Advance Sustainable Science, Engineering and Technology (ASSET)
Vol. 7, No.2, April 2025, pp. 02502017-01 ~ 02502017-013
ISSN: 2715-4211 DOI: https://doi.org/10.26877/asset.v7i2.1658

Enhancing Pose-Based Sign Language Recognition: A Comparative Study of Preprocessing Strategies with GRU and LSTM

Toby Purbojo[1], Andreas Parama Wijaya[1,2*]

[1]Center of Mathematics and Society, Faculty of Science, Parahyangan Catholic University, Ciumbuleuit No.94, Bandung, 40141, West Java, Indonesia.

[2]Modeling and Simulation Laboratory, Faculty of Science, Parahyangan Catholic University, Ciumbuleuit No.94, Bandung, 40141, West Java, Indonesia.

*a.p.wijaya@unpar.ac.id

Abstract. Recognizing isolated sign language gestures is difficult due to differences in body proportions and missing pose landmarks. Many current methods struggle to work well across different signers. To solve this, we propose reference-based normalization, which reduces body shape differences by separately normalizing body parts such as the full body, arms, face, and hands. We tested this method using LSTM and GRU models on two datasets: a custom American Sign Language (ASL) dataset with one amateur signer, and the public WLASL dataset with various signers. On the custom dataset, the highest accuracy (97.75%) was achieved using LSTM with normalization applied only to the full body and hands, since the signer was consistent. For the WLASL dataset, adding normalization for the arms and face improved accuracy by 3.10% for LSTM and 0.77% for GRU. The GRU model reached the best WLASL result (74.03%) with fewer parameters than other advanced models. These findings show that reference-based normalization improves sign recognition performance and has potential for real-world use, especially in recognizing signs in continuous sequences.

Keywords: Isolated sign language recognition, Preprocessing, Feature engineering, Machine learning, Gated Recurrent Unit, Long Short-Term Memory

(Received 2025-02-07, Accepted 2025-03-26, Available Online by 2025-04-30)

1. Introduction

According to the WHO, approximately 430 million people worldwide have hearing loss, a number expected to rise to 700 million by 2050 [1]. For these individuals, sign language serves as a primary means of communication. However, many people, particularly those without disabilities, often underestimate its importance. Developing automated sign language recognition (SLR) systems can bridge this communication gap, enabling the deaf and hard of hearing to fully engage in society through

sign language. SLR can be categorized based on the sequences of signs processed. Isolated Sign Language Recognition (ISLR) focuses on recognizing one sign at a time, corresponding to a single word or phrase [2]. In contrast, Continuous Sign Language Recognition (CSLR) handles a stream of signs, such as those used in natural conversation or sentences [3]. While CSLR is more relevant for real-world applications, ISLR provides a critical foundation by enabling the detection of individual signs in sequence.

ISLR is typically framed as a time series classification problem, where sequences of human movements are classified into glosses. These movements are often recorded as videos and processed by ISLR models using either RGB data or pose-based representation. Recent studies leveraging RGB data have achieved high sign recognition accuracy [4-5]. These methods commonly employ Convolutional Neural Networks (CNNs) or pretrained CNNs to extract spatial features, followed by temporal models such as Long Short-Term Memory (LSTM) networks [6-7]. However, training with video data is computationally expensive due to its high dimensionality, making real-time deployment on mobile devices challenging.

Pose-based models offer an alternative by utilizing skeletal landmarks to represent the body, hands and facial positions of the signer. This data is lower in dimensionality compared to raw video, resulting in more computationally efficient processing. Pose-based frameworks, trained on large datasets, are robust to variations in lighting and background conditions. However, landmark data often contains inconsistencies, such as variations in position, scale, and missing keypoints [8-9]. Preprocessing methods are therefore essential to standardize and refine landmark data, ensuring reliable inputs for ISLR models.

Recent studies have explored various approaches to ISLR. For instance, Li et al. [10] introduced the WLASL dataset and evaluated the performance of RGB-based and pose-based models for word-level sign language recognition. Graph-based methods have also gained attention, with Naz et al. [11] utilizing a Graph Convolutional Network (GCN) to model landmarks as graph nodes, while Laines et al. [12] proposed a tree-structured representation to capture spatial relationships between keypoints. Multi-stream architectures, such as the one proposed by Maruyama et al. [13], integrate skeletal information, local image features, and full-body motion to improve recognition accuracy. Transformer-based models have also been introduced, with Boháček et al. [9] presenting a bounding-box normalization and augmentation method tailored for Transformer inputs. Additionally, Roh et al. [14] proposed a preprocessing framework that addresses missing hand landmarks using interpolation and improved normalization techniques.

In this work, we retain the original landmark data instead of transforming it into alternative representations. While previous normalization methods have addressed signer distance from the camera and positional variations, they often overlook differences in body proportions among signers. To address this limitation, we introduce a reference-based normalization method that accounts for body proportional bias in addition to signer distance and positional variations. Furthermore, prior studies on ISLR have used GRU models with OpenPose-extracted landmarks [10]. With the advent of newer pose estimation frameworks like Mediapipe and advancements in preprocessing techniques, it is essential to evaluate the continued relevance of recurrent architectures. While Transformers have gained attention for their performance, LSTM and GRU remain foundational components in many state-of-the-art models, excelling in sequential data tasks such as chatbot development, stock price prediction, and battery state-of-charge estimation in electric vehicles [15-17]. This study aims to analyze the impact of different preprocessing methods on landmark data using Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks.

2. Methods

This section outlines the dataset, the process of landmark extraction using Mediapipe Holistic, and preprocessing steps, including normalization, hand interpolation, and frame duplication. Additionally, it outlines the model architecture and experimental setup.

2.1. Dataset

To evaluate the effectiveness of our preprocessing methods, we utilized both a single-signer dataset and a multi-signer dataset. The controlled single-signer dataset allowed us to assess the impact of preprocessing without variability introduced by different signers, while the multi-signer dataset presented a more realistic and diverse setting. For the single-signer dataset, we created a custom ASL dataset to ensure consistency in recording conditions. This dataset was performed by a single amateur signer under natural lighting. It includes 20 different glosses, with 12 videos per gloss, resulting in a total of 240 videos.[1] All videos were recorded with a resolution of 360×270, at 30 frames per second, and with a uniform length of 30 frames.

For the multi-signer dataset, we selected the Word-Level American Sign Language (WLASL) dataset due to its signer diversity, variability, and differences in quality and aspect ratios. While larger datasets like ASL Citizen include more vocabulary and instances, studies have reported higher recognition accuracy on those datasets compared to WLASL [18], making WLASL a more challenging benchmark. The WLASL dataset was compiled from various educational sites and YouTube tutorials, resulting in diverse recording conditions and signer dialects [19]. WLASL is divided into subsets based on the number of glosses: $K = 100, K = 300,$ and $K = 2000$. These subsets are ordered by the largest number of instances per gloss, with smaller subsets containing glosses that have more samples. For our research, we selected the $K = 100$ subset to prioritize glosses with a larger number of instances. The WLASL100 subset includes 2038 videos, with lengths ranging from 15 frames to under 200 frames. It is worth noting that we used the version of the WLASL dataset available on Kaggle, which had already been processed using Mediapipe Holistic to extract landmarks.[2] While the dataset authors did not explicitly state this, an analysis of their accompanying notebook, which includes several datasets, suggests that approximately 1100 of the 2038 samples are high-quality originals, while the remainder are resized versions. A comparison of the datasets is outlined in Table 1.

Table 1. Comparison of Custom ASL Dataset and WLASL100.

Attribute	**Custom ASL Dataset**	**WLASL100**
Signers	Single signer	Diverse signers
Videos	240	2038
Frames per video	Fixed at 30 frames	Varies (15-200 frames)
Vocabulary size (glosses)	20	100

2.2. Mediapipe Holistic

Several pose estimation frameworks are available, including OpenPose, MM-Pose, Apple vision API, and Mediapipe. Among these, Mediapipe has demonstrated superior accuracy and computational efficiency [20]. Mediapipe Holistic, developed by Google, is a real-time pose detection framework that identifies keypoints representing landmarks on the human body [21]. In our study, we utilized 23 landmarks from Mediapipe Pose and all 21 landmarks for each hand from Mediapipe Hand, as shown in Figure 1. Each landmark is defined by (x, y, z) coordinates, corresponding to the width, height, and depth of the frame. However, we excluded the z-coordinate due to its unreliability, as noted in the documentation. This resulted in a total of 65 2D landmarks being extracted from each video frame. To ensure consistency, the x and y coordinates were normalized to a range of 0 to 1 within the frame. Undetected landmarks were assigned default coordinates of $(0,0)$.

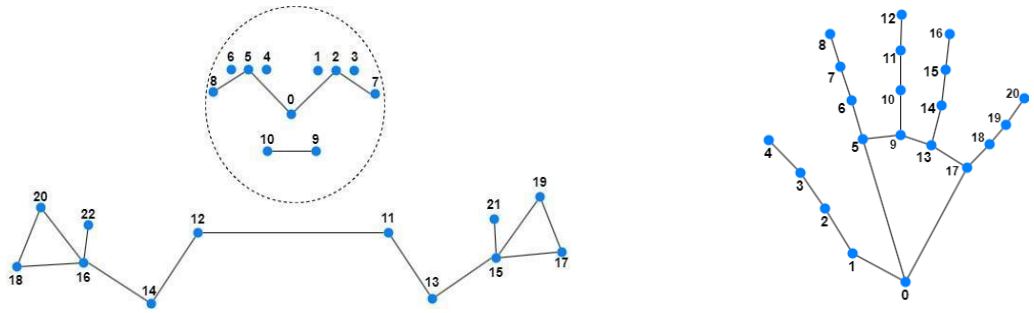

Figure 1. Mediapipe pose (left) and hand (right) landmarks index.

2.3. Reference-based normalization

Landmarks extracted from the video can vary significantly based on a person's position or body size. These variations in scale and position can hinder the model's ability to learn effectively. Normalization reduces such variations, enabling the model to focus on critical features for recognizing signs. In previous pose-based ISLR research, bounding box normalization was used to address positional bias [9]. Later, anchor-based normalization was introduced, demonstrating improved performance [14]. This method selects a specific landmark as an anchor point, adjusting and scaling the remaining landmarks based on the distance between the neck and face.

However, a limitation of anchor-based normalization is its inability to account for variations in body proportions. Scaling based solely on neck distance may cause the model to misinterpret signs due to differences in the size of the shoulders, arms, and hands. To address this issue, we implemented reference-based normalization as an extension of anchor-based normalization. The key improvement lies in mitigating body shape variability by normalizing the face, body, arms, and hands separately. This process is applied independently to each frame of the landmark data. A visual comparison of landmarks before and after normalization for the body, arm, and hands is presented in Figure 2. The next paragraph outlines the detailed steps of the normalization process.

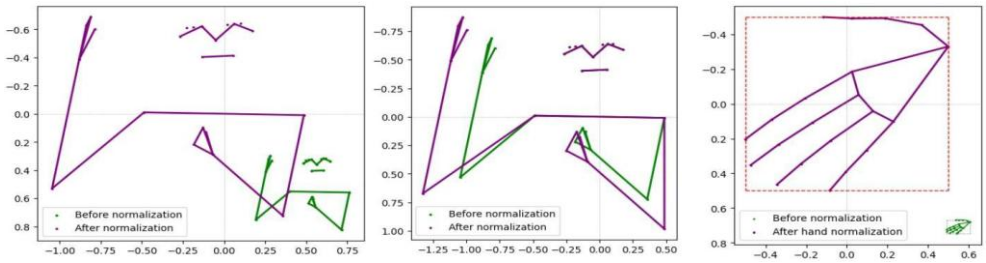

Figure 2. Result of applying full-body normalization (left), arm normalization (middle), and hand normalization (right). In each figure, the green lines represent landmarks before normalization, while the purple lines represent landmarks after applying the respective normalization. The red dotted line used in the right figure indicates the bounding box used in hand normalization.

Full-body normalization involves normalizing all body landmarks using the neck as the anchor point and scaling them by the distance between the left shoulder and the right shoulder. Let (x_k, y_k) represent the coordinates of the k-th pose landmark, where $k \in \{0, \ldots, 22\}$. The neck anchor point, denoted as (x_{neck}, y_{neck}), is calculated as the average of the left shoulder ($k = 12$) and the right shoulder ($k = 11$), using the formula:

$$(x_{neck}, y_{neck}) = \left(\frac{x_{11} + x_{12}}{2}, \frac{y_{11} + y_{12}}{2}\right).$$

After determining the neck landmark, each landmark is shifted relative to this reference point and scaled using the following formula:

$$(x'_k, y'_k) = \frac{(x_k - x_{neck}, y_k - y_{neck})}{\sqrt{(x_{11} - x_{12})^2 + (y_{11} - y_{12})^2}}.$$

Face normalization adjusts facial landmarks relative to the nose. Let $F = \{0, \dots, 10\}$ represent the set of facial landmarks. The normalization formula for these landmarks is as follows:

$$\left(x''_f, y''_f\right) = \left(x'_f - x'_0, y'_f - y'_0\right) \text{ for } f \in F.$$

Arm normalization adjusts the scale of arm landmarks to account for variations in arm proportions relative to shoulder length. Let $L = \{14,16,18,20,22\}$ represent the set of landmarks in the left arm. Their positions are calculated as follows:

$$(x''_l, y''_l) = \frac{(x'_l, y'_l)}{\sqrt{(x'_{12} - x'_{14})^2 + (y'_{12} - y'_{14})^2}} \text{ for } l \in L.$$

The normalization process for the right arm is similar to that of the left arm, using the set $R = \{13,15,17,19,21\}$ and scaling by the distance between landmarks 11 and 13.

Hand normalization differs from other normalizations because hands can take various poses, ranging from a flat palm to a clenched fist. The distance between two landmarks varies significantly with hand shape, making it an unreliable scaling factor. Instead, we use bounding box normalization, as propose in [9]. Let $H = \{0, \dots, 21\}$ represent the hand landmarks. The bounding box is defined by the minimum and maximum coordinates:

$$\begin{aligned} x_{min} &= x_h\,, \quad x_{max} = x_h \\ y_{min} &= y_h\,, \quad y_{max} = y_h\,. \end{aligned}$$

The center of the bounding box is calculated as:

$$(x_{center}, y_{center}) = \left(\frac{x_{min} + x_{max}}{2}, \frac{y_{min} + y_{max}}{2}\right).$$

Each hand landmark is shifted and scaled by the width and height of the bounding box:

$$(x'_h, y'_h) = \left(\frac{x_h - x_{center}}{x_{max} - x_{min}}, \frac{y_h - y_{center}}{y_{max} - y_{min}}\right), \quad h \in H.$$

This normalization ensures that the hand landmarks are always bounded within a box with coordinates $x \in [-0.5,0.5]$ and $y \in [-0.5,0.5]$.

2.4. Hands interpolation

Missing or undetected hand landmarks are a common issue in sign language datasets due to rapid hand movements or occlusion. To address this, we applied hand interpolation based on the method described in [14]. This approach uses linear interpolation across time frames to fill in the gaps in the hand keypoints. For interpolation, missing landmarks require both a preceding and a succeeding detected landmark. To ensure that missing landmarks are always bounded by detected ones, we initialize the first and last frames with the average values of the detected landmarks. Formally, given a hand landmark $f_t = (x_t, y_t)$ at the t-th time frame, the interpolated hand landmark is calculated using the following conditional formula:

$$f_t' = \begin{cases} \dfrac{\beta f_{t-\alpha} + \alpha f_{t+\beta}}{\alpha + \beta} & \text{if } f_t = (0,0) \\ f_t & \text{otherwise,} \end{cases}$$

where $f_t = (0,0)$ represent a missing landmark, α and β are the minimum positive integers such that the hand landmark in the $(t-\alpha)$-th and $(t-\beta)$-th frames are detected.

2.5. Frame duplication

To ensure a fair and consistent evaluation of different preprocessing techniques, our goal is to standardize the number of frames across all samples. The data used in this research, WLASL100, consists of instances with varying time lengths, ranging from as few as 15 frames to just under 200 frames. To address this variability, we applied frame duplication and truncation to standardize the input sequence length. Specifically, we extended the instances by replicating the landmark data until they exceeded 200 frames. After lengthening the instances, we truncated the excess frames to ensure that all samples contained exactly 200 frames. In contrast, our custom ASL dataset has a fixed length of 30 frames, so no frame duplication or truncation was applied.

2.6. Data preprocessing workflow

The data preprocessing workflow is illustrated in Figure 3. The process begins with two datasets: WLASL and a custom ASL dataset. These videos are processed with Mediapipe, which extracts landmarks from each frame. Since the number of frames varies across videos, frame duplication is applied to WLASL dataset, while the custom ASL dataset bypasses this step. The data is then processed into eight different configurations, each representing a unique combination of preprocessing techniques. The configuration with the complete set of preprocessing methods is also shown in Figure 3, providing a detailed breakdown of its steps. Normalization is applied sequentially to each frame, starting with full-body normalization, followed by face normalization, arm normalization, and hand normalization. Next, hand interpolation is performed by considering the entire video sample, rather than processing each frame independently. Once all configurations are generated, they are modeled and tested.

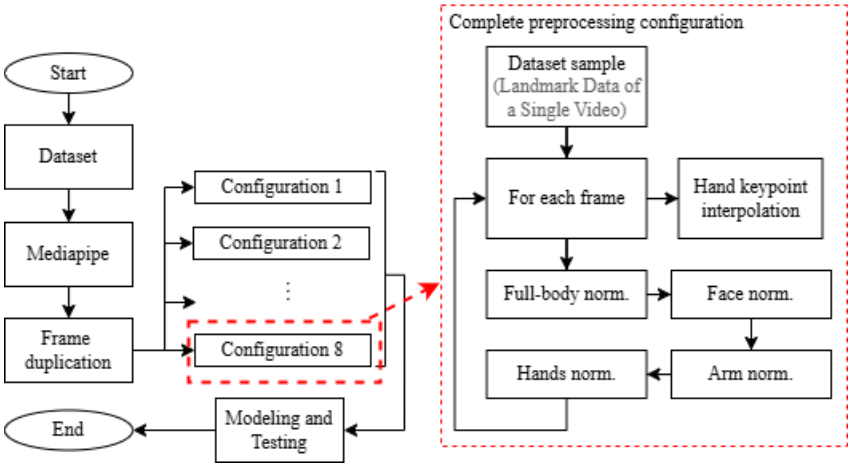

Figure 3. The preprocessing workflow and a detailed breakdown of preprocessing steps of configuration with complete preprocessing methods.

2.7. Model architecture and experimental setup

The preprocessing experiments will be conducted using recurrent neural networks (RNNs), specifically Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) models. Each dataset will be evaluated with different configurations to accommodate its characteristics. The custom ASL dataset, which has shorter sequences and fewer glosses, will use a simpler model with a single RNN layer. In

contrast, WLASL100, with longer sequences and more classes, will require a more complex architecture, as detailed in Table 2.

Table 2. Summary of LSTM/GRU model architecture for the Custom ASL (left) and the WLASL (right) data.

Layer	Description
Input	(30, 130 features)
LSTM/GRU	128 units
Dense	64 units, ReLu activation
Dense	20 classes, softmax activation

Layer	Description
Input	(200, 130 features)
LSTM/GRU	192 units, dropout=0.4
LSTM/GRU	128 units, dropout=0.4
Dense	100 classes, softmax activation

All the experiments were conducted on a Kaggle notebook with an NVIDIA T4 GPU. The models were trained using the Adam optimizer, categorical cross entropy as the loss function, and a learning rate of 10^{-3}. The experimental setup differed between the Custom ASL and the WLASL100, as detailed below:

- **Custom ASL data**. Training was carried out for 50 epochs with a batch size of 8. The data was split with a ratio of 8:4, where 8 samples per class were used for training and 4 for testing. The evaluation metric was top-1 accuracy, averaged over 10 runs with different seeds.
- **WLASL100**. Training was carried out for 200 epoch with a batch size of 32. The predefined training, validation, and testing splits were used. During training, checkpoints were saved based on the maximum validation categorical accuracy. The evaluation metric was top-1 accuracy, averaged over 5 runs with different seeds.

Hyperparameters, including learning rate, batch size, and the number of hidden units, were selected based on preliminary experiments. Several configurations were tested manually, and the final values were chosen based on the best validation performance.

3. Results and Discussions

In this section, we evaluate the impact of the preprocessing techniques applied in our study and compare the results with other methods.

3.1. Comparing preprocessing methods

The effectiveness of various preprocessing configurations in model accuracy is summarized in Table 3. For the custom ASL dataset, the highest accuracy (97.75%) was achieved using full-body and hand normalization with the LSTM model. In contrast, the WLASL100 dataset achieved the highest accuracy (74.03%) with the GRU model, and all normalization methods were applied. In general, the GRU models outperformed the LSTM models in both datasets.

Table 3. Comparison of preprocessing methods with corresponding accuracy of LSTM and GRU model on the WLASL100 and Custom ASL datasets (± 95% confidence interval).

Dataset	Body parts normalization				Hands	Accuracy (%)	
	Full body	Face	Arm	Hand	Interpolation	LSTM	GRU
	✗	✗	✗	✗	✗	68.13 ± 3.09	76.0 ± 2.28
	✓	✗	✗	✗	✗	83.62 ± 1.28	87.6 ± 1.24
	✓	✓	✗	✗	✗	82.62 ± 1.17	88.4 ± 1.05
Custom ASL						83.38 ± 1.39	87.3 ± 1.52

	✓	✗	✓	✗	✗		
	✓	✗	✗	✓	✗	97.75 ± 0.64	96.5 ± 0.88
	✓	✗	✗	✓	✓	96.88 ± 0.77	97.0 ± 1.10
	✓	✓	✓	✓	✗	97.5 ± 1.08	96.3 ± 0.68
	✓	✓	✓	✓	✓	97.25 ± 0.55	95.9 ± 0.71
	✗	✗	✗	✗	✗	13.95 ± 0.24	22.33 ± 0.74
	✓	✗	✗	✗	✗	29.84 ± 2.05	43.41 ± 1.86
	✓	✓	✗	✗	✗	29.61 ± 2.64	44.03 ± 0.95
WLASL100	✓	✗	✓	✗	✗	29.53 ± 2.44	44.96 ± 1.07
	✓	✗	✗	✓	✗	63.64 ± 2.33	73.26 ± 0.64
	✓	✗	✗	✓	✓	62.87 ± 1.28	69.61 ± 1.09
	✓	✓	✓	✓	✗	66.74 ± 1.47	74.03 ± 1.05
	✓	✓	✓	✓	✓	65.27 ± 2.02	70.08 ± 1.21

3.1.1. Normalization

In the custom ASL dataset, full-body and hand normalization resulted in the best performance. This approach was particularly effective because the dataset consists of videos performed by a single individual, meaning there is no body proportionality bias across signers. As a result, further normalization of the arms and face did not improve the accuracy. In contrast, the WLASL dataset comprises videos performed by multiple signers with varying body proportions. In this case, the addition of arm and face normalization, alongside full-body and hand normalization, helped standardized landmarks across signers, as shown in Figure 4. This preprocessing method led to an improvement in accuracy of 3.10% for the LSTM model and 0.77% for the GRU model, as presented in Table 3. The highest accuracy was achieved by the GRU model, reaching 74.03%.

The impact of these preprocessing strategies aligns with findings from prior pose-based ISLR studies. Boháček et al. [9] and Roh et al. [14] employed bounding-box hand normalization, which proved highly effective in their studies. Additionally, Roh et al. [14] applied anchor-based normalization to the full body, significantly improving accuracy. This study extends that approach by applying anchor-based normalization separately to different body parts, ensuring that the model focuses on motion while minimizing noise from signer-specific body proportions.

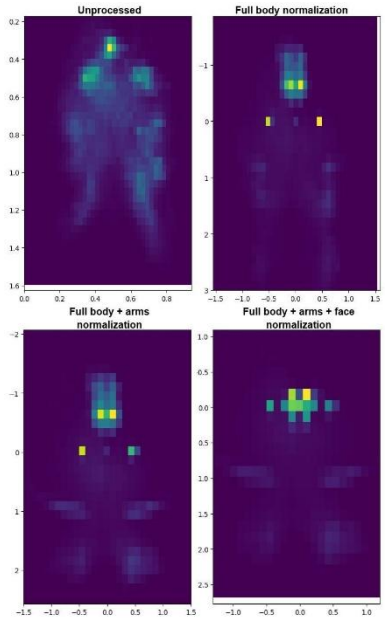

Figure 4. Heatmap comparison of WLASL100 data using different normalization methods. The heatmaps visualize the concentration of landmarks, where lighter colors indicate more observations, and darker color indicates fewer observations. The figure demonstrates how each normalization method makes the landmarks more compact and consistent, reducing variability and potential noise between signers.

3.1.2. Hand interpolation

Hand interpolation showed a limited impact on model performance. Although it recovered a significant number of missing landmarks, as shown in Table 4, it did not improve model accuracy. In the custom ASL dataset, applying interpolation led to a minor decrease in accuracy for the LSTM model and a 0.5% increase for the GRU model. In the WLASL100 dataset, interpolation decreased accuracy by 1.47% for LSTM and 3.95% for GRU. This finding contrasts with prior studies; Roh et al. [14] utilized hand interpolation with a transformer-based model and observed improvements in accuracy. The discrepancy suggests that the effectiveness of hand interpolation may depend on the model architecture, with transformers potentially benefiting more from additional interpolated data than recurrent models like LSTM and GRU.

Table 4. Comparison of the percentage of detected landmarks before and after applying hand interpolation.

Data	Region	Before	After
Custom ASL	Left hand	30.00%	45.00%
	Right hand	81.20%	100%
WLASL100	Left hand	33.50%	75.30%
	Right hand	62.20%	99.30%

Further analysis based on Figure 5 suggests that when the presenter's hands are resting outside the frame, the absence of hand landmarks (represented as zero values) helps the model identify when the hands are inactive. Recovering these missing landmarks through interpolation introduces redundant information, as there is no meaningful data to add when the hands are at rest. While hand interpolation can be beneficial when hands are undetected due to fast movement, the negative impact of reconstructing resting hand positions seems to outweigh this advantage, potentially explaining the drop in performance.

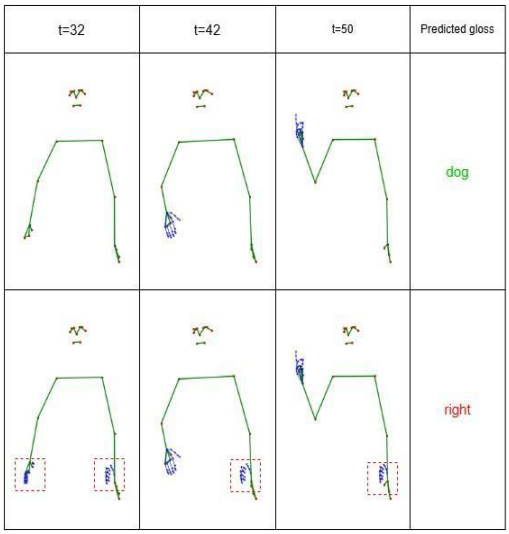

Figure 5. Comparison of inputs before (top) and after (bottom) hand interpolation. The red dotted box indicates interpolated hand landmarks. The predicted gloss are the results of two different models tested on the same data with different preprocessing methods.

3.2. Comparison with other methods

This section presents a comparison of our models (GRU and LSTM) against existing pose-based isolated sign language recognition methods, as shown in Table 5. Methods using OpenPose, such as Pose-GRU and GCNBERT, report lower accuracy compared to those using Mediapipe. In particular, Mediapipe-based models, including SignGraph, SL-TSSI, and Transformer Encoder, achieve higher accuracies, with Transformer Encoder reaching the highest at 83.26%.

Our proposed model, GRU, which uses Mediapipe, outperformed all models using OpenPose and the Apple Vision API, as well as SignGraph. However, it falls short in comparison to the Transformer Encoder and SL-TSSI in terms of accuracy. Several factors may explain the lower accuracy of our models. One key reason is that our model used fewer parameters: ours employed 0.32 M parameters, while SL-TSSI used 7.2 M and Transformer Encoder used 5.3M. Additionally, we used resized versions of videos from another source, leading to reduced landmark quality, which likely impacted accuracy.

Table 5. Comparison of our proposed model (GRU and LSTM) with existing methods using different pose estimation frameworks. The comparison includes models utilizing OpenPose, Apple Vision API, and Mediapipe. The number of trainable parameters is listed in millions (M).

Method	**Framework**	**Parameters**	**Acc. (\%)**
Pose-GRU [10]	OpenPose	-	46.51
Pose-TGCN [10]	OpenPose	-	55.43
GCNBERT [19]	OpenPose	-	60.15
SPOTER [9]	Vision API	5.92 M	63.18
SignGraph [11]	Mediapipe	0.62 M	72.09
SL-TSSI [12]	Mediapipe	7.2 M	81.47
Tf Encoder [14]	Mediapipe	5.3M	83.26
LSTM (ours)	Mediapipe	0.42 M	66.74
GRU (ours)	Mediapipe	0.32 M	74.03

3.3. Computational time

This subsection evaluates the computational time required for each stage of the pipeline: Mediapipe processing, preprocessing, and model inference. The experiments were conducted in a Kaggle Notebook environment utilizing an Intel Xeon(R) CPU 2.30GHz, with no GPU used for model inference. The Mediapipe and preprocessing times were measured using a sample of 3,000 frames, while the model inference time was averaged over 100 runs with an input of 200 frames per run. The final inference time per frame was calculated by dividing the total time by 200 frames. The results, shown in Table 6, indicate that Mediapipe is the most time-consuming stage, taking an average of 95.96 ms per frame. In contrast, preprocessing adds a negligible delay, ranging from 0 to 0.088 ms per frame, while model inference is similarly lightweight at 0.2878 ms per frame. The total processing time per frame ranges from 96.25 ms and 96.34 ms, resulting in an average frame rate of around 10.38 FPS. This shows that the preprocessing and model inference we implemented have a negligible impact on performance, with Mediapipe being the primary computational bottleneck.

Table 6. Processing time per frame for different preprocessing configurations (Fb: full-body normalization, Fc: face normalization, A: arm normalization, H: hand normalization, Intp: interpolation).

Preprocessing configuration	Preprocessing (ms/frame)	Mediapipe (ms/frame)	Inference (ms/frame)	Total (ms/frame)	Total (frame/s)
None	0	95.9637	0.2878	96.251	10.389
Norm. (Fb)	0.0093			96.261	10.388
Norm. (Fb,Fc)	0.0148			96.266	10.388
Norm. (Fb,A)	0.0303			96.282	10.386
Norm. (Fb,H)	0.0528			96.304	10.384
Norm. (Fb,A) + Intp.	0.0573			96.309	10.383
Norm. (Fb,Fc,A,H)	0.0826			96.334	10.381
Norm. (Fb,Fc,A,H) + Intp.	0.0877			96.339	10.380

4. Conclusions

In this study, we apply anchor-based normalization to independently normalize arms, face, and hands. This approach effectively minimizes body proportionality variations across different signers, making it particularly beneficial for datasets like WLASL that involve multiple performers. Among various preprocessing configurations, adding arm and face normalization to full-body and hand normalization resulted in a notable improvement in model performance. Specifically, it increased the accuracy of the LSTM model by 3.10% and the GRU model by 0.77%. Although hand interpolation successfully recovered a significant number of missing hand landmarks, our experiments revealed that it led to a decrease in overall model accuracy. This suggests that the recovered landmarks may introduce inconsistencies that adversely affect model prediction. These findings indicate that the normalization process can yield promising results when applied to CSLR, particularly in detecting individual signs in sequence.

Furthermore, our methods achieve competitive accuracy compared to existing research while utilizing significantly fewer parameters, demonstrating the efficiency of our approach. Compared to models such as SPOTER and SL-TSSI, which require over five million parameters, our LSTM and GRU models achieve accuracies of 66.74% and 74.03%, respectively, with fewer than 0.5 million parameters. This highlights the effectiveness of our preprocessing techniques and model design in improving performance while maintaining computational efficiency. The total processing time per frame for the model to process until output ranges from 96.25 ms to 96.34 ms, resulting in an average frame rate of around 10.38 FPS.

Acknowledgment
The authors would like to express their gratitude to Prof. Marcus Wono Setya Budhi for valuable discussions and insights that contributed to the development of this study. We also acknowledge Modeling and Simulation Laboratory in Catholic Parahyangan University for providing the necessary computational resources that facilitated the experiments conducted in this study.

References

[1] World Health Organization, "Deafness and hearing loss." Accessed: Nov. 14, 2024. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss.

[2] N. Sarhan and S. Frintrop, "Unraveling a decade: a comprehensive survey on isolated sign language recognition," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 3210–3219.

[3] S. Alyami and H. Luqman, "A Comparative Study of Continuous Sign Language Recognition Techniques," *arXiv preprint arXiv:2406.12369*, 2024.

[4] H. Hu, W. Zhao, W. Zhou, Y. Wang, and H. Li, "SignBERT: Pre-training of hand-model-aware representation for sign language recognition," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 11087–11096.

[5] R. Zuo, F. Wei, and B. Mak, "Natural language-assisted sign language recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 14890–14900.

[6] S. Yang and Q. Zhu, "Continuous Chinese sign language recognition with CNN-LSTM," in *Ninth international conference on digital image processing (ICDIP 2017)*, 2017, pp. 83–89.

[7] D. Kothadiya, C. Bhatt, K. Sapariya, K. Patel, A.-B. Gil-González, and J. M. Corchado, "Deepsign: Sign language detection and recognition using deep learning," *Electronics (Basel)*, vol. 11, no. 11, p. 1780, 2022.

[8] M. Pu, C. Y. Chong, and M. K. Lim, "Robustness evaluation in hand pose estimation models using metamorphic testing," in *2023 IEEE/ACM 8th International Workshop on Metamorphic Testing (MET)*, 2023, pp. 31–38.

[9] M. Boháček and M. Hrúz, "Sign pose-based transformer for word-level sign language recognition," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2022, pp. 182–191.

[10] D. Li, C. Rodriguez, X. Yu, and H. Li, "Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2020, pp. 1459–1469.

[11] N. Naz, H. Sajid, S. Ali, O. Hasan, and M. K. Ehsan, "Signgraph: An efficient and accurate pose-based graph convolution approach toward sign language recognition," *IEEE Access*, vol. 11, pp. 19135–19147, 2023.

[12] D. Laines, M. Gonzalez-Mendoza, G. Ochoa-Ruiz, and G. Bejarano, "Isolated sign language recognition based on tree structure skeleton images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 276–284.

[13] M. Maruyama, S. Ghose, K. Inoue, P. P. Roy, M. Iwamura, and M. Yoshioka, "Word-level sign language recognition with multi-stream neural networks focusing on local regions," *arXiv preprint arXiv:2106.15989*, 2024.

[14] K. Roh, H. Lee, E. J. Hwang, S. Cho, and J. C. Park, "Preprocessing Mediapipe Keypoints with Keypoint Reconstruction and Anchors for Isolated Sign Language Recognition," in *Proceedings of the LREC-COLING 2024 11th Workshop on the Representation and Processing of Sign Languages: Evaluation of Sign Language Resources*, 2024, pp. 323–334.

[15] I. D. Raharjo and E. R. Subhiyakto, "Implementing Long Short Term Memory (LSTM) in Chatbots for Multi Usaha Raya," *Advance Sustainable Science Engineering and Technology*, vol. 6, no. 4, p. 2404018, 2024.
[16] J. Xiao, T. Deng, and S. Bi, "Comparative Analysis of LSTM, GRU, and Transformer Models for Stock Price Prediction," in *Proceedings of the International Conference on Digital Economy, Blockchain and Artificial Intelligence*, 2024, pp. 103–108.
[17] J. Hong *et al.*, "Multi-forword-step state of charge prediction for real-world electric vehicles battery systems using a novel LSTM-GRU hybrid neural network," *Etransportation*, vol. 20, p. 100322, 2024.
[18] A. Desai *et al.*, "ASL citizen: a community-sourced dataset for advancing isolated sign language recognition," *Adv Neural Inf Process Syst*, vol. 36, pp. 76893–76907, 2023.
[19] D. Li, C. Rodriguez, X. Yu, X. Li, Y. Huang, and D. Metaxas, "WLASL: A Large-Scale Dataset for Word-Level American Sign Language Recognition." Accessed: Jan. 21, 2025. [Online]. Available: https://github.com/dxli94/WLASL
[20] M. De Coster, E. Rushe, R. Holmes, A. Ventresque, and J. Dambre, "Towards the extraction of robust sign embeddings for low resource sign language recognition," *arXiv preprint arXiv:2306.17558*, 2023.
[21] Google AI, "MediaPipe Holistic," Accessed: Jan. 21, 2025. [Online]. Available: https://github.com/google-ai-edge/mediapipe/blob/master/docs/solutions/holistic.md
[22] A. Tunga, S. V. Nuthalapati, and J. Wachs, "Pose-based sign language recognition using GCN and BERT," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2021, pp. 31–40.