



RESEARCH ARTICLE

OPEN ACCESS

AVATAR-DRIVEN SYNTHETIC DATA AUGMENTATION FOR ENHANCED MULTILINGUAL SIGN LANGUAGE RECOGNITION

Younes Ouargani*¹ and Noussaima El Khattabi²

^{1,2}Laboratory of Conception and Systems (Electronics, Signals, and Informatics), Faculty of Science, Mohammed V University, Rabat, Morocco.

¹<http://orcid.org/0000-0002-0804-9218>, ²<http://orcid.org/0009-0009-3390-275X>

E-mail: *younes_ouargani@um5.ac.ma, e.noussaima@um5r.ac.ma

ARTICLE INFO

Article History

Received: February 12, 2026

Reviewed: March 13, 2026

Accepted: April 5, 2026

Published: May 29, 2026

Keywords:

Sign Language,
Sign Language Recognition,
Synthetic Dataset,
Data Augmentation.

ABSTRACT

Sign Languages serve as the primary means of communication for millions of deaf and hard-of-hearing individuals. Despite this significance, the development of automatic Sign Language Recognition systems is limited due to the scarcity of large-scale, high-quality video datasets. This is mainly attributed to the inherently resource-intensive process required to collect such datasets, often requiring fluent signers, controlled recording environments, and rigorous annotation. In response to these challenges, this study introduces a novel method for generating synthetic sign language datasets. Supporting both isolated and continuous signing, our method enables the creation of diverse and scalable datasets, with precise control over factors such as signer appearance, sign content, and environment. First, we determine the most suitable task for evaluation by comparing three architectures: CNN-Transformer, CNN-GRU, and CNN-LSTM. With the CNN-LSTM emerging as the most effective, achieving recognition rates of 75%, 90%, 100%, and 100% for Top-1, Top-2, Top-3, and Top-5 metrics, respectively. Our proposed method is then compared to standard data augmentation techniques demonstrating remarkable improvements, with recognition rate improvements ranging from 87.5% increase in Top-1, to a 42.85% improvement in Top-5. 10-fold cross-validation was then used to confirm the robustness of the resulting model in real-world scenarios.



Copyright ©2026 by authors and Galileo Institute of Technology and Education of the Amazon (ITEGAM). This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

I. INTRODUCTION

Millions of deaf and hard-of-hearing individuals worldwide rely on sign languages as their primary means of communication; not only does it play a crucial role in their lives, but they also rely on it in their daily lives, education, and social integration. The World Health Organization estimates over 1.5 billion individuals worldwide live with some degree of hearing loss, with projections indicating that this number could reach 2.5 billion by 2050 [1]. Despite its vital importance, Sign Language Recognition still has an underdeveloped technological support. Sign language recognition systems hold great potential to revolutionize accessibility in sectors such as education, healthcare, public services, and digital communication. However, their progress is severely hindered by the scarcity of large, high-quality Sign Language datasets. This is mainly due to sign language recognition systems being heavily reliant on deep learning methods that require large-scale, diverse, and well-annotated datasets.

Unlike textual or spoken language datasets, which can often be compiled from readily available digital resources, sign language datasets cannot leverage already existing digital resources. Thus, creating these datasets is inherently resource-intensive, requiring substantial investment in data collection infrastructure and annotation expertise, often involving the recruitment of fluent signers, controlled environments, signer demographic diversity, and expert annotators for gloss labeling. Furthermore, technical challenges such as multi-camera setups and depth sensors add to the cost and complexity of creating such datasets. The MS-ASL dataset [2], one of the most extensive collections available for American Sign Language (ASL), with over 25,000 labeled video clips across 1,000 sign classes performed by 200 distinct signers. While large in scope compared to many earlier datasets, the dataset was collected from publicly available online videos from platforms like YouTube.

Although scalable, this strategy introduced challenges in annotation accuracy, label noise, signer variation, and uncontrolled recording conditions such as inconsistent lighting, backgrounds, and camera angles. The dataset construction required highly specialized manual verification to ensure label correctness, making it labor-intensive despite the automation in video sourcing. When it comes to German Sign Language (DGS), the RWTH-PHOENIX-Weather series is widely used for sign language recognition. This dataset further illustrates the time-intensive nature of producing high-quality datasets. The original RWTH-PHOENIX-Weather corpus [3] contains 3.25 hours of interpreted weather forecasts with 911 sign glosses from 7 signers. Despite its modest size, the dataset took around two years to compile due to the need for labor-intensive manual alignment between spoken German transcripts and DGS sign glosses. The later PHOENIX-2014 release [4] further expanded the vocabulary to 1,558 signs while increasing the dataset duration to 10.73 hours with 9 signers. However, this extension took approximately two years, primarily due to the complexity of multi-modal alignment and signer-specific variation across broadcast content. The How2Sign project [5] was created to support more natural environments, especially for sentence-level and continuous ASL recognition. The dataset includes approximately 16,000 English words and 35,000 sentences, collected over a six-month period with 11 signers across 65 recording sessions.

Despite its ambitious scale, How2Sign has several limitations, including limited variation in signer demographics (e.g., race, skin tone) and environmental factors (e.g., lighting, background, and camera quality). These constraints limit the generalization of models trained on the dataset, especially in diverse or in-the-wild scenarios. The difficulty in collecting large, diverse, and well-annotated sign language datasets stems from several factors: the need for proficient signers, the complicated annotation due to the lack of writing standards in many sign languages, and the limited availability of publicly available sign language data compared to textual or spoken language. Moreover, the requirement for multi-camera setups or depth sensors in some datasets [6] (e.g., RGB-D or motion capture-based datasets) introduces additional hardware costs and technical overhead, often limiting replication and scalability. Given these challenges, synthetic data generation emerges as a promising solution. Synthetic datasets offer scalability, controlled conditions, and consistent annotations, addressing the limitations of real-world data collection. This approach has shown utility in adjacent fields such as 3D human action recognition [7],[8] and pose estimation [9], suggesting potential for advancing Sign Language Recognition.

This paper makes several significant academic contributions to the field of Sign Language Recognition:

- We propose a novel signer-agnostic and language-independent method for sign language data generation.
- We conduct a systematic comparison of three architectures (CNN-Transformer, CNN-GRU, and CNN-LSTM) to identify the most effective model for sign language recognition tasks.
- We demonstrate through comparative analysis significant improvements in sign language recognition by leveraging the proposed method compared to standard data augmentation techniques.
- We validate the practical effectiveness of our approach by conducting a comprehensive evaluation using 10-fold cross-validation across various Top-K metrics.

The remainder of this paper is structured into five sections. Section 2 presents our proposed pipeline architecture, Section 3 describes the experimental setup, Section 4 reports the results, Section 5 discusses the findings, and Section 6 concludes with future directions.

II. PROPOSED PIPELINE

To address the challenges associated with creating sign language datasets, we propose a rule-based data generation pipeline designed for both continuous and isolated sign language datasets. This system emphasizes simplicity, accessibility, and reproducibility, leveraging skeletal animation and modular action composition to synthesize realistic yet computationally efficient corpora. The pipeline is signer-agnostic, enabling it to operate independently of specific signers, and is highly extensible to new languages and vocabularies. In this section, we provide a detailed explanation of each module within the proposed architecture, detailing its functionalities and interconnections.

II.1 OVERVIEW

Figure 1 presents a detailed overview of our proposed architecture. The initial step involves generating valid Sign Markup Language (SigML) definitions, which can be achieved through two modes of operation. First, direct SigML sign definitions may be provided without requiring pre-processing steps. Alternatively, sign definitions can be supplied in Hamburg Notation System (HamNoSys) format, which are then pre-processed using the HamNoSys2SigML tool [10] to produce accurate SigML representations. Concurrently, our architecture incorporates a viewpoint generation module, represented in green (Figure 1). This module operates by first employing a viewpoint augementer to generate multiple perspectives through systematic adjustments of the camera angle. Subsequently, a viewpoint renderer generates 3D visualizations of these viewpoints for user validation and verification.

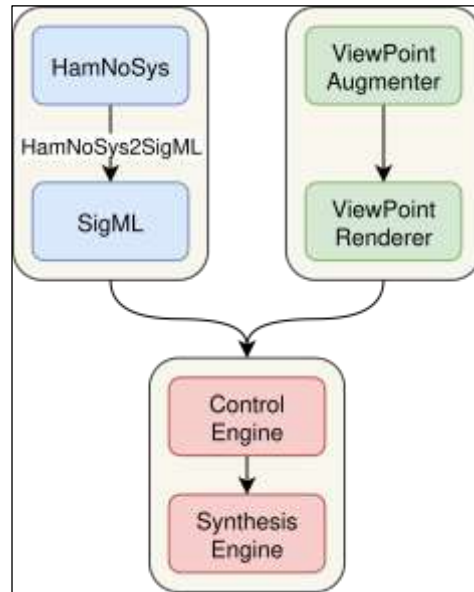


Figure 1: Our proposed data generation architecture.
Source: Authors, (2026).

The processed SigML files and viewpoints are then integrated into the synthesis module, highlighted in pink (Figure 1). Within this module, a control engine ensures smooth and efficient data processing by sequentially managing input parameters and directing them through the synthesis process until all data is fully processed. The synthesis engine, downstream from the control engine, assumes responsibility for rendering signs according to specified camera angles, ensuring accurate and precise representation of gestures in video format.

II.2 SIGN DEFINITION AND VOCABULARY

The primary standard for sign definition in our system is HamNoSys due to its universal compatibility, iconicity, economy, automation support, formal syntax, and extensibility. This guarantees support for diverse languages with efficient and accurate representation, and simplifies integration into our multilingual pipeline. Furthermore, HamNoSys reduces human error as it enables automated interpretation and manipulation of sign descriptions, and maintains robustness and adaptability to evolving linguistic needs. For visually coherent sign language animations, HamNoSys's SigML format is used. SigML is an XML-based representation of HamNoSys that provides a detailed description of both manual and non-manual components, and guarantees their synchronization. This includes handshape, orientation, location, and movement for manual components, and facial expressions, eye gaze, mouth gestures, and head movements for non-manuals.

II.3 CWASA AVATAR ANIMATOR

CWA Signing Avatar[11],[12] is the synthesis engine of choice in our pipeline; it is tasked with converting SigML definitions into fluid, lifelike signing sequences. It ensures the accurate representation of complex signs by integrating both manual and non-manual elements, including hand movements, facial expressions, eye gaze, and mouth gestures.

II.4 EXPLOITATION

The exploitation phase of our pipeline involves the orchestrated integration and execution of components designed to generate a comprehensive dataset for sign language animations. This subsection provides a detailed explanation of how various elements are integrated within our pipeline to facilitate this process.

II.4.1 Viewpoint Augmentation Via Uniform Spherical Perturbations

Sign language recognition benefits heavily from varied camera angles in the dataset as they promote the development of robust, inclusive, and generalizable systems. Training models solely on fixed viewpoint datasets tends to produce biased models that perform poorly when exposed to viewpoint variation in real-world scenarios. The introduction of diverse camera angles provides the model the ability to capture the full spectrum of manual and non-manual gestures and movements that have significant variations in relation to camera angles. Furthermore, viewpoint diversity enhances model generalization by delineating the patterns invariant to perspective changes and mitigates the potential biases caused by single viewpoint datasets.

In this context, uniform perturbations were chosen as a method to introduce diversity into training datasets. This is achieved by introducing controlled random variants to the spherical coordinates used to define the camera position; notably, the polar and azimuthal angles (θ , ϕ) used to define camera viewpoints in 3D space. Uniform perturbation ensures an even distribution of viewpoints across all possible angles, hence preventing bias towards specific directions. This approach simulates a wide range of realistic scenarios encountered by the model, especially those with unpredictably varying camera positions.

Uniform perturbations also guarantee a consistent and fair augmentation process, as all points within the specified range of the sphere have an equal sampling probability. This approach, given enough sampling points, prevents specific regions from being over-represented, ensuring a balanced distribution of viewpoints, which is a crucial aspect for creating generalizable models across diverse conditions. However, the perturbation magnitude is an important aspect to consider to preserve a realistic angular range and an ameliorative effect on diversity without introducing unrealistic or non-beneficial viewpoints.

Our proposed perturbation mode is defined as follows: starting with an initial set of points represented in spherical coordinates (θ, ϕ) , where $\theta \in [0, \pi]$ is the polar angle (measured from the positive z-axis), and $\phi \in [0, 2\pi)$ is the azimuthal angle (on the xy-plane). Our augmentation method consists of applying uniform perturbations to both angles. This approach helps enrich the dataset by generating additional samples in the vicinity of the original points while preserving the spherical coordinates conventions. For each original coordinate pair (θ, ϕ) , perturbations $\Delta\theta$ and $\Delta\phi$ are drawn independently from uniform distributions centered at zero:

$$\Delta\theta \sim U(-a_\theta, a_\theta) \quad (1)$$

$$\Delta\phi \sim U(-a_\phi, a_\phi) \quad (2)$$

where $a_\theta, a_\phi > 0$ are hyperparameters controlling the maximum angular deviation.

The perturbed coordinates are then computed as:

$$\theta' = \theta + \Delta\theta \quad (3)$$

$$\phi' = \phi + \Delta\phi \quad (4)$$

Due to the polar angle interval constraint $\theta \in [0, \pi]$, perturbations that cause the new angle θ' to fall outside this range must be rectified. A reflection strategy is used to achieve boundary correction, and to maintain the point's correct position on the sphere, an azimuthal rotation is applied:

$$\text{if } \theta' < 0, \theta' \leftarrow -\theta', \phi' \leftarrow \phi' + \pi \quad (5)$$

$$\text{if } \theta' > \pi, \theta' \leftarrow 2\pi - \theta', \phi' \leftarrow \phi' + \pi \quad (6)$$

This reflection ensures $\theta' \in [0, \pi]$ while shifting the azimuthal angle ϕ' by π radians to preserve the points' spatial orientation on the sphere. Since the azimuthal angle ϕ is periodic with period 2π , the modulo operation is used to normalize ϕ' to the interval $[0, 2\pi)$:

$$\phi' \leftarrow \phi' \bmod 2\pi \quad (7)$$

The overall augmentation procedure can be summarized as follows:

$$\left\{ \begin{array}{l} \Delta\theta \sim U(-a_\theta, a_\theta) \\ \Delta\phi \sim U(-a_\phi, a_\phi) \\ \theta' = \theta + \Delta\theta \\ \phi' = \phi + \Delta\phi \\ \text{if } \theta' < 0 : \theta' \leftarrow -\theta', \phi' \leftarrow \phi' + \pi \\ \text{if } \theta' > \pi : \theta' \leftarrow 2\pi - \theta', \phi' \leftarrow \phi' + \pi \\ \phi' \leftarrow \phi' \bmod 2\pi \end{array} \right. \quad (8)$$

Our proposed method ensures that augmented spherical coordinates remain valid and uniformly distributed around the original points, facilitating robust data augmentation for applications involving spherical data representations.

II.4.2 Viewpoint Rendering

The viewpoint renderer is a critical component in the dataset generation pipeline; it's designed to convert augmented spherical coordinates into Cartesian coordinates and render them in a 3D environment. By enabling users to render the augmented coordinates, this tool allows the user to validate the generated viewpoints, ensure their accuracy and suitability for training machine learning models, and understand how different perturbations affect the generated dataset's diversity. The viewpoint renderer is an essential bridge between the abstract spherical coordinates and tangible 3D visualizations. Converting the augmented viewpoints into Cartesian coordinates facilitates their placement in a 3D scene. The conversion from spherical coordinates (r, θ, ϕ) to Cartesian coordinates (x, y, z) is done as follows:

$$x = r \sin \theta \cos \phi \quad (9)$$

$$y = r \sin \theta \sin \phi \quad (10)$$

$$z = r \cos \theta \quad (11)$$

where r is the radius (distance from the origin), θ is the polar angle (angle from the positive z-axis), and ϕ is the azimuthal angle (angle around the z-axis). The Cartesian coordinates are then rendered in a user-friendly, interactive interface, allowing the users to manipulate the view by zooming or panning to examine each viewpoint.

The viewpoints can also be viewed simultaneously, allowing comparison of their spatial distribution. Adjusting the a_θ and a_ϕ hyperparameters allows the users to control perturbation strength in both polar and azimuthal directions. Users can also vary viewpoint density by varying the number of viewpoints. Figure 2 displays an example visualization of the viewpoint renderer module. The red pyramid presents the original camera position, the blue pyramids present the generated camera positions, and the green sphere presents the target avatar position.

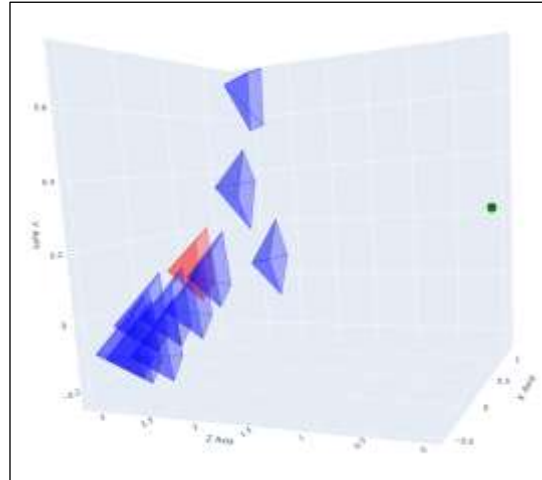


Figure 2: ViewPoint renderer visualization.
Source: Authors, (2026).

II.4.3 Control Engine

The control engine serves as the central orchestration system of our pipeline, managing data flow between key modules: viewpoint generation, pre-processing (HamNoSys to SigML conversion), and synthesis using CWASA. It ensures seamless integration by coordinating inputs at appropriate times. Initially, the engine interfaces with the viewpoint generation module, which systematically adjusts camera positions to generate diverse perspectives essential for realistic sign language representation. Concurrently, the preprocessing module converts signs from HamNoSys to SigML format using the hamnosys2sigml tool, ensuring compatibility with subsequent stages. The control engine integrates these processed definitions into the pipeline.

The prepared inputs are sequentially fed to the synthesis engine (CWASA), where camera angles and sign definitions are combined to produce visual representations. Accuracy here is crucial for reliable dataset generation. Beyond data transfer, the engine ensures smooth operation by coordinating input flow, minimizing disruptions between processing stages. Upon completion of each sign's articulation, the control engine captures video sequences and organizes them into a structured dataset. Each sign is stored in its own directory, with filenames incorporating metadata such as avatar name, spherical camera position, and direction for easy retrieval and analysis.

III. EXPERIMENTAL SETUP

III.1 DATASET

To establish a baseline for evaluating our introduced method, we created a German Sign Language (DGS) dataset focusing on numerical signs from 0 to 9. As avatar availability presents a common constraint in sign language datasets, we leverage it to demonstrate the capabilities of our proposed methodology. Each sign was captured at 10 distinct camera angles using two different avatars, with elevation angle a_θ set to 20 degrees and polar angle a_ϕ set to 30 degrees, ensuring comprehensive coverage of spatial and temporal dynamics. The resulting dataset served as the foundation for training multiple baseline models, enabling us to thoroughly validate and compare against our introduced method in subsequent experiments. The dataset was split into a 90% subset for training and a 10% subset for validation.

III.2 MODEL SELECTION

Three distinct architectures, aimed to capture both spatial and temporal features for sign language gestures, were explored while developing models for our baseline evaluation. All compared models incorporated a common base architecture, incorporating time-distributed layers to effectively process the sequential nature of the input data. The spatial processing was handled by a VGG16 [13] network followed by Global Average Pooling (GAvP), a fully connected layer with 64 units activated by ReLU, and a dropout rate of 50% to prevent overfitting. A subsequent temporal processing component is employed, followed by another dropout layer set to 50%. Then, a fully connected layer with 32 units using ReLU activation, followed by a final output layer.

The main difference between the models lies in the temporal processing component. While all models follow the stated architecture, the main difference between them is the type of memory layer used: LSTM, GRU, or Transformer. This not only allows us to assess how temporal dependencies are captured by the different approaches, but also provides an indication of the overall performance and generalization capabilities of each method. The resulting architectures are as follows:

- **CNN-Transformer [13],[14]:** Combining convolutional networks with transformer blocks, this architecture uses the CNN component to extract spatial features from the visual representations; and to process the sequence of features, the transformer is used to capture

long-range dependencies in the temporal domain, enhancing the model's ability to understand complex sign language temporal patterns.

- **CNN-GRU [15]:** In addition to convolutional layers, this architecture employs Gated Recurrent Units for sequential processing of the features. By maintaining a hidden state that evolves over time, GRUs allow capturing temporal dynamics, ensuring temporal aspects of signing are effectively modeled.
- **CNN-LSTM [16]:** Comparable to the CNN-GRU architecture, this architecture employs Convolutional Neural Networks to extract spatial features. However, instead of GRUs, it uses Long Short-Term Memory (LSTM) units that are particularly effective at capturing temporal nuances while selectively forgetting irrelevant information.

Each architecture was carefully designed to leverage different strengths of CNNs and temporal processing units, providing a comprehensive evaluation of their suitability for sign language recognition. The use of time-distributed layers ensured that each model could effectively process the sequential input data, while the choice of decoder (CNN-Transformer, CNN-GRU, or CNN-LSTM) allowed us to explore various approaches to sequence modeling in the context of sign language recognition. In our training protocol, we employed the Adam optimizer [17], a widely adopted optimization algorithm that effectively combines the benefits of AdaGrad and RMSProp to adaptively adjust learning rates during the training process.

The learning rate was set to 0.0005. This optimizer was chosen for its ability to maintain robust performance across varying scales of parameters, which is crucial given the complexity of our model architecture, and especially for its good performance in sign language translation and recognition tasks[18, 19]. The input size was set to 244x244 pixels, reflecting the dimensions of the image frames utilized in our dataset. This resolution was selected to ensure a balance between capturing sufficient spatial detail and maintaining computational efficiency during training. Each sequence comprises a length of 20 frames, which is essential for capturing the temporal dynamics inherent in sign language recognition tasks. To mitigate overfitting, we implemented early stopping [20] with a patience of 100 epochs. This mechanism monitors the validation accuracy throughout the training process and halts it if no improvement is observed within 100 consecutive epochs.

III.3 REAL-WORLD PERFORMANCE ASSESSMENT

Assessing the generalization performance of our final model was achieved through 10-fold cross-validation. To achieve this, the target dataset is split into ten equal-sized subsets. In each step, nine of the ten subsets are used for training while the remaining subset is used for validation. Repeating this process ten times causes each data point to be included in the validation set exactly once, ensuring a comprehensive assessment of model performance on unseen data. A dynamic adaptive learning rate has been implemented to enhance the model's performance by dynamically changing the learning rate during training. The strategy consists of a delayed learning rate reduction, by monitoring the validation loss, the learning rate is reduced by a factor of 0.2 when no improvement is detected for over 20 epochs, starting from epoch 50. With a cooldown period of 0, allowing the immediate resumption of the strategy after a learning rate adjustment. The minimum learning rate is set to 5e-5 to avoid excessively small learning rates. Furthermore, for additional training stability enhancement, our strategy resets the target optimization parameters on each learning rate update, ensuring an effective parameter exploration even after learning rate adjustment.

III.4 EVALUATION METRICS

For a comprehensive evaluation of our models, the top-k recognition rate metric is used. It's a critical metric that allows measuring the probability of the top-k predicted labels by the models containing the true class label. It allows a better understanding of a model's performance compared to relying on the single best guess. The formula for calculating the top-K recognition rate is:

$$Top - k = \frac{1}{N} \sum_{i=1}^N I(y_i \in \{\hat{y}_{i(1)}, \hat{y}_{i(2)}, \dots, \hat{y}_{i(k)}\}) \quad (11)$$

Where N is the total number of instances, y_i is the true class label for instance i, $\{\hat{y}_{i(1)}, \hat{y}_{i(2)}, \dots, \hat{y}_{i(k)}\}$ the top K predicted labels for instance i, and I is an indicator function that returns 1 if the condition inside the parentheses is satisfied and 0 otherwise. This metric is mainly used in multi-class classification problems, as the model outputs a ranked list of predictions. Both the model's confidence and its ability to identify relevant class labels beyond the highest-ranked prediction can be assessed from the true label being in the top-K predictions of a model. In sign language, where signs can often be similar and confused by models, this top-K metric becomes of tremendous importance as it not only provides a reliable indication of the model's performance, but also gives an indication of areas of improvement by providing a better insight into how the model can distinguish between closely related gestures.

Table 1: Top-K accuracy comparison of different architectures.

Architecture	Top-1	Top-2	Top-3	Top-5
CNN-GRU	55%	75%	85%	95%
CNN-Transformer	20%	20%	30%	50%
CNN-LSTM	75%	90%	100%	100%

Source: Authors, (2026).

IV. EXPERIMENTAL RESULTS

IV.1 MODEL SELECTION

The top-K recognition rates of our tested models are presented in Table 1, with the K value set to 1, 2, 3, and 5. The CNN-Transformer-based model shows the lowest Top-K recognition rates, with 20%, 20%, 30%, and 50% for the top-1, top-2, top-3, and top-5. This indicates that the Transformer-based model performs poorly and struggles to correctly classify gestures for sign language in this situation, making it unsuitable for interpreting the complex spatio-temporal relationships inherent in sign language. The intermediate position performance-wise was occupied by the CNN-GRU model, with a moderate 55% top-1 recognition rate, with a top-2, top-3, and top-5 performance of 75%, 85%, and 95%, respectively. Despite having a good temporal dynamic capture from the extracted features, it still lacks in performance compared to the LSTM-based model. In contrast, the CNN-LSTM model achieves an impressive top-1 recognition rate of 75%, a top-2 recognition rate of 90%, and a perfect top-3 and top-5 recognition rate of 100%. Based on its superior performance on our trials, the CNN-LSTM model was selected as the model of choice for the rest of our experiments and analysis.

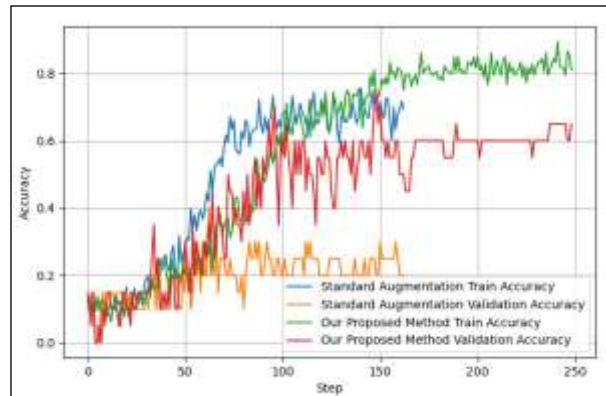


Figure 3: Train and test accuracies of tested methods.
Source: Authors, (2026).

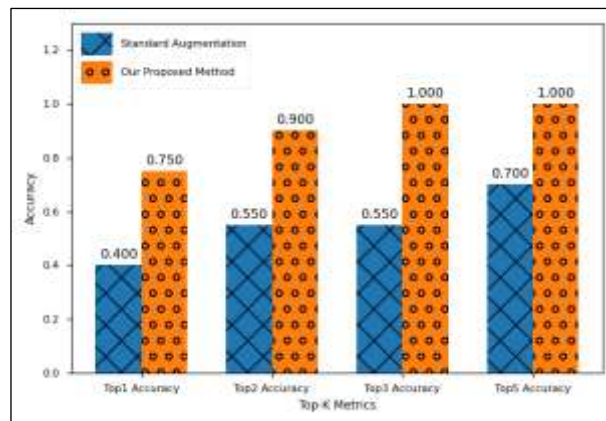


Figure 4: Recognition rates of tested methods.
Source: Authors, (2026).

IV.2 STANDARD DATA AUGMENTATION COMPARISON

The performance comparison between models trained using standard data augmentation and our proposed method was evaluated through both training and validation accuracies, as well as Top-K recognition rates. Figure 3 presents the training and validation accuracies of the models trained on the standard data augmentation method and our proposed method. The standard augmentation method exhibited a peak training accuracy of 75.56%, yet the validation accuracy remained significantly lower at 40%. This disparity indicates that the model experienced overfitting, memorizing the training data rather than effectively learning to generalize patterns. In contrast, our proposed method demonstrated superior performance with a training accuracy of 89.44% and a validation accuracy of 75%. These results highlight the effectiveness of our advanced augmentation technique in reducing overfitting and enhancing generalization capabilities.

IV.3 REAL-WORLD PERFORMANCE ASSESSMENT

For evaluating our model's real-world performance, a 10-fold cross-validation training was employed to ensure a reliable estimate of real-world performance. Figure 5 presents the key metrics resulting from each fold. The figure reports the Top-K recognition rates resulting from the validation set of each fold (Top-1, Top-2, Top-3, and Top-5). The CNN-LSTM architecture exhibited a balanced performance across the folds, with an average Top-1 recognition rate of 70% and consistently high Top-K recognition rates. The cross-validation results revealed a slight variability in performance across folds, with the majority of folds having a top-1 recognition rate of 75% or more.

The best recognition rates achieved were observed in fold 3 reaching a remarkable Top-1 recognition rate of 80%, with all the other recognition rates (Top-2, Top3, and Top-5) at 100%, whereas fold 9 showed the lowest recognition rate with a Top-1 recognition rate of 55% and a Top-2, Top-3, and Top-5 recognition rates of 70%, 85%, and 100% respectively.

V. DISCUSSION

V.1 MODEL SELECTION

The CNN-Transformer exhibited relatively poor performance across all metrics. Despite indicating some pattern recognition capability, the Transformer's architecture appeared less suited for capturing the complex spatial-temporal relationships inherent in sign language gestures. The minimal improvement beyond Top-1 accuracy highlights its struggle to effectively rank true gestures beyond the most probable prediction. This underperformance suggests that while Transformers are effective in certain domains, they may face challenges in handling sequential data with dynamic temporal dependencies, particularly on relatively small datasets. In contrast, the CNN-GRU model occupied an intermediate position between the LSTM and Transformer architectures. It achieved a moderate accuracy of 55% with incremental improvements in Top-K accuracies (Top-2: 0.75; Top-3: 0.85). This performance reflects a competent but less refined capability compared to LSTM, potentially due to GRU's simpler architecture and reduced capacity for capturing temporal nuances. While GRUs are computationally efficient and capable of handling sequential data, their effectiveness in sign language recognition tasks appears inferior to that of LSTMs, especially when dealing with limited amounts of data.

The CNN-LSTM model demonstrated superior performance with the highest recognition rates (Table 1). Its ability to achieve perfect Top-3 and Top-5 scores further underscores its robust capacity to rank true labels effectively within the top predictions. This strong performance can be attributed to the LSTM's inherent capability to capture temporal dependencies, which is critical for processing sequential gesture data in sign language recognition. The integration of VGG16's convolutional capabilities with the sequence memory provided by LSTM proved highly effective in extracting and interpreting spatiotemporal features. The CNN-LSTM model excels due to its ability to capture temporal dependencies, crucial for sign language, which involves sequences of gestures, and specifically so when trained on smaller datasets, as shown in [21]. Our findings also align with [22], where LSTM models are found to outperform other architectures such as Transformers and GRUs on moderately sized datasets.

V.2 STANDARD DATA AUGMENTATION COMPARISON

The findings reveal the effectiveness of our proposed data augmentation method in enhancing model performance. The reduction in overfitting, as evidenced by the improved validation accuracies, suggests that our method introduced more diversity into the training data, allowing the model to learn a broader range of patterns. This is crucial for generalizing well to unseen data. The Top-K recognition rates further underscore the benefits of our approach. The superior performance across all top-K metrics indicates that our method results in a model that not only excels at identifying the most probable gesture but also considers additional possibilities effectively. This broadened perspective is vital for reliable sign language recognition systems.

V.3 REAL-WORLD PERFORMANCE ASSESSMENT

The cross-validation results revealed notable performance from the CNN-LSTM model on the sign language recognition task, as evidenced by a consistent average validation top-1 recognition rate of 70% across 10-fold cross-validation results. The model's high Top-K accuracies further underscore its reliability in handling nuanced classifications. While minor fluctuations in performance were observed in specific folds, such variability likely stems from data distribution imbalances rather than inherent model weaknesses. The integration of VGG16's convolutional capabilities with LSTM's temporal processing strengths positions the CNN-LSTM architecture as particularly effective for sign language recognition, aligning with existing research on classification and sequence tasks [23],[24]. This robust performance across diverse data splits highlights its promising potential for real-world applications in accurately interpreting sign language gestures and offers a solid foundation for future enhancements to further improve reliability.

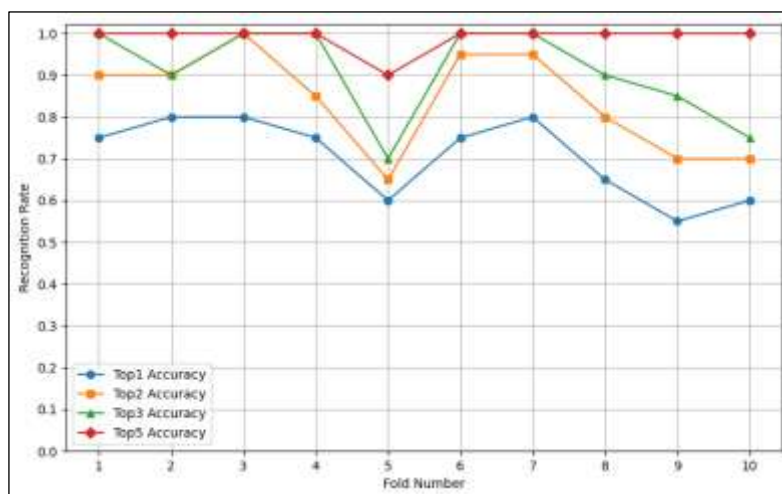


Figure 5: Recognition rates from 10-fold training.
Source: Authors, (2026).

VI. CONCLUSION

In this paper, we introduced a novel signer-agnostic method for generating synthetic sign language datasets, supporting both isolated and continuous sign language generation. This approach enables the creation of diverse signer profiles and environmental variations, providing data suitable for training modern recognition architectures. Our methodology addresses a critical research gap in sign language recognition research, particularly benefiting low-resource languages where data scarcity is a significant challenge. Our proposed data augmentation method has proven effective in developing a more robust and reliable sign language recognition model when compared to standard data augmentation techniques.

The enhanced performance metrics suggest promising practical applications in real-world scenarios requiring accurate and generalized gesture recognition. The 10-fold cross-validation reached a high average Top-1 recognition rate and a good Consistency across folds, highlighting the model's reliability for real-world scenarios. Future research will enhance our methodology by incorporating multi-modal outputs, including depth maps and motion capture data. These additions aim to provide comprehensive training for recognition models through diverse input channels, advancing sign language technology further.

VII. AUTHOR'S CONTRIBUTION

Conceptualization: Younes Ouargani, Noussaima El Khattabi.

Methodology: Younes Ouargani.

Investigation: Younes Ouargani.

Discussion of results: Younes Ouargani, Noussaima El Khattabi.

Writing – Original Draft: Younes Ouargani.

Writing – Review and Editing: Younes Ouargani.

Resources: Younes Ouargani.

Supervision: Noussaima El Khattabi.

Approval of the final text: Younes Ouargani, Noussaima El Khattabi.

VIII. REFERENCES

- [1] "World report on hearing." Accessed: Jun. 02, 2025. [Online]. Available: <https://www.who.int/publications/i/item/9789240020481>
- [2] H. R. V. Joze and O. Koller, "MS-ASL: A Large-Scale Data Set and Benchmark for Understanding American Sign Language," Nov. 20, 2019, arXiv:1812.01053. doi: 10.48550/arXiv.1812.01053.
- [3] J. Forster et al., "RWTH-PHOENIX-weather: A large vocabulary sign language recognition and translation corpus.," in LREC, 2012, pp. 3785–3789. Accessed: Feb. 04, 2026. [Online]. Available: <http://www-i6.informatik.rwth-aachen.de/publications/download/773/forster-lrec-2012.pdf>
- [4] J. Forster, C. Schmidt, O. Koller, M. Bellgardt, and H. Ney, "Extensions of the Sign Language Recognition and Translation Corpus RWTH-PHOENIX-Weather.," in LREC, 2014, pp. 1911–1916.
- [5] A. Duarte et al., "How2Sign: A Large-Scale Multimodal Dataset for Continuous American Sign Language," in 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA: IEEE, 2021, pp. 2735–2744. doi: 10.1109/CVPR46437.2021.00276.
- [6] O. M. Sincan and H. Y. Keles, "AUTSL: A Large Scale Multi-modal Turkish Sign Language Dataset and Baseline Methods," IEEE Access, vol. 8, pp. 181340–181355, 2020, doi: 10.1109/ACCESS.2020.3028072.
- [7] D. Schneider et al., "SynthAct: Towards Generalizable Human Action Recognition based on Synthetic Data," in 2024 IEEE International Conference on Robotics and Automation (ICRA), May 2024, pp. 13038–13045. doi: 10.1109/ICRA57147.2024.10611486.
- [8] H. Hwang, C. Jang, G. Park, J. Cho, and I.-J. Kim, "Eldersim: A synthetic data generation platform for human action recognition in eldercare applications," IEEE Access, vol. 11, pp. 9279–9294, 2021, Accessed: Feb. 04, 2026. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9324837/>
- [9] S. Juraev, A. Ghimire, J. Alikhanov, V. Kakani, and H. Kim, "Exploring human pose estimation and the usage of synthetic data for elderly fall detection in real-world surveillance," IEEE Access, vol. 10, pp. 94249–94261, 2022, Accessed: Feb. 04, 2026. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9870817/>
- [10] C. Neves, L. Coheur, and H. Nicolau, "HamNoSys2SiGML: Translating HamNoSys Into SiGML," in Proceedings of the 12th Language Resources and Evaluation Conference, Marseille, France: European Language Resources Association, May 2020, pp. 6035–6039. Accessed: Mar. 31, 2021. [Online]. Available: <https://www.aclweb.org/anthology/2020.lrec-1.739>
- [11] "CWA Signing Avatars - Virtual Humans." Accessed: Jun. 05, 2025. [Online]. Available: https://vh.cmp.uea.ac.uk/index.php/CWA_Signing_Avatars
- [12] V. Jennings, R. Elliott, R. Kennaway, and J. Glauert, "Requirements For A Signing Avatar," p. 4.
- [13] Y. LeCun et al., "Backpropagation applied to handwritten zip code recognition," Neural computation, vol. 1, no. 4, pp. 541–551, 1989, Accessed: Feb. 04, 2026. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/6795724/>
- [14] A. Vaswani et al., "Attention is All you Need," in Advances in Neural Information Processing Systems, Curran Associates, Inc., 2017. Accessed: Aug. 19, 2023. [Online]. Available: <https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html>
- [15] K. Cho et al., "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," Sep. 02, 2014, arXiv: arXiv:1406.1078. Accessed: Aug. 19, 2023. [Online]. Available: <http://arxiv.org/abs/1406.1078>
- [16] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735. [17] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," arXiv.org. Accessed: Nov. 08, 2023. [Online]. Available: <https://arxiv.org/abs/1412.6980v9>

- [17] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” arXiv.org. Accessed: Nov. 08, 2023. [Online]. Available: <https://arxiv.org/abs/1412.6980v9>
- [18] Y. Ouargani and N. E. Khattabi, “Transformer-Based Optimization for Text-to-Gloss in Low-Resource Neural Machine Translation,” ITEGAM-JETIA, vol. 11, no. 51, pp. 99–111, Jan. 2025, doi: 10.5935/jetia.v11i51.1423.
- [19] J. Martinez, R. Hossain, J. Romero, and J. J. Little, “A Simple yet Effective Baseline for 3D Human Pose Estimation,” 2017, pp. 2640–2649. Accessed: Jun. 02, 2025. [Online]. Available: https://openaccess.thecvf.com/content_iccv_2017/html/Martinez_A_Simple_yet_ICCV_2017_paper.html
- [20] L. Prechelt, “Early Stopping - But When?,” in Neural Networks: Tricks of the Trade, vol. 1524, G. B. Orr and K.-R. Müller, Eds., in Lecture Notes in Computer Science, vol. 1524, Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 55–69. doi: 10.1007/3-540-49430-8_3.
- [21] C. Aparna and M. Geetha, “CNN and Stacked LSTM Model for Indian Sign Language Recognition,” in Machine Learning and Metaheuristics Algorithms, and Applications, S. M. Thampi, L. Trajkovic, K.-C. Li, S. Das, M. Wozniak, and S. Berretti, Eds., Singapore: Springer, 2020, pp. 126–134. doi: 10.1007/978-981-15-4301-2_10.
- [22] J. Xiao, T. Deng, and S. Bi, “Comparative Analysis of LSTM, GRU, and Transformer Models for Stock Price Prediction,” Oct. 20, 2024, arXiv: arXiv:2411.05790. doi: 10.48550/arXiv.2411.05790.
- [23] R. Nichani, L. Gasmi, S. Kabou, and N. Laiche, “Novel Insights on the Comparative Study Between LSTM and Transformer Models for Financial Time Series Prediction,” ITEGAM-JETIA, vol. 11, no. 55, pp. 212–221, Oct. 2025, doi: 10.5935/jetia.v11i55.2658.
- [24] I. D. Suliman and D. Vasumathi, “Predicting Cardiovascular Diseases through Integrated Deep Learning Features,” ITEGAM-JETIA, vol. 12, no. 57, pp. 69–76, Jan. 2026, doi: 10.5935/jetia.v12i57.2740.