Landmark-based American Sign Language Recognition using GNNs

Brian Sam-Bodden
Harvard University
brs017@g.harvard.edu

Julia Beltrame
Harvard University
jub816@g.harvard.edu

Taha Teke
Harvard University
tahateke@g.harvard.edu

Wilson Guo
MIT
wilsonguo@alum.mit.edu

Abstract

The ability to accurately perform "Real-time Isolated ASL Gesture Sequence Recognition" will bridge the communication gap between the deaf and hard-of-hearing community and the hearing population. By developing an application that translates American Sign Language (ASL) into English in real-time, we aim to provide a platform that promotes inclusivity and paves the way for more accessible communication technologies. Our approach integrates machine learning techniques, such as random forests for detecting and classifying ASL alphabet hand configurations and graph neural networks (GNNs), to accurately interpret and translate ASL gestures.

The project aims to capture complex combined sequences of hand, face, and pose gestures and translate them into their corresponding English words or phrases. Automating ASL to written language is a step in providing a complete round-trip conversion from spoken, written, and sign languages, which remains a complex, multidimensional problem yet to be fully solved.

1 Introduction

Approximately 0.1% of children in the United States are born with significant hearing loss (Data and Statistics About Hearing Loss in Children, CDC(Centers for Disease Control and Prevention)), often to parents unfamiliar with American Sign Language (ASL). This lack of early exposure to sign language can lead to Language Deprivation Syndrome, hindering essential language acquisition and affecting various aspects of life. For English speakers, mastering ASL can be as challenging as becoming proficient in a complex programming language.

Untouchable, invisible, tasteless, and yet sound has molded itself into every part of our lives. From jamming out to today's hit on Spotify to listening to friends speak, communicating and listening is imperative. This project seeks to bridge a communication gap between non-hearing/non-speaking individuals and hearing/speaking individuals, lowering communication barriers. Our project employs real-time recognition using a consumer-grade web camera (WebCam) and machine learning models to detect non-verbal ASL signs in real-time.

While the computer vision and NLP communities have made great strides over the years, there has not been much attention in the ASL recognition field. The current State Of Art can recognize and accurately identify forty signs with up to ninety percent accuracy. We learned and employed multiple techniques and models, such as Random Forest Models(Abdallah et al., 2023) and Graph Neural Networks, as well as classic computer vision techniques. While we did create our dataset for the static ASL alphabet recognition, we used Google's Kaggle Isolated Sign Language Recognition Dataset to train our system to recognize ASL gestures.

2 Background

Unlike spoken languages primarily using auditory and textual elements, sign languages like American Sign Language (ASL) are visual and spatial. ASL uses hand configurations, movements, facial expressions, and body positions to convey meaning, making translation and comprehension a complex learning task for those not proficient in ASL.

Traditionally, bridging the communication gap between ASL users and non-signers has relied on proficient human translators. However, the availability of interpreters is limited, and the need for real-time translation underscores the importance of developing accessible technological solutions.

Machine learning (ML) methods hold considerable promise in sign language translation. With advancements in ML algorithms and techniques, the potential for developing real-time, on-device translation tools is becoming more feasible.

Sign language translation is a complex classification problem involving sequential tasks that must consider a mix of hand gestures, facial expressions, and body movements. This complexity goes beyond simple image classification, demanding a more nuanced approach to capture the rich semantics of ASL accurately.

The typical architecture for addressing this challenge, highlighted in "Video-based isolated hand sign language recognition using a deep cascaded model" (Multimedia Tools and Applications, 2020), emphasizes multi-frame, sequence-to-sequence classification. This approach underlines the importance of processing and interpreting sequential data in machine learning models. Some challenges posed by classifying gestures from the raw video include that each video might have different backgrounds with varied objects, the humans performing the signs might be wearing different clothing and jewelry, and have different hairstyles. This highlights the high dimensionality and noise of raw, unstructured video data. One way to deal with this is to extract a condensed set of information focusing on each video frame's hands, face, and pose components. In this context, a critical intermediate component in ASL gesture recognition is the concept of 'landmarks.' These landmarks, defined as points in 3D space with x, y, and z coordinates, represent critical positions of hands, facial features, and body posture. As spatial anatomical markers, they encode the semantics of the essential elements of gestures. At the single frame level, they provide the necessary spatial configuration of the signer; across multiple frames, they provide essential temporal information.

One of the challenges and opportunities in using landmarks for ASL translation is feature engineering, in other words, deriving meaningful features from the relationship between these landmarks. This involves understanding the 'graph nature' of these landmarks, where each point is interconnected, forming a semantic network representing the language.

The unique structure of landmark data in ASL allows for exploring various machine learning architectures, notably Graph Neural Networks (GNNs). GNNs are particularly suited for handling the graph-like nature of landmark data, offering a promising approach to capturing the complex, dynamic relationships inherent in ASL gestures(Cao et al., 2020; Scarselli et al., 2023). In addition to GNNs, Random Forest algorithms are promising, analyzing a vast array of features from ASL data. Random Forest classifiers can discern small differences between signs, contributing to the overall precision and reliability of the translation tool.

3 Methodology

A dataset from the "Google - Isolated Sign Language Recognition" Kaggle Competition was used to train gesture recognition models. It comprised 250 signs from 21 participants and was used to train the GNN machine learning models. Additionally, we created our datasets for ASL letters to test the efficiency of Random Forest models. We wanted to ensure that our application could take in new data, extract features, and output correct inferences from real-time examples.

3.1 Random Forest Fingerspelling Recognition

In creating our dataset, we used open-cvs libraries to access the WebCam and take a hundred photos of each sign. When taking photos, we moved our arms forward and back, so the model would have training examples up close to the camera and back. Once we had the raw photos, we used Google's media pipe library to "feature" our data. We only featurized the hands since ASL letters are signed using only one hand. We were specifically interested in where the fingers were located relative to one another.

Once we had created a featured dataset, we split the dataset into a testing set and a training set and tested multiple classification models. We first looked at multiclass logistic classifiers and swept through multiple L2 regularization strengths. Regularizing logistic regression models helps ensure that our model is not overfitting to training data. Experimenting with different strengths, we chose the best model on the testing set.

Later on, we also tested random forest models, which typically perform better when the problem involves grouping closely resembled data points. We performed a similar approach to the one we used for logistic regression classification. For random forest models, we have two tuning parameters - the number of classification trees and the depth of the classification trees.

We used the same dataset as we did to train our logistic regression classifier. We swept between one and fifty random tree classifiers to obtain the best random forest model. During each random forest, we swept through a depth of between one and ten and kept the depth with the best accuracy. Once we had the best depths of each random forest model, we compared the accuracies of each random forest model on the testing set and selected the best one among them.

3.2 Graph Neural Networks for ASL Gesture Recognition

The final architecture chosen for the GNN classifier is a "Spectral Graph Neural Network," specifically a "Graph Convolution Neural Network" (GCN). Such an architecture is well-suited for tasks involving the processing of graph structures where spectral properties and topological structure are essential. A GCN is well-suited for capturing the complex spatial relationships inherent in sign language gestures for ASL sign classification from video sequences.

4 Data Processing

The training data set from "Google - Isolated Sign Language Recognition" Kaggle competition consists of landmarks extracted from raw videos with the MediaPipe(Samaan et al., 2022) holistic model representing 250 unique ASL signs. The dataset comprises individual Parquet files containing a collection of landmarks for each video frame depicting an ASL sign. The landmarks include 543 landmarks per frame for the face, both hands and pose.

Since the MediaPipe model is not fully trained to predict depth, we drop the z values, keeping only the x and y coordinates for each landmark. To reduce the task's computational complexity, we keep only selected landmarks from each of the face, hands, and pose sets. For the face, we keep a minimal set of face landmarks for the outline of the face, eyes, and lips. For the face oval, we keep the transition points from the sides of the face to the chin, the corner points where the face starts curving toward the chin, and some points for the chin. For eyes and lips, we keep the corners and skip every other point.

We keep all landmarks for the hands since we consider the hands to carry the most semantic load in ASL. For the pose, we only keep landmarks from the waist up (excluding wrists, hands, and face, including the nose tip). This results in a final set of 118 landmarks.

Each training video is interpolated or downsampled to a total of 40 frames. We arrived at this number based on the average frame count across all signs, which is 37.94.

We calculated temporal features for each landmark, including velocity and acceleration deltas between the same landmarks across adjacent frames.

5 Graph Building

GNNs operate on graphs, and the graph's topology is crucial to capturing the semantics of a scene. In addition, our training graphs must capture the complete collection of frames representing the video. In the case of anatomical landmarks, it follows that a sensible approach to the shape of our graphs is to follow the natural anatomical connections. Therefore, we connected our subset of landmarks following the predefined connections for the hands, face, and pose provided by the MediaPipe visualization code, taking into account our reduced landmark set. We also connected each frame comprising the video (40 frames) landmark-to-landmark.

5.1 Input Graph Type Description

1. Multi-Modal Graph: The constructed graph is multi-modal, encompassing different types of nodes representing landmarks from hands, face, and pose. Each landmark serves as a node in the graph, as shown in Figure 1.

2. Features of Nodes: Nodes in the graph are characterized by spatial (x, y coordinates) and temporal features (velocity and acceleration). This combination of spatial and temporal attributes enriches the node features, making them representative of both the static position and dynamic movement.

3. Edge Connections:

    - Spatial Edges: Within each frame, edges are created based on predefined anatomical connections between landmarks. These connections reflect the natural structure of hands, face, and pose, maintaining the integrity of the human body's topology.
    - Temporal Edges: Additionally, edges are established between corresponding landmarks in consecutive frames. These temporal edges link the same landmark across different time steps, capturing the movement and transition of each landmark.(Kumar et al., 2018)

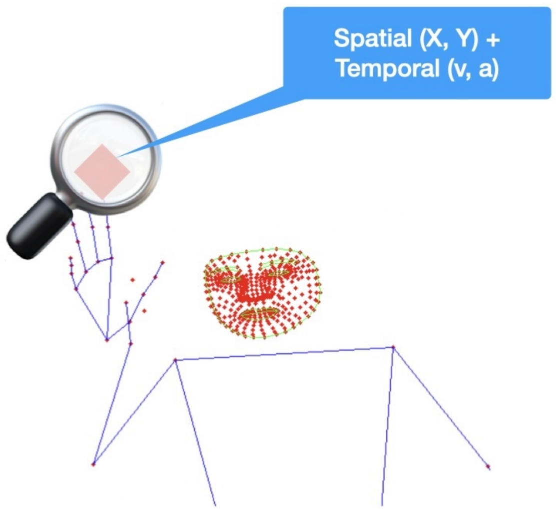

Figure 1: Single Frame Graph

The graph is a spatiotemporal graph, where nodes represent multi-modal landmarks with combined spatial and temporal features, and edges encode anatomical and dynamic movement connections, as shown in Figure 2.

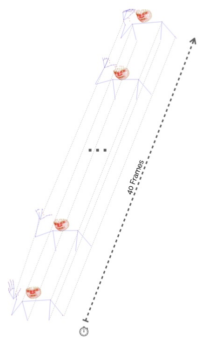

Figure 2: Multi-Frame Input Graph

This type of graph is ideal for capturing the complex nuances of sign language gestures in space and time.

6 GNN Architecture

The model comprises multiple Graph Convolutional Network (GCN) layers at its core. These layers are adept at aggregating and learning from the features of each node and its neighbors within the graph, capturing the intricate spatial and temporal relationships present in sign language gestures. Following the GCN layers, batch normalization stabilizes and accelerates the learning process by normalizing layer inputs. This is complemented by dropout layers, which prevent overfitting by randomly omitting a subset of features during training. Additionally, layer normalization is used after the final GCN layer to ensure the network's activations are standardized, leading to more stable training. Linear layers are included towards the end of the architecture to transform the learned features into a format suitable for classification. The GCN architecture is depicted in Figure 3.

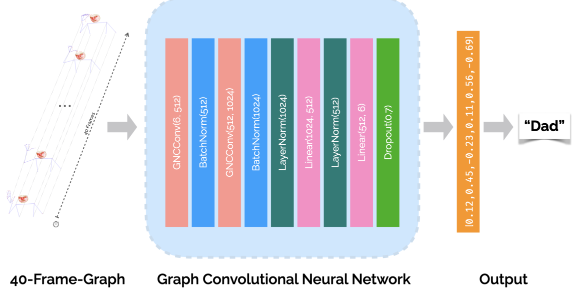

Figure 3: GNN Architecture

7 Training Process

The data is loaded and processed into graphs, capturing key features and connections. This graph data is then divided into training and testing subsets for model validation.

During training, the multi-layer GNN model learns from the graph representations. The training cycle involves processing batches of graph data, where the model predicts sign language gestures based on node features and graph topology. Optimization is guided by a loss function, integrating regularization to mitigate overfitting.

Adaptive learning rate strategies and early stopping enhance training efficiency and model generalization. After training, the model is evaluated on the test dataset to assess its predictive performance.

8 Experiments

We tested five different-sized models for the Graph Neural Networks: XS, S, M, L, and XL. We designed each model with the same architecture but varying neuron counts at each layer. The model sizes increase in XS, S, M, L, and XL. Our idea to create and test models of various sizes was inspired by Tan and Le's work on EfficientNetV2(Tan and Le, 2021). In our case, we wanted to analyze how increasing model size impacted classification performance.

9 Results

9.1 Random Forests

As described in our methodology section, we tested multiple logistic regression classifiers at different L2 regularization strengths:

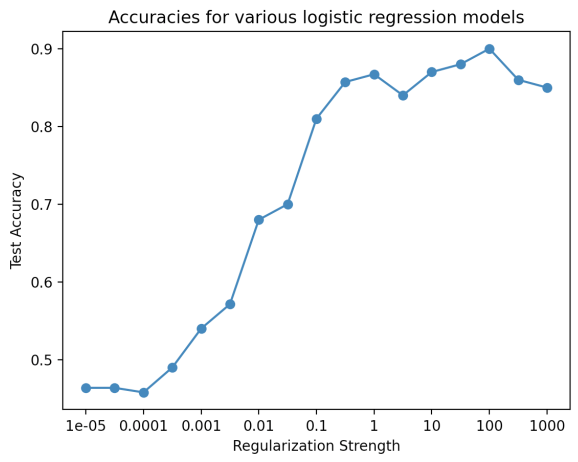

Figure 4: Logistic Regression Classifier Accuracy at various regularization levels

The best model had a test accuracy of ninety percent at a regularization strength of a hundred. The models at a lower regularization strength most likely overfit the training data. When the regularization strength is lower, the model is more expressive. The higher regularization strengths underfit as the regularizer coefficient prevented the model from becoming more expressive.

Similarly, following the approach described in the methodology section:

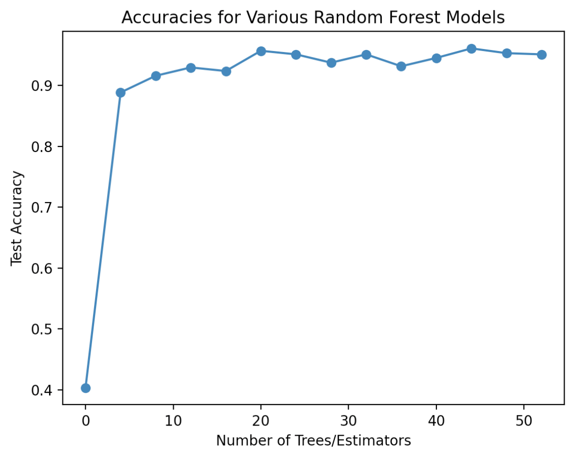

Figure 5: Random Forest Classifier Accuracy at various regularization levels

While the model with the highest model appeared between forty and fifty random tree classifiers, we chose to employ the model with twenty trees. The test accuracy for twenty trees is comparable, if not higher than the accuracies from other trees. With forty or fifty trees, our model could be more expressive but might also overfit the training data. A random forest model with twenty tree classifiers has the advantage of being able to generalize new data points better.

With the two best, more foundational models - logistic regression classifier and random forest classifier - we chose the random forest classifier. The random forest classifier has a higher test accuracy and a theoretical framework that aligns with grouping similar data points.

When testing our classification model in real-time, we found that while many of the signs were predicted correctly, the model attempted to classify hand signs captured when we transitioned between signs. Additionally, there were many signs that the model did not correctly classify. This is because of the angle and distance changes between our hands and cameras.

The Random Forest model was tested using a Python desktop application shown in Figure 6:

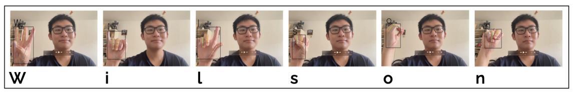

Figure 6: Fingerspelling Demonstration Application

9.2 GNNs

To alleviate some of these problems, we began exploring Graphical Neural Networks, which capture spatial and temporal relations. For the Graph Neural Networks, we tested the five different-sized models we designed: XS, S, M, L, and XL. We analyzed their final validation accuracies at the end of the training process. The results were rather interesting:

Model	Validation Accuracy
XS	79.3%
S	81.6%
M	80.7%
L	82.0%
XL	80.9%

Table 1: Validation Accuracy for Different Model Sizes

We observed that the final validation accuracies are close to one another with only single-digit percentage points of difference. The highest validation accuracy was seen with the L model, 82.082.0

To get a fuller understanding, we graphed the validation accuracy of each model in the training process over the epochs:

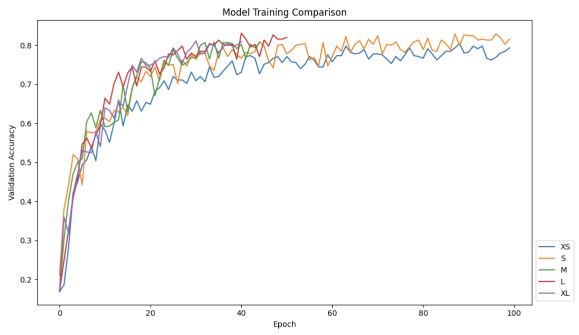

Figure 7: Validation Accuracy for different capacity GNN models

As mentioned, we can now visually validate that the validation accuracies have similar peaks for each model. However, because we implemented early stopping during training, we noticed that the training times for different models vary. The XS and S models required the entire 100 epochs to finish training, while the M, L, and XL models triggered early stopping around the 40-50 epoch marks.

We see that the number of epochs to train the more prominent models is significantly lower than that of the smaller models with a 50/100 = ½ overall reduction. Since the training times differ and the final validation accuracies are similar, we can take away that the larger models may be faster learners than the smaller models. However, both have the overall same performance potential. For the same reasons, the smaller models also show great promise for compute-constrained environments, like on-device usage.

The GNNs model was tested for Real-time recognition using a Python desktop application shown in Figure 8:

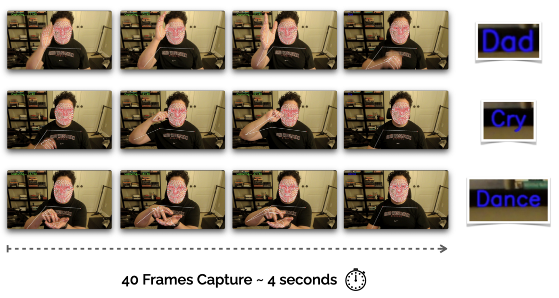

Figure 8: Isolated ASL Real-time Recognition Demonstration Program

10 Next Steps

Future research will focus on implementing the ASL recognition model by experimenting with various combinations of hand, pose, and face landmarks. The goal is to identify which combinations carry the most semantic significance in signing, optimizing our model's ability to interpret these more accurately. Besides this, we see a significant opportunity in targeted feature engineering. Despite the trend of relying on network-driven feature extraction, manually engineering features, especially given the bounded nature of hand configurations in ASL, could lead to a reduction in problem dimensionality. This step could streamline the model's processing capabilities, potentially increasing efficiency and accuracy.

Additionally, it is on the horizon to focus more research on temporal feature analysis. Building on our initial work with essential velocity and acceleration changes, we plan to explore more methods, such as optical flow, to capture the nuanced motion dynamics in ASL.

Moreover, we will expand into various GNN methodologies, including 3D Convolutional Neural Networks, Time-Distributed CNNs, and Temporal Convolutional Networks. These could increase our model's proficiency in interpreting the complex spatial-temporal patterns of ASL.

Finally, we have also researched other approaches to go about capturing information. For example, some researchers have abandoned the use of cameras. Instead, they focus on capturing moments using a "smart glove." A glove can capture information about the spatial relations between each finger much better than a camera. Additionally, a glove could control for angle much better than a two-dimensional image. With more time and resources, we would also explore working with these types of signals and implement machine learning algorithms once we have that dataset.

11 Conclusions

We tested many GNN models during our project to determine the most effective approach for translating ASL. During our selection process, we considered the ability of these models to process temporal features, which are essential for understanding the dynamic nature of ASL. Recognizing the significance of movement and transition in sign language, we focused on GNN architectures that could accurately interpret these temporal aspects despite the challenges posed by limitations in current frameworks.

We eventually decided to use PyTorch Geometric, a decision influenced by the technical challenges due to the lack of maintenance we faced with PyTorch Geometric Temporal. At the same time, we focused efforts on feature engineering. Given the complexity of ASL, selective engineering features have the potential to streamline the model's processing significantly. However, the depth of our experimentation in this area was limited due to time constraints. We also put effort into implementing various data augmentation techniques, which enhanced model training and robustness.

Although challenging, this project was gratifying. It is essential to improve communication for the deaf and hard-of-hearing community, highlighting the importance of ongoing research and development in this area.

References

Mohamed S. Abdallah et al. 2023. Light-weight deep learning techniques with advanced processing for real-time hand gesture recognition. *Sensors*, 23(1):2.

Z. Cao, G. Hidalgo, T. Simon, S. E. Wei, and Y. Sheikh. 2020. Spatial temporal graph convolutional networks for skeleton-based action recognition. Details to be added.

Centers for Disease Control and Prevention. Data and statistics about hearing loss in children. https://www.cdc.gov/ncbddd/hearingloss/data.html. Accessed: n.d.

S. S. Kumar, T. Wangyal, V. Saboo, and R. Srinath. 2018. Time series neural networks for real time sign language translation: Introduction of neural networks for asl gloss recognition and english sentence translation. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*.

Gerges H. Samaan et al. 2022. Mediapipe's landmarks with rnn for dynamic sign language recognition. *Electronics*, 11(19):3228.

F. Scarselli, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. 2023. Graph neural networks for temporal graphs: Comprehensive overview of gnns for temporal data analysis. Details to be added.

Mingxing Tan and Quoc V. Le. 2021. Efficientnetv2: Smaller models and faster training. https://doi.org/10.48550/arXiv.2104.00298. ArXiv:2104.00298.