SignLens: A Sign Language Translation Model for Specially Abled

Gagan Singhal, Ananya Singh, Andrew Wood
Boston University, Boston, MA
gsinghal@bu.edu xananya@bu.edu aewood@bu.edu

Abstract

This project addresses the significant communication barriers faced by the deaf and mute community in our increasingly digital world. We propose an innovative approach that bridges this gap by translating sign language into text in real time. Our solution aims to facilitate seamless communication across a wide array of contexts, from casual social interactions to professional environments and educational settings. By leveraging artificial intelligence through computer vision and NLP, we strive to create a more inclusive digital landscape where sign language users can effortlessly communicate with those who may not understand sign language. Our proposal addresses this by developing a robust model that can translate American Sign Language (ASL) using a single camera by segmenting the signs using Computer Vision and then translating them. We want to use the translation to go through our NLP model to make the machine use the relationship between words to form meaningful sentences with high accuracy. The goal is to create a model that can do this in real time.

1 Introduction

Our project addresses the critical gap between deaf or mute people who know sign language and people who don't know sign language by developing an innovative solution that translates sign language into text in real time, enabling seamless communication between those people.

Sign language is an intricate system of visual-gestural communication that presents unique challenges in digital interpretation. Sign language is the primary mode of communication between the deaf and mute communities. The language uses the movement of hands, eyes, neck, lips, and body. Thus, the signs can be very different from each other and also very similar. They can be identical in weaving signs with the hands, while eye and neck positioning could be non-identical. Training a model on such features could be difficult and, thus, might need a multi-modal approach. Our research plans to leverage the WLASL dataset to capture and accurately translate the multifaceted elements of sign language, including hand movements, facial expressions, and body posture. This approach opens up new possibilities for inclusive communication across various settings.

Our model will aim to be efficient and accessible, ensuring easy integration into existing devices and applications without straining system resources. This focus on universal design will aim to make sign language interpretation widely available, potentially transforming educational, professional, and social interactions for the deaf and mute communities.

Our project will deliver a system involving a camera and a multi-modal architecture for processing. An example of the use case of our project would look like the following. Let's say a human would stand before the camera and start using sign language for communication. The video stream from the camera will then go into our model as input, and the model will output the sentences on the screen being spoken in sign language by the human in front of the camera. The architecture would at least have two models at work. The first would be to process the video frames, extract feature language in the frames, and translate those signs into words. The second model would use this continuous stream of words to generate meaningful sentences to improve accuracy and not mess up the sentences the human communicates in front of the camera.

2 Related Work and Challenges

Sign language translation has been an area of research for several years, and significant strides have been made in using computer vision, deep learning, and natural language processing (NLP) to bridge

the communication gap between deaf and mute individuals and others. Below are some key approaches and contributions in this field:

2.1 Computer Vision for Sign Language Recognition

The primary challenge in sign language recognition lies in interpreting the visual-gestural communication, which includes hand movements, facial expressions, body postures, and finger spelling. Several studies have utilized computer vision and deep learning techniques to automate the recognition of sign language. Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have been widely used to extract features from video frames and predict sign language gestures.

For example, Zhou et al. [12] proposed a model that used CNNs to recognize static hand gestures in American Sign Language (ASL). The model was trained on a large-scale dataset of hand gesture images and achieved promising results. Similarly, Mollahosseini et al. [9] applied CNNs for both image-based and video-based recognition of ASL signs and showed improvements in recognition accuracy through temporal information processing.

Additionally, some researchers have focused on incorporating multi-modal inputs, such as the combination of depth information from sensors like Microsoft Kinect, which allows for more robust detection of hand gestures and body postures. Du et al. [3] used a combination of depth images and RGB data to improve gesture classification, enabling better handling of occlusions and complex hand positions.

2.2 Datasets for Sign Language Recognition

A key challenge in training sign language recognition systems is the lack of large, labeled datasets. Recent efforts have focused on curating datasets that include both hand movements and the accompanying contextual features such as facial expressions and body postures.

The WLASL (Word-Level American Sign Language) dataset, for instance, contains over 2,000 ASL signs across more than 200 categories, making it a valuable resource for training machine learning models on hand gesture recognition [8]. Other datasets like RWTH-PHOENIX-Weather [6] and the Sign Language MNIST dataset [4] also provide valuable resources for developing machine learning systems for sign language recognition.

2.3 Natural Language Processing for Sentence Generation

While significant progress has been made in recognizing individual signs, translating these signs into coherent sentences presents a more complex challenge. Recent work has explored combining sign language recognition with NLP models to generate meaningful text-based outputs. One approach involves training sequence-to-sequence models, such as the Transformer architecture, to map sequences of hand gestures to sequences of words or phrases.

For instance, Zhou et al. [11] proposed a method that integrates hand gesture recognition with a deep learning-based sentence generator. The system uses gesture-to-word mappings, followed by an NLP model to ensure the output sentences are syntactically and semantically accurate. Other research has incorporated pre-trained language models like BERT or GPT-2, to improve the fluency of generated sentences and handle context more effectively.

2.4 Challenges and Future Directions

While existing approaches have shown promise, several challenges remain. One major challenge is ensuring real-time processing of sign language input, as the speed and complexity of hand gestures can vary greatly. Additionally, current models often struggle with handling diverse signing styles, regional dialects, and signs that may appear visually similar but carry different meanings.

Furthermore, the integration of facial expressions and body posture remains underexplored, despite their crucial role in conveying meaning in sign language. Multi-modal approaches that combine gesture, facial expression, and body posture recognition could help address these issues and improve overall translation accuracy.

Finally, many systems still lack the ability to translate sign language in real-time, which is essential for practical applications such as live conversations or lectures. Future research could focus on optimizing the efficiency of models, making them lightweight and suitable for deployment on portable devices, while also ensuring that they handle a wide variety of signs with high accuracy.

In summary, significant progress has been made in the field of sign language translation, but challenges remain in areas such as real-time processing, multi-modal recognition, and fluent sentence generation. Our project aims to build upon these

advances, leveraging computer vision and NLP to create a system capable of real-time, accurate sign language translation in a wide range of contexts.

3 The WLASL Dataset

Our project plans to use the WLASL - World Level American Sign Language dataset [7], containing over 2000 words performed by over 100 signers. It's one of the most extensive datasets available for ASL. This dataset has about 14 hours of video spread among approximately 21000 videos. Each gloss in the database has an average of 10.5 samples. Each video is annotated with the corresponding ASL word, providing clear classification targets for the learning models. It features a diverse group of 100 signers in terms of age, gender, and signing style to enhance the robustness of recognition systems. The dataset was created by collecting videos from different trusted sign language websites where experts verified the mappings of glosses and signs before being uploaded. A temporal boundary feature has been used to indicate a sign's start and end frames in each video. YOLOv3[10] was used as a person detection tool to identify body-bounding boxes of signers in the videos to reduce side effects caused by backgrounds. The dataset includes a feature called bbox containing the bounding box coordinates. All features included with the dataset are:

Feature	Description	Data Type
gloss	The sign labels	str
bbox	bounding box (xmin, ymin, xmax, ymax)	[int]
fps	frame rate (=25)	int
frame_start	The starting frame of the gloss in the video indexed from 1.	int
frame_end	The ending frame of the gloss in the video	int
instance_id	id of the instance in the same class/gloss.	int
signer_id	id of the signer	int
source	a string identifier for the source site.	str
split	indicates which sample belongs to which subset.	str
url	used for video downloading.	str
variation_id	id for dialect (indexed from 0)	int
video_id	a unique video identifier.	str

Table 1: **Table of different features in the dataset (Adapted from [2])**

Dataset	Gloss	Videos	Signers
WLASL100	100	2038	97
WLASL300	300	5117	109
WLASL1000	1000	13168	116
WLASL2000	2000	21083	119

Table 2: **Table of different subsets of the dataset**

4 Baseline Approach

The baseline model aims to evaluate the effectiveness of two state-of-the-art models, Pose-TGCN (Pose-based Temporal Graph Convolutional Network) and I3D (Inflated 3D Convolutional Network), in the task of sign language recognition. These models are selected for their ability to capture motion dynamics and spatio-temporal features, respectively. By assessing their performance, we establish a foundation for understanding the strengths and weaknesses of each model in processing sign language data.

4.1 I3D (Inflated 3D Convolutional Network)

I3D is a deep learning architecture designed to process raw video data in an end-to-end fashion. It utilizes 3D convolutions, where both spatial and temporal dimensions are taken into account, making it highly suitable for tasks that involve dynamic sequences, such as action recognition in videos. The model inflates pre-trained 2D convolutions into 3D to handle temporal dynamics effectively. This approach allows I3D to learn the intricate relationships between spatial motion and temporal progression in the context of sign language.

However, while I3D excels in end-to-end video analysis, it primarily relies on frame-level features

extracted from the raw video without considering human pose or skeletal information explicitly. Furthermore, incorporating pose-based models like Pose-TGCN may address these limitations, providing a more detailed understanding of human motion and improving recognition performance.

4.2 Pose-TGCN (Pose-based Temporal Graph Convolutional Network)

Pose-TGCN focuses on modeling the temporal dynamics of sign language gestures by processing pose data rather than raw video frames. Pose data is often generated through pose estimation models that capture keypoints of the human body, typically in the form of joints and their spatial relationships. Pose-TGCN leverages this information to create temporal graph representations, where each pose is treated as a node in a graph, and the edges encode the relationships between body joints over time.

Pose-TGCN is effective in capturing the motion dynamics that are critical for sign language recognition by relying on pose data instead of just raw video frames.

	WLASL-100	WLASL-300	WLASL-1000	WLASL-2000
I3D	65.89	56.14	47.33	32.48
Pose-TGCN	55.43	38.32	34.86	23.65

Table 3: **Top - 1 accuracies for different sub-datasets in the two models**[7]

	WLASL-100	WLASL-300	WLASL-1000	WLASL-2000
I3D	84.11	79.94	76.44	57.31
Pose-TGCN	78.68	67.51	61.73	51.75

Table 4: **Top - 5 accuracies for different sub-datasets in the two models**[7]

	WLASL-100	WLASL-300	WLASL-1000	WLASL-2000
I3D	89.92	86.98	84.33	66.31
Pose-TGCN	87.60	79.64	71.91	62.24

Table 5: **Top - 10 accuracies for different sub-datasets in the two models**[7]

5 Methodology

5.1 Data Preprocessing

1. We resized the resolution of all original video frames to 224×224 because the dataset contained videos of different lengths and breaths.

2. The following random modifications were applied to each video frame:

   - Horizontal and vertical flipping.
   - Rotation.
   - Changed channel intensities.
   - Shearing.
   - Modification of brightness intensities.

3. Not all frames went through all modifications. We randomly chose 2 - 3 modifications for each frame and applied them to each.

4. Each video is divided into 12 chunks, and 1st and last chunks of frames are dropped as they are assumed to have no action involved.

5. Data labels or targets are in string format. They are converted to integer format for easy classification.

6. Data is then divided into train, validation and test set.

5.2 Model Experimentation

As we followed the traditional approach of training our data using different models for spatial and temporal data, we used the following Neural Networks:

5.2.1 Spatial Features

Spatial features are pivotal in interpreting the visual content of sign language, encompassing hand gestures and body movements. In our videos, these features include:

- **Hand shape and position**: The configuration and location of the hands in space.
- **Hand orientation**: The direction the hands face (e.g., upward, outward).
- **Facial expressions**: Convey additional meaning or grammatical context.
- **Environmental features**: Such as clothing color and the background color of individuals in the videos.

These elements collectively provide the essential visual cues required to understand and interpret the signs being performed.

To identify the most suitable model for our task, we trained and evaluated three convolutional neural network (CNN) architectures. Each model was initialized with pre-trained weights from the ImageNet dataset and fine-tuned on a sub-dataset containing 50 words. The evaluation revealed the following results:

- **VGG16**: Achieved the best performance, effectively extracting and training on spatial features.
- **DenseNet121**: Ranked second in performance.
- **MobileNetV2**: A close third but with notable advantages in terms of its small architecture and computational efficiency.

Despite VGG16's superior performance, we chose **MobileNetV2** as our primary model. This decision was influenced by its relatively strong performance combined with its lightweight design, making it well-suited for deployment on resource-constrained edge devices such as mobile platforms. MobileNet CNNs are specifically designed to balance performance and efficiency, aligning perfectly with the goals of our project.

5.2.2 Temporal Features

After extracting the spatial features from the sign language videos, we focus on modeling the temporal dynamics of the gestures, which is crucial for understanding the sequence of movements that convey meaning in sign language. In sign language, a single gesture can consist of a sequence of movements that need to be analyzed together over time to fully capture the intended meaning. This is where Recurrent Neural Networks (RNNs), particularly Gated Recurrent Units (GRUs) and Long Short-Term Memory (LSTM) networks, come into play.

Gated Recurrent Units (GRU) [13]

GRUs are a type of RNN that are specifically designed to model sequential data. They are well-suited for tasks involving time-series data or sequences, like sign language, where the order of gestures is essential to understanding the meaning. GRUs address the problem of vanishing gradients — a challenge faced by traditional RNNs by using gating mechanisms that allow the network to retain information over longer sequences.

In a GRU, there are two main gates: - Update gate: Controls how much of the previous hidden state needs to be carried forward. - Reset gate: Determines how much of the previous state should be forgotten.

These gates allow GRUs to selectively remember or forget information, making them particularly effective at learning temporal dependencies in data where long-term context is important. For instance, in sign language, the relationship between movements at different time points (such as hand position or gesture changes) plays a key role in interpreting the overall sign.

Long Short-Term Memory (LSTM) [5]

LSTM networks, another type of RNN, have a more complex architecture than GRUs. LSTMs include three gates: - Input gate: Controls how much of the new input should be stored in the cell state. - Forget gate: Determines which information from the previous cell state should be discarded. - Output gate: Decides how much of the cell state should be passed to the output at each time step.

LSTMs also introduce a memory cell, which acts as a container for long-term information. The memory cell enables the network to maintain a memory over extended sequences, allowing the model to capture long-term dependencies more effectively. This makes LSTMs particularly powerful for tasks that involve complex and long-range temporal relationships, such as analyzing lengthy sequences of gestures or signs in sign language.

In our experiments, both GRU and LSTM were evaluated for modeling the temporal features of the sign language gestures. The primary difference between the two lies in the complexity of their architecture and their computational requirements:

- LSTM networks, due to their additional gates and memory cell, are more powerful in capturing long-range dependencies. However, they come with increased computational cost, requiring more memory and longer training times. LSTMs are better suited for tasks where capturing long-term context and dependencies is critical, such as understanding complex sign language gestures involving multiple movements over time.
- GRU, on the other hand, is computationally more efficient. With fewer gates and no sepa-

rate memory cell, GRUs are faster to train and less resource-intensive, making them a practical choice when training time or available resources are limited. Despite their simpler architecture, GRUs have been shown to perform similarly to LSTMs on many tasks that require capturing temporal dependencies.

In summary, the use of GRUs for modeling the temporal dynamics of sign language gestures offers a balance between performance and computational efficiency. While LSTMs may provide superior accuracy due to their more complex architecture, GRUs offer a faster and more resource-efficient solution without sacrificing too much in terms of accuracy. The ability of GRUs to learn the sequential nature of gestures in sign language, combined with their lower computational cost, makes them a suitable choice for real-time sign language translation, especially when deploying on devices with limited resources.

5.3 Predicting Unseen Words Using the N-gram Model

In the sign language translation model, when the system encounters an unseen word during translation, instead of returning a placeholder (such as `UNK` or `None`), we implemented a mechanism using the *N-gram model* to predict the next word based on the previously translated words.

The approach leverages the *N-gram model* to predict the next word in a sequence by considering the last $n-1$ words.

1. **Extract Context**: Once the model has translated the first four words, the context (the last N words) is extracted for the prediction of the next word. For instance, if the model has translated "I am happy," the context for predicting the 5th word will be the last two words: `['am', 'happy']`.

2. **Probability Computation**: Using the N-gram model, we compute the probability of possible next words based on the last $n-1$ words (e.g., for a trigram, the previous two words).

3. **Selecting the Best Word**: The word with the highest probability is selected as the next word in the sequence.

4. **Back-off Strategy**: If no valid trigram exists for the given context (i.e., the N-gram model has not seen this particular sequence of words during training), the model falls back to lower-order N-grams.

5. **Fallback Strategy**: In cases where all N-gram models fail to produce a prediction (e.g., for completely unseen sequences), the model predicts the most frequent word in the dataset.

5.4 Proposed Model

The proposed model for sign language translation integrates spatial feature extraction, temporal modeling, and language generation. Below is a detailed explanation of each component of the model, including the architectural choices and the formulas used in various stages:

Spatial Feature Extraction:

- The spatial features of sign language gestures are extracted using the `MobileNetV2` architecture, pre-trained on the ImageNet dataset. MobileNetV2 is chosen for its lightweight architecture, making it suitable for edge devices with limited resources.

- A `TimeDistributed` wrapper is used to apply `MobileNetV2` on each frame of the video sequence. This wrapper allows the model to treat each frame independently while preserving the sequential nature of the input.

- After passing through MobileNetV2, the features are pooled using `GlobalAveragePooling2D` to reduce the dimensionality of the extracted features, ensuring that only the most important spatial information is retained.

- Formula for feature extraction from a single frame:

$$\mathrm{F}_{frame} = GlobalAveragePooling2D(MobileNetV2(\mathbf{I_{frame}}))$$

  where $\mathbf{I_{frame}}$ is the input image (frame), and $\mathbf{F_{frame}}$ is the output feature vector.

Temporal Sequence Modeling (GRU):

- To capture the temporal dynamics of the gesture sequences, we use a `GRU`-based Recurrent Neural Network (RNN). GRUs are particularly well-suited for learning long-term dependencies in sequential data, as they can effectively handle the vanishing gradient problem.

- The `GRU` layers are used in a two-layer configuration:

  - The first `GRU` layer has 256 units and returns sequences, allowing the second `GRU` layer to learn from the entire sequence.
  - The second `GRU` layer has 256 units and outputs the final sequence representation.

- Batch normalization is applied after the first `GRU` layer to stabilize and accelerate training by normalizing the activations within each mini-batch.

- Formula for GRU-based sequence modeling:

$$\mathbf{h_t} = GRU(\mathbf{h_{t-1}}, \mathbf{F_t})$$

  where $\mathbf{h_t}$ is the hidden state at time step t, and $\mathbf{F_t}$ is the spatial feature vector for the frame at time step t.

Feedforward Network for Classification:

- After processing the sequence with the GRU layers, the output is passed through a feedforward neural network consisting of two fully connected layers with 200 and 150 units, respectively. These layers use `ReLU` activation functions.

- Dropout (with a rate of 0.66) is applied to prevent overfitting by randomly disabling a fraction of the neurons during training.

- The final output layer consists of units equal to the number of possible sign language classes, with a `softmax` activation function to generate the probability distribution over the classes.

- Formula for classification:

$$\hat{y} = Softmax(\mathbf{W} \cdot \mathbf{h} + \mathbf{b})$$

  where $\mathbf{W}$ is the weight matrix, $\mathbf{h}$ is the final hidden state, and $\mathbf{b}$ is the bias vector.

N-gram Language Model[1]:

- To improve the fluency and coherence of the translated sentences, we use an N-gram language model to predict the next word in the sequence based on the previously predicted words.

- The N-gram model is implemented in conjunction with the sign language translation model. After the sign is translated into the corresponding word sequence, the N-gram model refines the translation by considering the context of the preceding words.

- The N-gram model predicts the next word as follows:

$$\text{P}(\text{w}_n|w_1, w_2, \ldots, w_{n-1}) = \frac{Count(w_1, w_2, \ldots, w_{n-1}, w_n)}{Count(w_1, w_2, \ldots, w_{n-1})}$$

- The most likely next word is selected by maximizing the conditional probability:

$$\hat{w}_n = \arg\max_{w_n} P(w_n|w_1, w_2, \ldots, w_{n-1})$$

Final Model Architecture:

- The final model architecture consists of the following stages:
  - Feature extraction: MobileNetV2 for spatial feature extraction.
  - Temporal modeling: GRU-based RNN to capture the temporal dependencies.
  - Classification: Fully connected layers to classify the gesture into one of the predefined sign language classes.
  - Language generation: N-gram model for predicting the next word and enhancing the fluency of the translation.

- This end-to-end model is trained on a labeled sign language dataset, with both spatial and temporal components learning to predict the correct translation for each gesture sequence.

6 Obeservations

- As the database contains videos and each video has high number of frames, the CNNs and RNNs take a huge time to train.

- To tackle the above problem, we only trained our models on a subset of 20 and 100 words from the 2000 word dataset of WLASL.

- Resizing the frames made a huge impact to model training resource and time usage.

- Capturing just spatial features using CNN and no RNN only gave an accuracy of about 2.7% on the 100 vocabulary sub-dataset.

- Running the model on the 20 words vocabulary and 100 words vocabulary gave the following results:

6.1 Performance Metrics

After training the model on the WLASL100 dataset for 300 epochs and 12 hours of training, we calculated the top k scores for k = [1, 5 10] and calculated the precision, recall, and F1 scores.

Fig. 1: **Performance results for training, validation & testing sets for WLASL20**

Metric	WLASL20	WLASL100
Accuracy	40.44%	22.21%
Precision	0.4397	0.2632
Recall	0.4375	0.2583
F1 Score	0.40	0.2421

Table 6: **Different Performance Metrics for Test data**

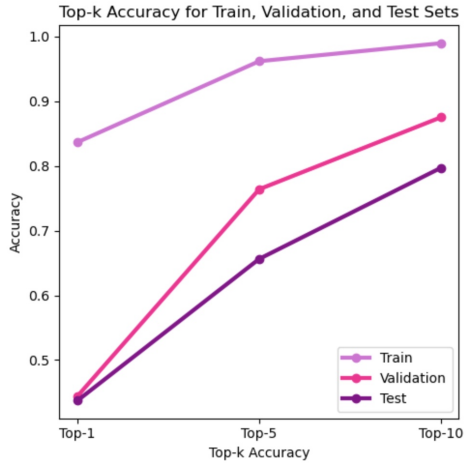

Fig. 2: **Top K accuracy for training, validation & testing sets for WLASL20**

7 Conclusion and Future Work

In this paper, we talked about a traditional approach to creating a model for real-time sign language translation, leveraging computer vision and natural language processing (NLP) techniques. Our model is designed to bridge the communication gap between individuals who use sign language and those who do not, facilitating more inclusive interactions across various contexts such as education, social interactions, and professional environments.

By combining spatial modeling, temporal modeling, and an N-gram-based sequence prediction, our approach provides a proposal of a framework for accurate sign language translation.

Our evaluation of the model on multiple datasets, including the WLASL100 and WLASL20 subsets, demonstrates weak results right now due to lack of resources and time constraints as each model took too long to train and experiment with. As the current model shows weak performance, we acknowledge the challenges of training complex Neural Networks for this use case and the need for further refinement in capturing the nuanced aspects of sign language translation.

Future work will focus on improving the model's accuracy, enhancing the system's real-time processing capabilities, and extending the model to support more sign languages. Ultimately, we believe our approach can contribute to breaking down communication barriers and fostering greater inclusion for the deaf and mute community.

This approach allows for smoother translation by leveraging the N-gram model to predict the next word, even in cases of unseen word sequences. By using back-off strategies, we ensure that the translation remains as accurate as possible, avoiding the use of placeholders like `UNK` and enhancing the overall fluency of the sign language translation system.

All in all, we'd like to point out that our initial objective, although ambitious as we wanted to generate conversations but could barely reach the point of generating meaningful sentences. The journey took us to an amazing path of learning and application. The depths we had to go while choosing models, tuning models, processing data, etc was a great experience which will motivate us further for other problems and solutions. We aim to continue on this project till we have something meaningful.

References

[1] Ciprian Chelba, Mohammad Norouzi, and Samy Bengio. 2017. N-gram language modeling using recurrent neural network estimation. *Preprint*, arXiv:1703.10724.

[2] Dongxu. 2020. Wlasl. https://github.com/

dxli94/WLASL/tree/master. Accessed: 2024-10-23.

[3] Y. Du et al. 2015. Gesture recognition using depth images from kinect sensor for sign language recognition. In *IEEE International Conference on Computer Vision*, pages 1133–1140. IEEE.

[4] A. Georgieva et al. 2020. Sign language mnist: A large dataset for gesture recognition in sign language. *arXiv preprint arXiv:2001.00130*.

[5] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9:1735–80.

[6] O. Koller et al. 2018. The rwth-phoenix-weather 2014t dataset for sign language recognition. In *International Conference on Pattern Recognition*, pages 1–8. IEEE.

[7] Dongxu Li, Cristian Rodriguez, Xin Yu, and Hongdong Li. 2020. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 1459–1469.

[8] Y. Liu et al. 2020. Wlasl: A large-scale dataset for word-level american sign language recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1786–1795.

[9] A. Mollahosseini et al. 2017. Hand gesture recognition for american sign language using deep learning. *Computer Vision and Image Understanding*, 162:78–88.

[10] Joseph Redmon and Ali Farhadi. 2018. Yolov3: An incremental improvement. *Preprint*, arXiv:1804.02767.

[11] H. Zhou et al. 2021. Gesture-to-word and sentence generation in sign language recognition with deep learning. *IEEE Transactions on Neural Networks and Learning Systems*, 32:270–280.

[12] X. Zhou et al. 2018. A deep learning-based approach for sign language recognition using cnns. *Journal of Artificial Intelligence Research*, 64:337–350.

[13] Fuyu Zhu, Hua Wang, and Yixuan Zhang. 2023. Gru deep residual network for time series classification. In *2023 IEEE 6th Information Technology,Networking,Electronic and Automation Control Conference (ITNEC)*, volume 6, pages 1289–1293.