

Supplementary Material

Sign Language Recognition: A Large-scale Multi-view Dataset and Comprehensive Evaluation

Anonymous WACV Algorithms Track submission

Paper ID 2430

1. Multi-view Framework

We described our multi-view framework in the main manuscript, Section 4.1. To summarize, for each baseline, we initialize three separate encoders for each view (right, front, left) to extract the view-wise visual features. The latent features of the three views are concatenated to effectively integrate multi-view information and passed through a Multi-layer Perceptron network for sign language classification. Through this simple design, the model is capable of learning visual features from different views and combining them to predict the gloss logits. Figure 1 illustrates our multi-view SLR framework.

2. Experiment Reproducibility

In this section, we describe the experimental setup, including the construction of our 3-view architecture for sign language recognition, hyper-parameter configurations, pre-training strategies, and other relevant details. This section provides a comprehensive overview of the implementation.

2.1. Environment

We train all of the models using a system containing 4 GeForce RTX 3080 GPUs, 125 GB of RAM and 48 Intel Xeon Silver 4214 CPUs with a frequency of 3.0GHz. The Deep Learning sign language recognition models are implemented and trained with Python 3.7.11, Torch version 2.0.0, and TorchVision version 0.15.0. Video reading and processing are handled with Pillow version 9.0.1, and OpenCV version 4.10.0. We utilize mmPose version 1.3.1 and mmCV version 2.1.0 to extract pose key points for the VTNPF [1] baseline.

2.2. Hyper-parameters

For each base model and its corresponding single-view and three-view, Adam serves as the main optimizer. Batch sizes are adjusted between 2 and 8 based on available VRAM. The maximum number of training epochs to 100

(total_epoch). However, the training process typically stops before reaching this limit due to the use of early stopping. Specifically, we apply early stopping with a patience factor of 15 epochs and a delta of 0, comparing improvement on the validation accuracy for each epoch. The training process is halted if no improvement is observed after 15 consecutive epochs.

We employ Grid Search to determine the optimal hyperparameters for both single-view and three-view configurations. For instance, in the case of I3D [1] 1-view, the learning rate was set to 1×10^{-4} with a decay factor of 0.8 applied every 10 epochs, and the weight decay for the three-view variant was increased to 1×10^{-4} . Similarly, for MVIT [2], the learning rate was set to 1×10^{-4} , decreasing by 0.5 every 5 epochs for 1-view, with the three-view configuration having a longer step size of 10 epochs. Detailed hyper-parameters for all models, including three-view variants, are summarized in Table 1.

2.3. Pre-training

Regarding weight initialization, before training the single-view models, we pre-train the model on the large and diverse AUTSL [4] sign language dataset. This pre-training step is essential as it allows the model to familiarize itself with a wide range of sign language gestures, ensuring the learning of both low-level and high-level features. Low-level features capture basic hand movements and orientations, while high-level features represent more complex patterns and contextual nuances.

After pre-training on AUTSL, the checkpoint is then used to fine-tune the model on the front-view of our Multi-VSL dataset. Subsequently, the weights from the single-view Multi-VSL model are reused to fine-tune for training on the three-view recognition task.

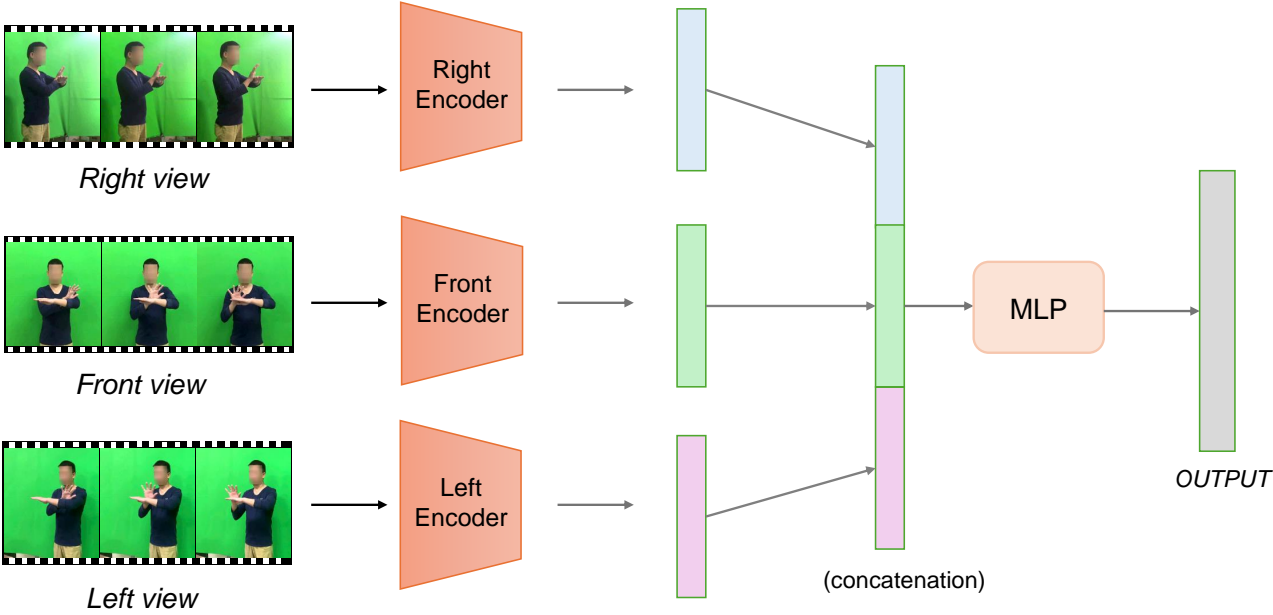


Figure 1. **Our Multi-view Sign Language Recognition Framework.** Each view data is handled by a separate encoder, and the latent features of three-views are combined through vector concatenation and passed through a Multilayer Perceptron for prediction.

Table 1. **Learning rates, gamma values, step size, and weight decay settings for different models.** The table shows the configurations used for both single-view and three-view variants.

Model	Learning rate		Gamma		Step size		Weight decay	
	1 view	3 view	1 view	3 view	1 view	3 view	1 view	3 view
I3D [7]	1×10^{-4}	1×10^{-5}	0.8	0.8	10	10	1×10^{-4}	1×10^{-4}
MViT [2]	1×10^{-4}	1×10^{-5}	0.5	0.7	5	10	1×10^{-4}	1×10^{-4}
Swin. [3]	1×10^{-4}	5×10^{-5}	0.5	0.5	5	5	2×10^{-2}	2×10^{-2}
VTNPF [1]	1×10^{-4}	1×10^{-5}	0.8	0.8	10	10	1×10^{-4}	1×10^{-3}

3. Gloss annotation tool

3.1. Purpose of the Annotation Tool

Our annotation tool streamlines the labeling of video data for various applications, such as machine learning and content analysis. It divides videos into segments, labeling each with start and end times, a corresponding word (represented by a unique ID), and an order action to account for different visual representations. The tool’s flexibility allows it to handle words appearing in multiple contexts and is applicable across various domains requiring video annotation.

3.2. Key Features and Functions

The annotation tool offers a range of features designed to support video data labeling efficiently:

- **Labeling Functionality:** Users can label video segments with start-time, end-time, the ID of a mapped word, and an order action, which enables the representation of different ways to express the same word.

- **Data Types Supported:** The tool supports video data exclusively.
- **Collaborative Work:** The tool is designed for collaboration, allowing multiple users to label video segments simultaneously in real time.
- **CSV Integration:** Users can upload data from CSV files, which allows for easier bulk editing and importing of labeling data. Users can also specify the start time of labeling in seconds and set the start index from which labeling begins.
- **Data Export:** Labeled data can be downloaded for further use, such as in machine learning models or other analytical purposes.

In addition to these features, the tool is highly customizable, providing flexibility in the labeling process. The tool is capable of handling large datasets, making it suitable for video projects that require detailed annotations across multiple segments.

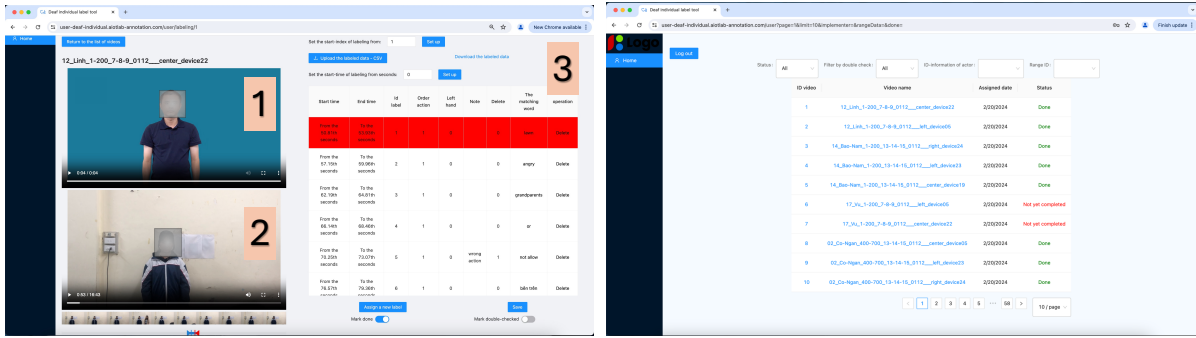


Figure 2. **The graphical user interface (GUI) of our annotation tool**, where left figure is the annotation interface and right figure provides interface for the list of videos with it corresponding status. In the left figure, **1** is the reference video provided for annotators, offering an example to guide the annotation process. **2** is the dataset video, from which annotators identify and mark the timestamps corresponding to each gloss ID based on the reference video. **3** is the annotation table, which records the labeled timestamps and associated gloss information for each video.

3.3. User Interaction

The tool operates through a highly usable graphical user interface (GUI), which is accessible via any modern web browser. This design allows users to connect to the system from anywhere, ensuring low latency and high usability, even in collaborative environments. Moreover, our interface makes the labeling process easier, as it gives a reference video for each gloss, and an annotation table for double check (see Figure 2).

The system is optimized for team collaboration, enabling multiple users to work together in real-time. Each user can see updates made by others, which enhances the efficiency of the labeling process, especially when dealing with large datasets.

3.4. Current Solution and Deployment

The annotation tool is currently deployed on our custom server infrastructure using MinIO for data storage. This deployment is designed to be cost-effective, eliminating the need to rely on external cloud services like AWS S3. By hosting MinIO ourselves, we avoid the increasing costs associated with scaling up data storage in the cloud, especially for large datasets that can reach terabytes in size.

Cost-Saving Example: For instance, using AWS S3 to store 1TB of video data could cost hundreds of dollars per month, depending on data transfer and retrieval frequency. By contrast, deploying MinIO on a local server incurs only the initial server and maintenance costs, significantly reducing ongoing expenses. This approach allows us to scale storage to meet future data needs without incurring high cloud service costs.

4. Dataset samples

This section provides some examples from our dataset to demonstrate that our Multi-VSL dataset was built in an

door environment, with a diverse range of people, glosses, backgrounds, and, especially, recorded from three views. The images are cropped by detecting human poses with YoloV9 [6] to mitigate noisy background signals. Human faces are detected by a YoloV9 model [5] and blurred with Gaussian filters to preserve the signers' privacy. (See Figures 3, 4, 5, 6)

References

- [1] Mathieu De Coster, Mieke Van Herreweghe, and Joni Dambre. Isolated sign recognition from rgb video using pose flow and self-attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3441–3450, 2021. 1, 2
- [2] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Kartikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Mvitv2: Improved multiscale vision transformers for classification and detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4804–4814, 2022. 1, 2
- [3] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3202–3211, 2022. 2
- [4] Ozge Mercanoglu Sincan and Hacer Yalim Keles. Autsl: A large scale multi-modal turkish sign language dataset and baseline methods. *IEEE access*, 8:181340–181355, 2020. 1
- [5] spacewalk01. YoloV9 face detection. <https://github.com/spacewalk01/yolov9-face-detection>, 2023. Accessed: 2024-08-28. 3
- [6] Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. YoloV9: Learning what you want to learn using programmable gradient information. *arXiv preprint arXiv:2402.13616*, 2024. 3
- [7] Xianyuan Wang, Zhenjiang Miao, Ruyi Zhang, and Shanshan Hao. I3d-lstm: A new model for human action recognition. In *IOP conference series: materials science and engineering*, volume 569, page 032035. IOP Publishing, 2019. 2



signer #3
gloss #1 (grass)



signer #20
gloss #3 (grandparent)

Figure 3. Multi-view Sample of Gloss #1 and #3, with their corresponding English meanings



signer #5
gloss #134 (chance)



signer #28
gloss #183 (unfamiliar)

Figure 4. Multi-view Sample of Gloss #134 and #183, with their corresponding English meanings



signer #18
gloss #199 (brush one's teeth)



signer #10
gloss #359 (pass the ball)

Figure 5. Multi-view Sample of Gloss #199 and #359, with their corresponding English meanings



signer #12
gloss #519 (yam)



signer #27
gloss #700 (mane of hair)

Figure 6. Multi-view Sample of Gloss #519 and #700, with their corresponding English meanings