background that led to the rise of Sign Language Recognition.

1.3 Isolated Sign Language

Broadly speaking, sign language recognition(SLR) can be classified into two primary categories: isolated sign recognition, which pertains to the identification of individual sign gestures, and continuous sign recognition, which involves interpreting sequences of sign gestures that form sentences[12].

Isolated SLR focuses on the identification and interpretation of individual signs, which are often performed in isolation, without any temporal or contextual dependencies. This approach lends itself well to applications that require the recognition of single gestures or signs, such as in educational settings or for assistive technologies. In contrast, continuous SLR addresses the more complex challenge of interpreting entire sentences or phrases in sign language, taking into account the temporal and contextual relationships between signs. This necessitates the development of more sophisticated algorithms capable of capturing the nuances and intricacies of signed communication as it unfolds over time. In either category, a sign gesture typically comprises both hand movements and hand shapes, collectively referred to as manual components[3].

In this project, the aim is to develop a SLR model for educational tasks on single signs, which defines the problem as isolated sign language recognition.

1.4 Information about PopSign's educational games

PopSign is a smartphone game app that makes learning American Sign Language fun, interactive, and accessible[16]. The primary target audience for PopSign includes parents of deaf children, though the application is readily accessible to any individual interested in acquiring ASL vocabulary. The application's learning mechanism entails users pairing ASL sign videos with corresponding bubbles containing written English words, thereby promoting the association between signs and their respective meanings.

To make ASL learning more accessible and engaging for a wider audience, real-time SLR technology is considered to be added to the current PopSign game[16], which is why the Isolated SLR competition was proposed. By adding a sign language recognizer from this competition, PopSign players will be able to sign the type of bubble they want to shoot, providing the player with the opportunity to practice the sign themselves in addition to watching videos of other people signing[9].

1.5 Constraints

To run the machine learning model with limited latency on multiple platforms, including mobile devices, it is required to use TensorFlow Lite[21]. Also, the final model for submission should require less than 40 MB in memory and perform inference with less than 100 milliseconds of latency per video.

2 Litreature Review

In recent years, the emergence of deep learning methodologies has prompted a surge in research aimed at advancing the field of sign language recognition[17]. This phenomenon reflects the growing interest and investment in developing more sophisticated and accurate systems that can effectively interpret and understand sign language communication.

In concern to the acquisition process, the SLR system is classified as the sensor-based and vision-based approaches[13]. The sensor-based approach involves the use of physically attached sensors to capture the movement and trajectories of the signer's head, fingers, and body. Sensor-equipped gloves are employed to monitor the signer's hand articulations, enabling the recognition of specific signs as they are performed. In contrast, the vision-based approach relies on the use of cameras to capture and recognize gestures through vision or image-based techniques. By analyzing the acquired images or videos, the system extracts features related to the movement of the palm, fingers, and hand. Utilizing these extracted features, subsequent classification is conducted to identify and interpret the sign gestures. Researchers had conducted various research on SLR with both systems.

Classical manual SLR was proposed early in the field with attempts using sequential Pattern Tree-based multi-class classifier, Multi-Stream Hidden Markov Models(HMM) classifier, and Support Vector Machines[13]. Bauer and Heinz from the Aachen University of Technology achieved prominent results using HMM with hand shape, hand orientation, and location as feature vectors[2]. It is important to note that their approach solely utilizes the shape and orientation of the dominant hand as input. Given that the research was conducted in 2000, they employed a colored glove to assist their model in discerning hand shapes; however, this method proved to be inefficient and incapable of tracking the status of individual fingers. Additionally, the dataset used in the study was relatively small, comprising only 97 distinct signs and featuring videos for training and testing data collected from a single individual. These limitations may have contributed to the wide range of accuracy rates observed, which spanned from 94% to a mere 2.2%. Dahamani and Larabi developed a model utilizing a Support Vector Machine (SVM) classifier to facilitate the recognition of finger patterns in sign language fingerspelling for images[4]. By employing geometrical features, such as relative area and distance, the model was better equipped to accurately identify hand shapes. The performance of their model proved to be noteworthy, with accuracy rates ranging from 85% to 97% across three datasets. Impressively, the model maintained a performance level above 90% even under complex conditions.

Deep learning based approaches with raised with advancements in hardware support and the architecture of convolutional neural networks(CNN) for feature extraction[13]. Diverse research based on single and fusion parallel 3D CNN was conducted. Sincan and Keles performed CNN and LSTM based SLR model for Turkish SLR with the feature extraction improved by FPM (Feature Pooling Module), and convergence speeded up using the attention model[19]. Yuan et al pointed out deep convolution neural network(DCNN) and LSTM based model for hand gesture recognition. The residual module has overcome the gradient vanishing and overfitting problem. Complex hand gesture long-distance dependency problem addressed by improved deep feature fusion network.[25]

Sign recognition from videos still presents a significant challenge, as the meaning of a word relies on a composite of subtle body movements, hand configurations, and various other gestures. However, current pose-based frameworks for SLR can either simultaneously model both spatial and temporal dependencies among poses in distinct frames or solely focus on temporal information without fully capitalizing on the available spatial information.[23] In order to enhance the accuracy of SLR, various features can be combined, which can be broadly classified into three categories incrementally: 1) utilizing solely hand pose features, 2) incorporating both hand and facial pose features, and 3) integrating hand, facial, and body pose features[17]. By fusing all these distinct elements, researchers can develop more robust and precise recognition systems that are better equipped to interpret sign language communication. Because of the nature of the MediaPipe Holistic solution we employed for data input in this task, pose-based methodologies will constitute our primary references.

Anirudh, Sai, Juan introduce a novel pose-based approach that separately captures spatial and temporal information, followed by a late fusion process[23]. They proposed an architecture that explicitly extracts spatial interactions within the video using a Graph Convolutional Network (GCN). Meanwhile, temporal dependencies between frames are captured through the implementation of Bidirectional Encoder Representations from Transformers (BERT). Experimental outcomes on WLASL, a standard word-level SLR dataset, reveal that their model surpasses the state-of-the-art in pose-based methods by achieving up to a 5% improvement in prediction accuracy.

In 2022, Iyer and his colleagues developed a pose-based sign language detection model based on long short-term memory (LSTM) networks[10]. Their approach successfully predicted dynamic signs using three sets of LSTM layers with ReLU activation functions. They achieved a high accuracy rate of 90.8% for the training data and 87.5% for the test data set. However, the authors did not provide any details regarding their dataset. Moreover, based on the examples included in the article, it appears that their model is limited to three output classes.

In Kim et al.'s study, titled "Global-Local Motion Transformer for Unsupervised Skeleton-Based Action Learning"[11], the authors developed a transformer for unsupervised motion recognition called the GL-Transformer. This transformer incorporates both global and local attention mechanisms, as well as a novel multi-task learning strategy called multi-interval pose displacement prediction. This approach enables the GL-Transformer to predict multiple pose displacements across different intervals concurrently. Furthermore, the new strategy separately tracks the body's central joint and other body joints, such as fingers and arms. By introducing these innovative features to the GL-Transformer, the authors successfully achieved an accuracy rate exceeding 80%.

3 Data Insights

Collaborating with Google, the Georgia Institute of Technology's dataset focuses on landmarks extracted from raw videos using the MediaPipe Holistic model[14]. This extensive dataset includes 94,477 distinct body movements and 250 unique ASL signs, significantly surpassing the scope of prior research. The dataset consists of the frame number from the original video, landmark category (face, left hand, pose, right hand), landmark index, and normalized spatial coordinates[9].

To ensure the reliability and generalizability of the results obtained from analyzing this dataset, the potential bias is minimized by collecting videos from a diverse group of 21 signers from various regions across the United States. This approach aimed to capture the nuances and variations in signing styles, regional accents, and dialects present in American Sign Language. By incorporating signers with different backgrounds, the dataset becomes more representative of the broader ASL-using population, enhancing the model's ability to learn and recognize a wide range of signs[9].

3.1 Number of Samples per Sign

The number of samples per sign is close to a uniform distribution as shown in figure2. Each sign contains at least 299 samples and no more than 415 samples and the mean number of samples per sign is around 378 with a small standard deviation as shown in figure3. The uniform distribution of the number of samples per sign in the dataset is an important factor that can greatly affect the performance and generalizability of the ASL recognition model. By ensuring that each sign has a similar number of samples, the model is less likely to overfit specific signs and more likely to generalize to a wide range of signs.

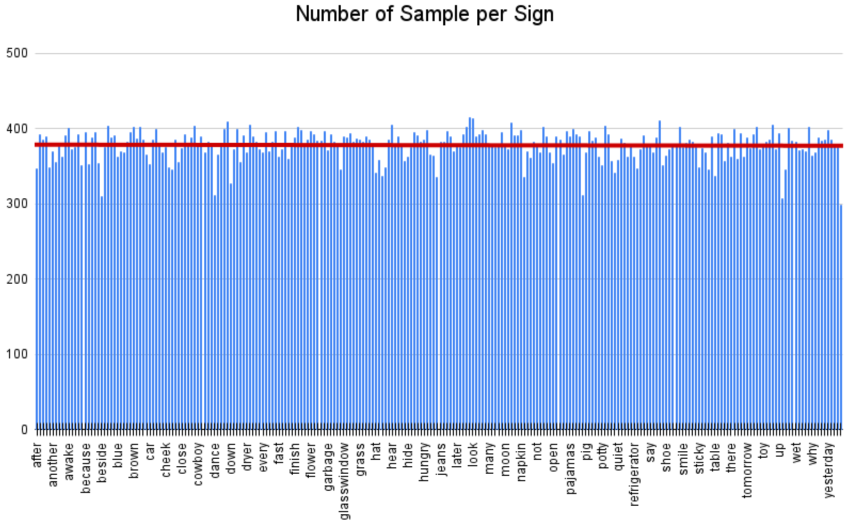


Figure 2: Number of Samples per Sign

3.2 Frame Distribution

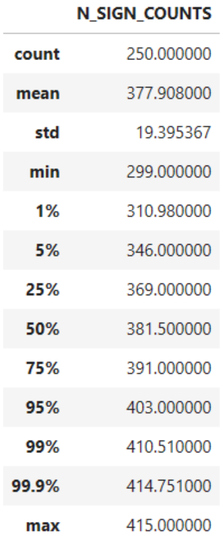
The distribution of the number of unique frames per sequence over the dataset is an important factor to consider when analyzing and processing data in a video-based study. It provides insight into the variability and consistency of the recorded sequences and can also affect the performance of models used for analysis. As we processed the dataset, we found that the distribution of unique frames is not Gaussian distributed and falls into a right-skewed distribution shown in figure 4. 50% sequences in the dataset contain less than 22 frames, with the minimum number of frames in a sequence being only 2, the maximum frame being 537, and the mean number of frames in sequences being 37.935.

Besides, there is a small number of samples with missing frames as shown in figure 5.

3.3 preprocessing

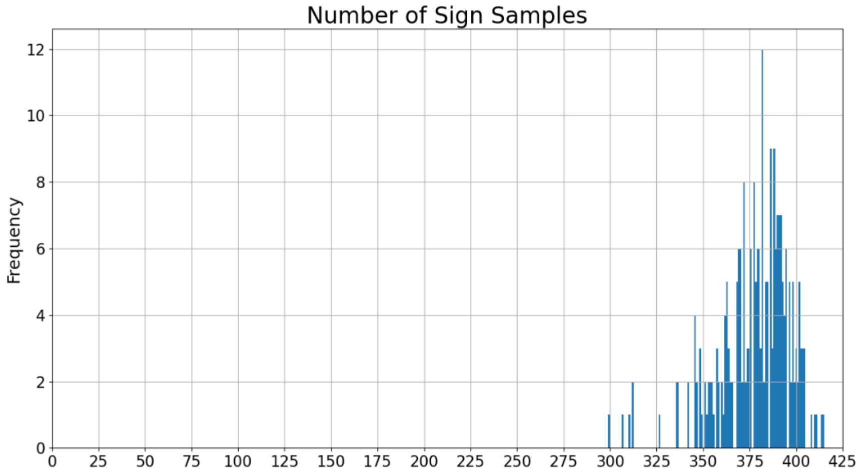
The dataset provided by Google Isolated Sign Language Recognition competition processed by MediaPipe is exceptionally high with the exploratory data analysis above. To modify the dataset to better suit our need, we preprocessed the dataset with the following operations:

1. We removed the Z value in coordinates. Z represents the landmark depth with the depth at the center of the head, hand, or body being the origin, and the smaller the value the closer the landmark is to the camera. However, the MediaPipe Holistic model is not fully trained to predict the depth and



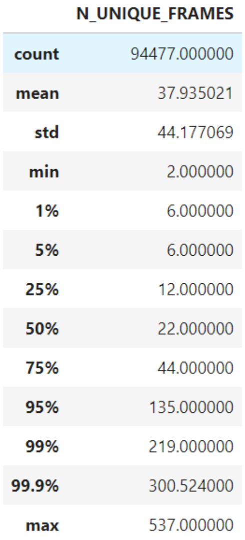
	N_SIGN_COUNTS
count	250.000000
mean	377.908000
std	19.395367
min	299.000000
1%	310.980000
5%	346.000000
25%	369.000000
50%	381.500000
75%	391.000000
95%	403.000000
99%	410.510000
99.9%	414.751000
max	415.000000

(a) Sign Counts Statistics



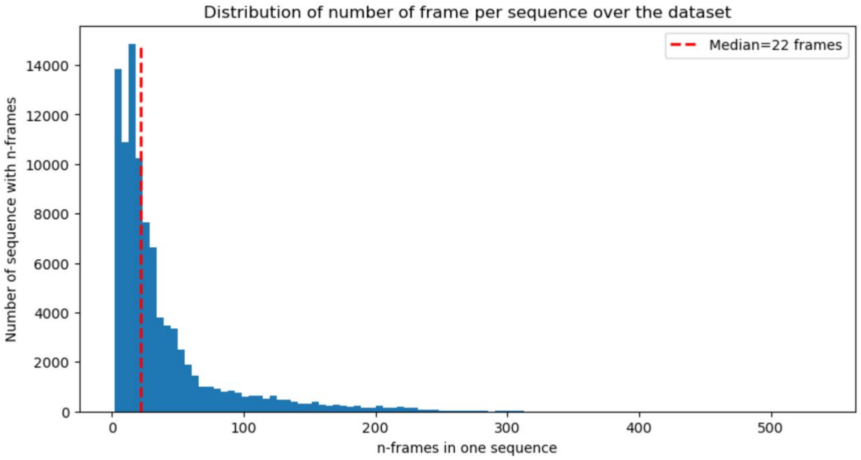
(b) Sign Counts Distribution

Figure 3: Sign Counts



	N_UNIQUE_FRAMES
count	94477.000000
mean	37.935021
std	44.177069
min	2.000000
1%	6.000000
5%	6.000000
25%	12.000000
50%	22.000000
75%	44.000000
95%	135.000000
99%	219.000000
99.9%	300.524000
max	537.000000

(a) Unique Frame Statistics



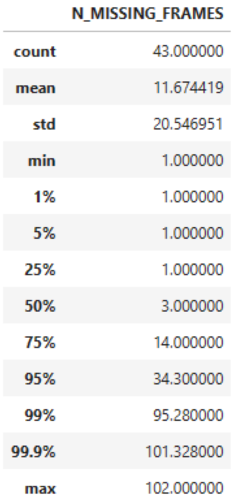
(b) Unique Frame Distribution

Figure 4: Unique Frames

it is instructed to discard the z axis for better model performance[15].

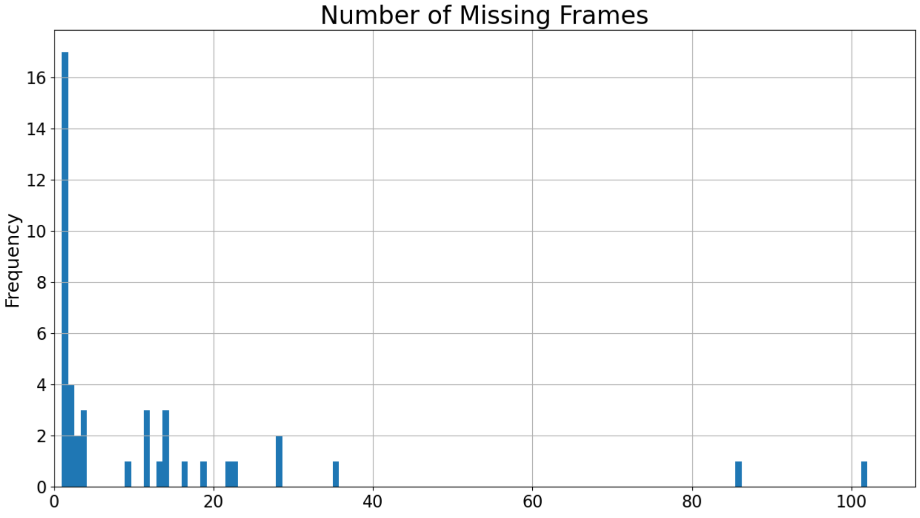
2. Find and save non-empty frame indices. Although there is a limited number of empty frames, we decide to minimize its impact and only use non-empty frames for model training and validation.

3. Find the dominant hand and normalize hand positions. It is crucial to identify the dominant hand in ASL to precisely tell the meaning of a series of actions. We identify the dominant hand by counting its presence in frames and normalize its position to the center, i.e. add 0.50 to left hand (original right hand) and subtract 0.50 of right hand (original left hand), to balance left-handed and



	N_MISSING_FRAMES
count	43.000000
mean	11.674419
std	20.546951
min	1.000000
1%	1.000000
5%	1.000000
25%	1.000000
50%	3.000000
75%	14.000000
95%	34.300000
99%	95.280000
99.9%	101.328000
max	102.000000

(a) Missing Frame Statistics



(b) Missing Frame Distribution

Figure 5: Missing Frames

right-handed signers.[1]

After careful consideration and preliminary experiments, data augmentation was not adapted in the preprocessing step. In the context of SLR, employing data augmentations can be particularly challenging. This is because even subtle differences in finger, hand, or arm configurations and positions can lead to the interpretation of an entirely distinct sign. Consequently, while augmentations can be an effective technique for expanding training datasets and improving model generalization in other domains, it is not recommended to apply them to sign language data without specific prior knowledge of the dataset[6].

4 Methodology

We propose three distinct branches of attempts at this problem: transformer-based approach, recurrent neural network(RNN) based approaches, and an ensemble of available models. With specific constraints of running time and space of the model, we were unable to find a baseline for this study. Therefore, a comparison between different methods will constitute our research question.

4.1 RNN-based method

Due to their sequential nature, Recurrent Neural Networks (RNNs) are well-suited for processing sequential data, such as time series or speech signals[18], and therefore an ideal option for SLR. RNNs can maintain a hidden state that captures the temporal dependencies between the input data, allowing them to recognize patterns in the sequence of gestures over time. This makes RNNs particularly useful for recognizing sign language, where the meaning of a sign is often dependent on the context and sequence of other signs around it. By processing the input sequence of gestures with an RNN-based model, we can effectively capture the temporal dependencies and recognize sign language.

4.1.1 GRU

The first RNN-based method we used implemented is Gated recurrent unit (GRU). The GRU Model is developed using the TensorFlow framework. The model consists of a single GRU layer with 512 GRU units and a dropout layer with a 0.5 dropout rate to prevent overfitting. The output of the GRU layer is connected to a residual block and then fed into a Multi-Scale Dilated Convolution (MSD) layer with Softmax activation function as the final output layer of the neural network model as shown in Figure 6.

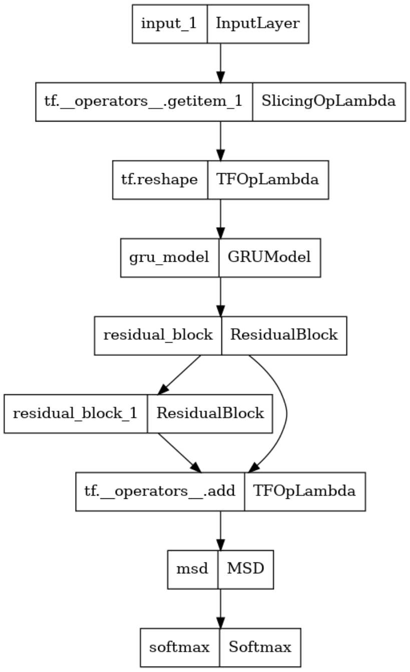

input_1 | InputLayer
tf.__operators__.getitem_1 | SlicingOpLambda
tf.reshape | TFOpLambda
gru_model | GRUModel
residual_block | ResidualBlock
residual_block_1 | ResidualBlock
tf.__operators__.add | TFOpLambda
msd | MSD
softmax | Softmax

Figure 6: GRU Model Structure Generated by TensorFlow

4.1.2 LSTM

The other proposed RNN-based methodology for sign language recognition involves using a neural network architecture based on Long Short-Term Memory (LSTM) cells. The LSTM model comprises several layers, including an Average Pooling 2D layer that reduces the spatial dimensions of the input while preserving the number of landmark points. This reshaping layer converts the tensor into a sequence of vectors, an LSTM layer that learns the long-term dependencies in the sequence, an Average Pooling 1D layer that reduces the temporal dimension of the output while preserving the number of output features, and a fully connected layer with a softmax activation function that produces the final output of the model, as shown in Figure 7.

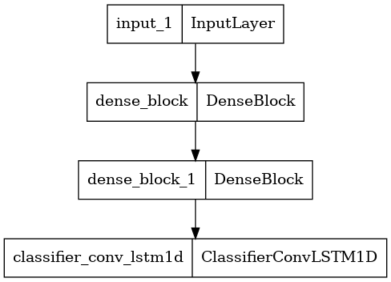

input_1 | InputLayer
dense_block | DenseBlock
dense_block_1 | DenseBlock
classifier_conv_lstm1d | ClassifierConvLSTM1D

Figure 7: LSTM Model Structure Generated by TensorFlow

4.2 Transformer-based method

The Transformer architecture, introduced by Vaswani et al. in 2017[24], represents a significant advancement in the field of sequential data processing. However, in contrast to RNNs, Transformers simultaneously process the entire input sequence. The attention mechanism delivers contextual information for any given position within the input sequence, which eliminates the need for recurrent connections and enables highly parallelized training and significantly improves the scalability of the model.[24]. This makes the transformer model highly efficient and effective for modeling complex sequences of data.

In sign language recognition, transformers could be used to learn the underlying structure of sign language gestures and recognize different signs and sequences. To achieve that goal, several types of embeddings are integrated into the transformer model: lips embedding, hand embedding, and pose embedding for different parts of landmark data, learnable landmark weights, and positional embedding for the frame index in the sequence of frames, as shown in Figure 8.

In a Transformer model, positional embeddings generally play a crucial role in encoding the sequential information present in the input data since the Transformer architecture processes the entire input sequence in parallel. In this project, we will also investigate whether the transformer model's performance will be significantly impaired due to a lack of understanding of the elements' order within the input sequence. Specifically, experiments will be conducted with and without the positional embedding enabled in the transformer model and their accuracy will be compared.

4.3 Attempts to ensemble

Ensemble techniques are strategies that combine the predictions of multiple base models to enhance overall performance and accuracy. By leveraging the strengths and mitigating the weaknesses of individual models, ensemble techniques often yield superior results compared to relying on a single model. We employed several public notebook solutions on the Kaggle competition for ensemble including two existing ensembles of various outperforming models[5][7] and an ensemble of LSTM and DNN models[22] with reference to the best ensemble model on the leaderboard[8].

5 Experiments

5.1 RNN model

5.1.1 GRU

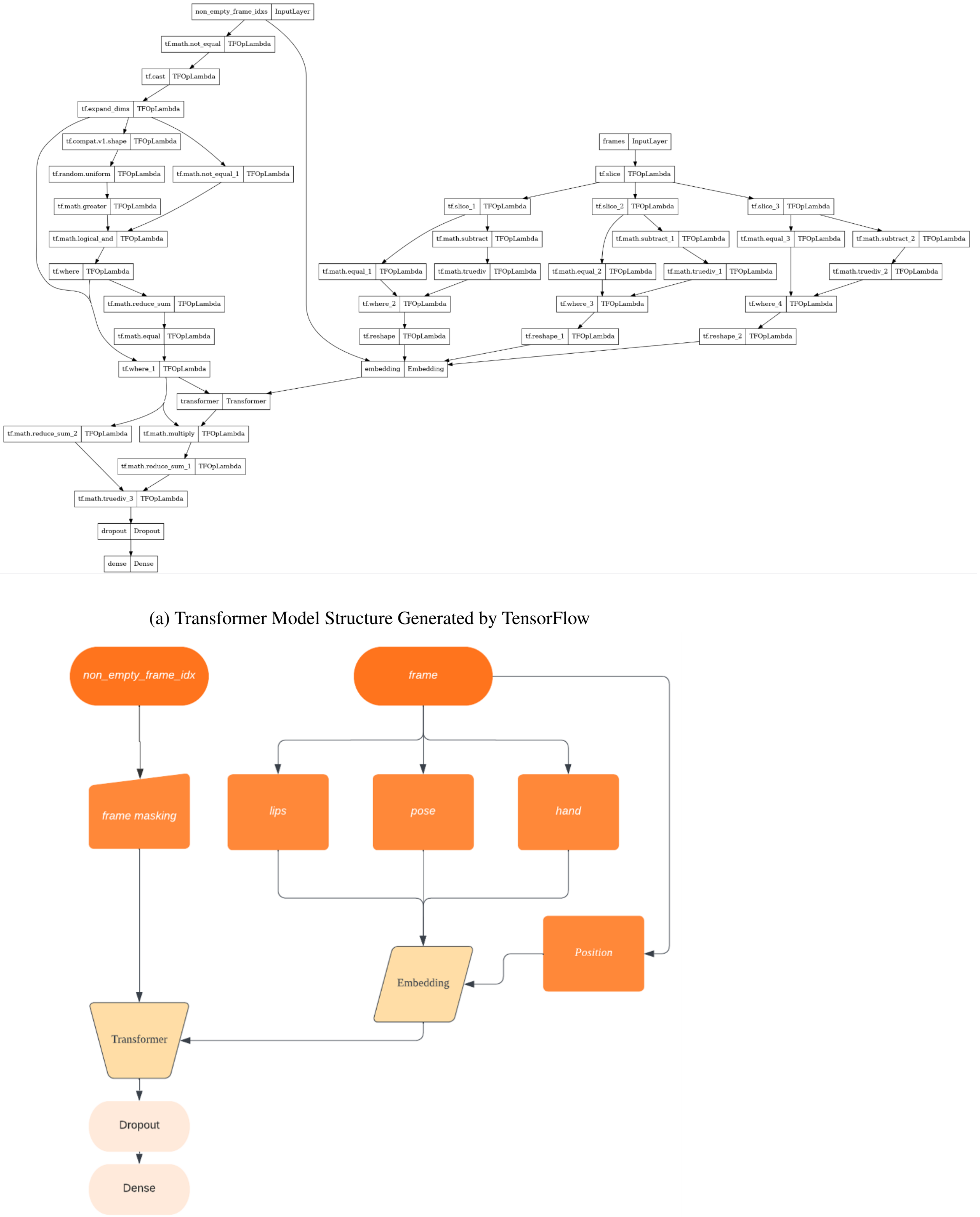
Our implementation of the GRU model yielded impressive results within the context of the current dataset. Achieving a 93% accuracy on 80% of the dataset utilized for training and a remarkable 99.3% for the top 5 categorical accuracies in the training dataset. Moreover, the GRU achieved a 75% accuracy on the test dataset as shown in Figure 9.

5.1.2 LSTM

In comparison to the GRU, the performance of the LSTM model is notably inferior. The best performance out of several trails of training on 80% of the dataset remained below 80%, and a significant drop in performance was observed on the test dataset, with accuracy plummeting to approximately 40%. This represents the largest discrepancy between test and training dataset performance across all models utilized in this study. The accuracy rate and loss over epochs for the model can be found in Figure 10.

5.1.3 Conclusion over RNN-based methods

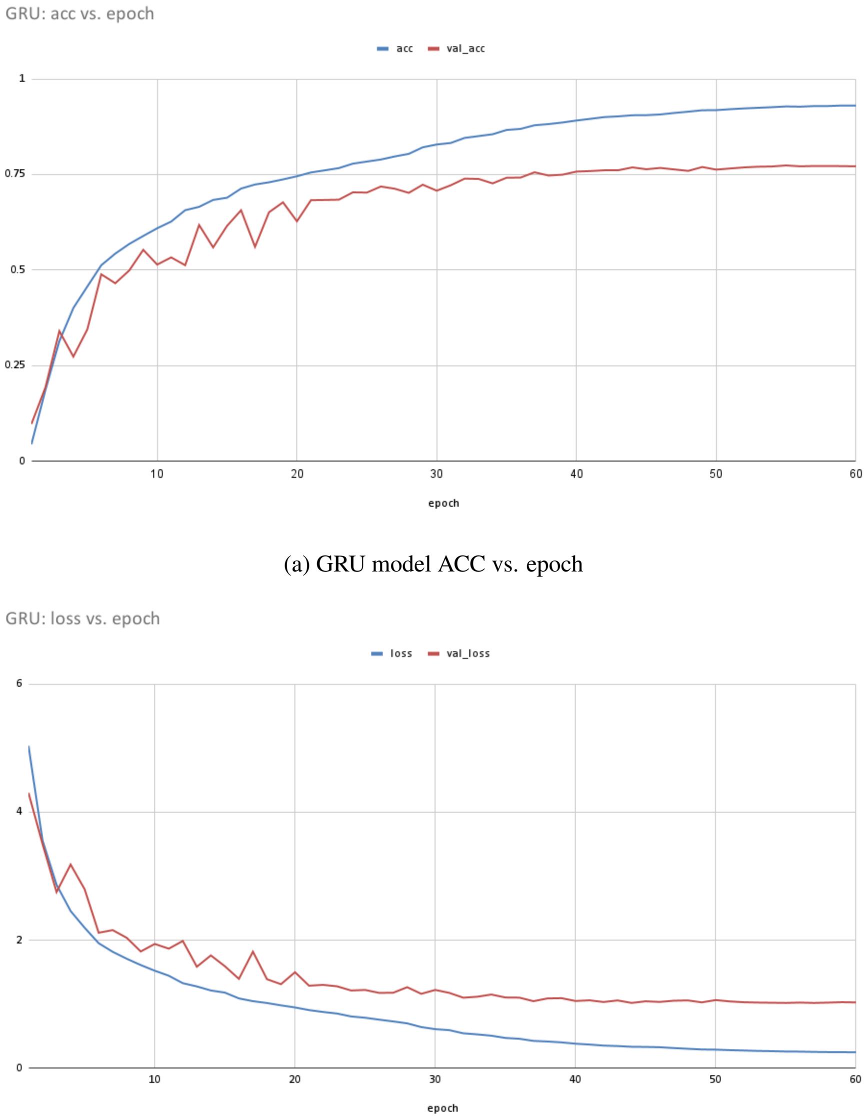
It is important to note that GRU performs significantly better than LSTM does. We suspect that the poor performance, potentially an underfitting of LSTM is due to an insufficient amount of data to adequately train the LSTM model. As previously mentioned, we have an average of only 397 samples per sign, which may not be sufficient for an LSTM model. To test this hypothesis, we reduced the test dataset from 80% to 10%. The performance on the training dataset increased to 85%, while the test dataset's accuracy fell below 20%. Besides, LSTM is designed to remember longer sequences with an extra memory unit in its architecture than GRU. With a median of 22 and average of 38 frames per sample, the length of input sequence of landmarks is possibly not sufficient for LSTM to outperform GRU.


non_empty_frame_idxs | InputLayer
tf.math.not_equal | TFOpLambda
tf.cast | TFOpLambda
tf.expand_dims | TFOpLambda
tf.compat.v1.shape | TFOpLambda
tf.random.uniform | TFOpLambda
tf.math.not_equal_1 | TFOpLambda
tf.math.greater | TFOpLambda
tf.math.logical_and | TFOpLambda
tf.where | TFOpLambda
tf.math.reduce_sum | TFOpLambda
tf.math.equal | TFOpLambda
tf.where_1 | TFOpLambda
frames | InputLayer
tf.slice | TFOpLambda
tf.slice_1 | TFOpLambda
tf.slice_2 | TFOpLambda
tf.slice_3 | TFOpLambda
tf.math.subtract | TFOpLambda
tf.math.subtract_1 | TFOpLambda
tf.math.equal_3 | TFOpLambda
tf.math.subtract_2 | TFOpLambda
tf.math.equal_1 | TFOpLambda
tf.math.truediv | TFOpLambda
tf.math.equal_2 | TFOpLambda
tf.math.truediv_1 | TFOpLambda
tf.math.truediv_2 | TFOpLambda
tf.where_2 | TFOpLambda
tf.where_3 | TFOpLambda
tf.where_4 | TFOpLambda
tf.reshape | TFOpLambda
tf.reshape_1 | TFOpLambda
tf.reshape_2 | TFOpLambda
embedding | Embedding
transformer | Transformer
tf.math.reduce_sum_2 | TFOpLambda
tf.math.multiply | TFOpLambda
tf.math.reduce_sum_1 | TFOpLambda
tf.math.truediv_3 | TFOpLambda
dropout | Dropout
dense | Dense
non_empty_frame_idx
frame
frame masking
lips
pose
hand
Position
Embedding
Transformer
Dropout
Dense

(a) Transformer Model Structure Generated by TensorFlow

(b) Simplified Transformer Model Structure

Figure 8: Transformer Model

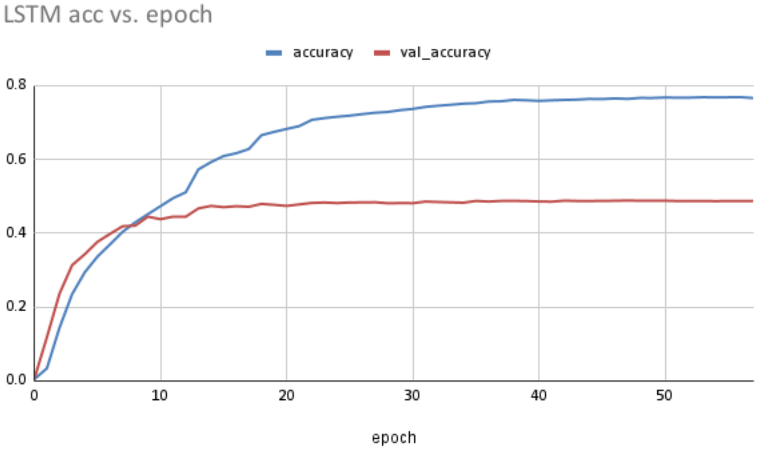


(a) GRU model ACC vs. epoch

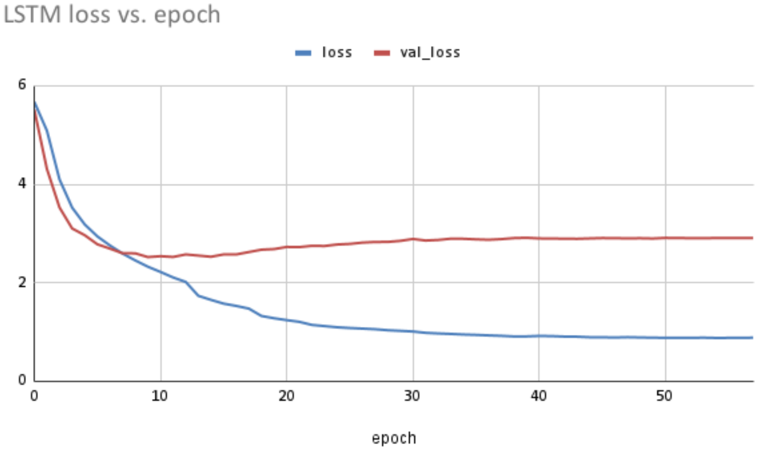
(b) GRU model loss vs. epoch

Figure 9: GRU Experiment Results

Additionally, training the LSTM was less efficient than the GRU. With 80% of the data serving as training data, the GRU's average training time was 7 seconds per epoch, whereas the LSTM model required 43 seconds per epoch on an NVIDIA Tesla P100 GPU provided by Kaggle.



(a) LSTM model ACC vs. epoch

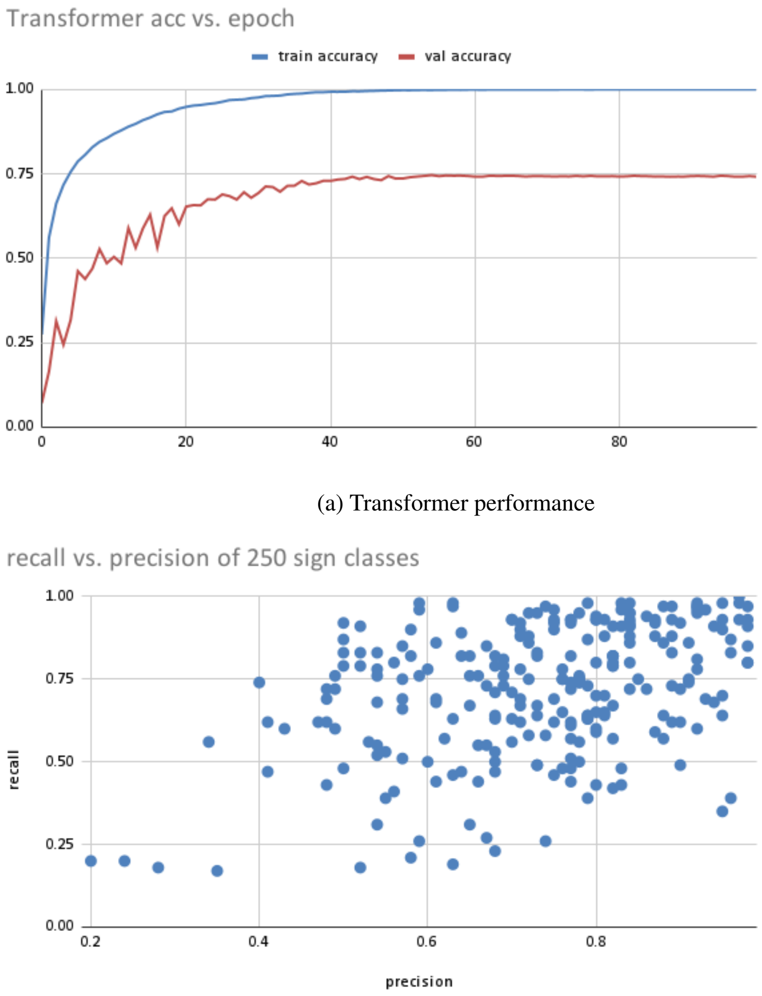


(b) LSTM model loss vs. epoch

Figure 10: LSTM Experiment Results

5.2 Transformer model

Our Transformer model demonstrated highly promising results within the scope of the current dataset. Remarkably, the model achieved a perfect 100% accuracy on 80% of the data designated for training with a solid performance on the validation data, with a 74% accuracy rate. With the recall vs. precision plot, it is noted that most classes of signs show similar performance with comparatively high recall and precision while few classes are low on both.



(a) Transformer performance

(b) Recall vs. precision of 250 sign classes

Figure 11: Transformer Experiment Results

To investigate the significance of positional embedding in a transformer at processing sequential input, the model was trained twice under the same settings with and without positional embedding. Their performance is shown in Figure 12. Interestingly, the inclusion of positional embeddings does not appear to have a substantial impact on the model's accuracy. Both versions of the model, with and without positional embeddings, can achieve a remarkable 100% accuracy on training data. However, the validation accuracy of the model without positional embeddings is marginally lower, with a decrease of approximately 2%. This suggests that while the positional embeddings may not dramatically influence the performance, they do contribute to a modest improvement in the model's ability to generalize to unseen data.

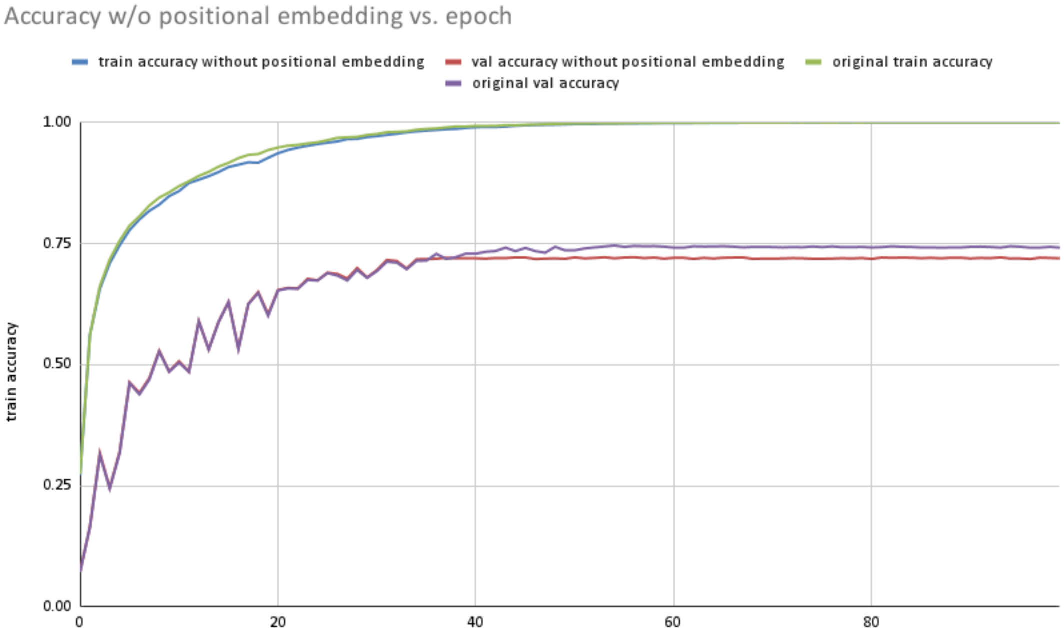
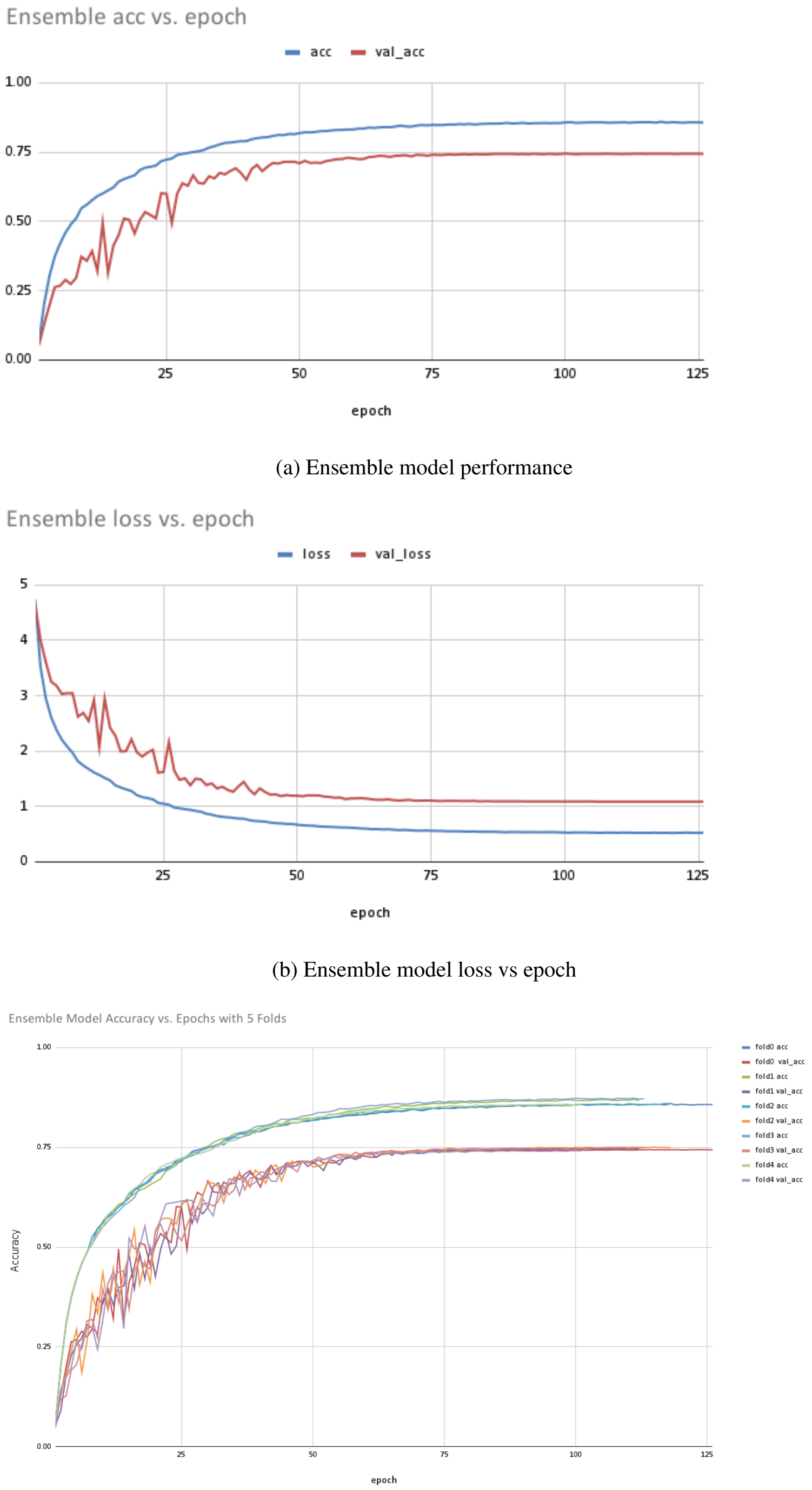


Figure 12: Transformer performance with/without positional embedding

5.3 Ensemble method

Our ensemble model's performance on the training set is 85% with a 75% accuracy rate over the test set as shown in Figure 13. The tiny performance difference between these two datasets shows that the ensemble model has the best generalizability. This gives us the direction for how to prevent overfitting in future studies. We also train and test the ensemble model under 5 folds setup, and its performance is consistent. It reached 85% on the training dataset and 75% on the test dataset on all five folds.



(a) Ensemble model performance

(b) Ensemble model loss vs epoch

(c) Ensemble model performance with 5 folds

Figure 13: Ensemble Experiment Results

6 Evaluation

In conclusion, GRU, transformer, and ensemble methods reached similar performances in validation with an accuracy of approximately 75%, while LSTM falls behind with lower accuracy on both training and testing data, as shown in Figure 14.

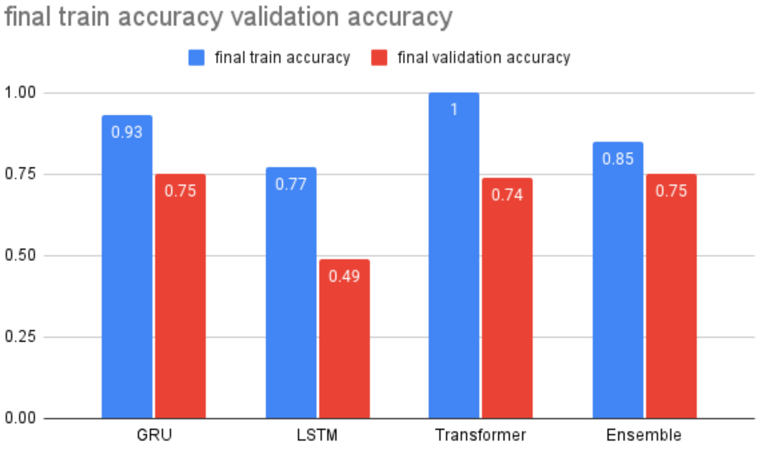


Figure 14: Comparison of accuracy between different methods

The key factor that contributes to the differences in the training and testing accuracy could possibly be generalization capability. It is evident that the Transformer model exhibits overfitting to the training data, as it demonstrates exceptionally high accuracy on the training set while its performance on unseen data is comparatively lower, which also applies to the GRU model. Key advantages of the ensemble method include improved generalization, reduce overfitting, and increased robustness, which explain the small discrepancy between its training and validation accuracy.

Without systematically conducting an experiment on a sufficiently large dataset, we applied these models to the landmark extracted from several real-world sign language instruction videos and the results are generally more inferior to the validation results. In addition to model overfitting, the number of frames in the samples typically being smaller than those found in conventional modern videos is also a major cause. This disparity may adversely impact the model's generalization capability, rendering it less effective in real-world scenarios.

7 Future Work

The architectural design of graph neural networks (GNN) suits the nature of SLR exceptionally[23]. In pose-based SLR, the key elements of signs can be effectively represented as a graph structure where nodes correspond to body parts and edges represent the spatial relationships between them. GNNs excel in capturing the local and global contextual information within such graph structures, enabling the recognition of intricate patterns that are essential for accurate interpretation. Furthermore, GNNs are inherently invariant to different input graph permutations and can generalize well across diverse sign languages and individual signing styles. This adaptability, combined with their capacity to model rich spatial dependencies, makes GNNs an ideal choice for pose-based sign language recognition systems. Therefore, an experiment with GNN-based models on SLR with the dataset extracted by MediaPipe holistic solutions is desired.

Besides, the size of the dataset poses a significant challenge in this study. Our current dataset is insufficient for effectively training the LSTM model, which also results in inadequate generalization and issues during the validation process as illustrated above. The lack of a comprehensive dataset not only hampers the model's performance but also limits its ability to adapt to variations in real-world signing scenarios. Therefore, a training set constitutes of landmark features extracted from a more extensive and diverse sign language video dataset would undoubtedly enhance the model's generalization capabilities and improve its overall performance in recognizing and interpreting signs.

References

[1] *American Sign Language: Which hand for signing?* Accessed: 2023-04-23. URL: https://www.lifeprint.com/asl101/pages-layout/rightorlefthand.htm#:~:text=If%5C%20you%5C%20are%5C%20right%5C%20handed,%5C%22one%5C%2Dhanded%5C%20signs.%5C%22.

[2] B. Bauer and H. Hienz. "Relevant features for video-based continuous sign language recognition". In: *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*. 2000, pp. 440–445. DOI: 10.1109/AFGR.2000.840672.

[3] Helen Cooper, Brian Holt, and Richard Bowden. "Sign Language Recognition". In: *Visual Analysis of Humans: Looking at People*. Ed. by Thomas B. Moeslund et al. London: Springer London, 2011, pp. 539–562. ISBN: 978-0-85729-997-0. DOI: 10.1007/978-0-85729-997-0_27. URL: https://doi.org/10.1007/978-0-85729-997-0_27.

[4] Djamila Dahmani and Slimane Larabi. "User-independent system for sign language finger spelling recognition". In: *Journal of Visual Communication and Image Representation* 25.5 (2014), pp. 1240–1250. ISSN: 1047-3203. DOI: https://doi.org/10.1016/j.jvcir.2013.12.019. URL: https://www.sciencedirect.com/science/article/pii/S1047320313002332.

[5] *GISLR – How to Ensemble*. Accessed: 2023-04-23. URL: https://www.kaggle.com/code/dschettler8845/gislr-how-to-ensemble.

[6] *GISLR TF Data Processing Transformer Training comments*. Accessed: 2023-04-23. URL: https://www.kaggle.com/code/markwijkhuizen/gislr-tf-data-processing-transformer-training/comments.

[7] *GISLR TF: On the Shoulders ENSAMBLE*. Accessed: 2023-04-23. URL: https://www.kaggle.com/code/aikhmelnytskyy/gislr-tf-on-the-shoulders-ensamble?scriptVersionId=121543912.

[8] *GISLR TF: On the Shoulders ENSAMBLE V2 0.69*. Accessed: 2023-04-23. URL: https://www.kaggle.com/code/aikhmelnytskyy/gislr-tf-on-the-shoulders-ensamble-v2-0-69.

[9] Google. *Google - Isolated Sign Language Recognition*. Accessed: 2023-03-31. 2023. URL: https://www.kaggle.com/competitions/asl-signs/overview.

[10] Vishwa Hariharan Iyer et al. "Sign Language Detection using Action Recognition". In: *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*. 2022, pp. 1682–1685. DOI: 10.1109/ICACITE53722.2022.9823484.

[11] Boeun Kim et al. "Global-local motion transformer fornbsp;unsupervised skeleton-based action learning". In: *Lecture Notes in Computer Science* (July 2022), pp. 209–225. DOI: 10.1007/978-3-031-19772-7_13.

[12] Lim et al. "Isolated sign language recognition using Convolutional Neural Network hand modelling and Hand Energy Image". In: *Multimed Tools Appl* (2019). DOI: https://doi.org/10.1007/s11042-019-7263-7.

[13] Dr. M. Madhiarasan and Prof. Partha Pratim Roy. *A Comprehensive Review of Sign Language Recognition: Different Types, Modalities, and Datasets*. 2022. arXiv: 2204.03328 [cs.CV].

[14] *MediaPipe Holistic*. Accessed: 2023-03-31. URL: https://ai.googleblog.com/2020/12/mediapipe-holistic-simultaneous-face.html.

[15] *MediaPipe Holistic Github*. Accessed: 2023-04-23. URL: https://github.com/google/mediapipe/blob/master/docs/solutions/holistic.md.

[16] Tanmoy Panigrahi. *Pop Sign Learning: A bubble-shooter game that helps its players learn American Sign Language while playing.* Accessed: 2023-03-31. 2022. URL: https://devpost.com/software/pop-sign-learning.

[17] Razieh Rastgoo, Kourosh Kiani, and Sergio Escalera. "Sign Language Recognition: A Deep Survey". In: *Expert Systems with Applications* 164 (2021), p. 113794. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2020.113794. URL: https://www.sciencedirect.com/science/article/pii/S095741742030614X.

[18] Robin M. Schmidt. *Recurrent Neural Networks (RNNs): A gentle Introduction and Overview*. 2019. arXiv: 1912.05911 [cs.LG].

[19] Ozge Mercanoglu Sincan and Hacer Yalim Keles. "AUTSL: A Large Scale Multi-Modal Turkish Sign Language Dataset and Baseline Methods". In: *IEEE Access* 8 (2020), pp. 181340–181355. DOI: 10.1109/access.2020.3028072. URL: https://doi.org/10.1109%5C%2Faccess.2020.3028072.

[20] Hand Speak. *Signs for ABANDON*. the word Abandon in American Sign Language. 2023. URL: https://www.handspeak.com/word/3/.

[21] *TensorFlow-Lite*. Accessed: 2023-03-31. URL: https://www.tensorflow.org/lite.

[22] *TFLite Ensemble*. Accessed: 2023-04-23. URL: https://www.kaggle.com/code/aleksandrkruchinin/tflite-ensemble.

[23] Anirudh Tunga, Sai Vidyaranya Nuthalapati, and Juan Wachs. *Pose-based Sign Language Recognition using GCN and BERT*. 2020. arXiv: 2012.00781 [cs.CV].

[24] Ashish Vaswani et al. *Attention Is All You Need*. 2017. arXiv: 1706.03762 [cs.CL].

[25] Guan Yuan et al. "Hand Gesture Recognition Using Deep Feature Fusion Network Based on Wearable Sensors". In: *IEEE Sensors Journal* 21.1 (2021), pp. 539–547. DOI: 10.1109/JSEN.2020.3014276.