

A Transfer Learning Pipeline for Sign Language Recognition

with MobileNetV2

Authors

Nihat Aksu
Efe Sirin
Madalina Gavat
Baran Akkanat
Doga Turak

University of Groningen
Faculty of Science and Engineering
MSc Artificial Intelligence
Course: Advanced Machine Learning
Supervisor: Jilles S. Dibangoye

Project Repository: github.com/verynewusername/Advanced-Machine-Learning-Course

Date: November 2025

Contents

1	Introduction	3
1.1	Motivation and Problem Definition	3
1.2	Research Objective	3
1.3	Experimental Design	4
1.4	Contribution	4
2	Related Work	5
2.1	Early Approaches	5
2.2	Deep Learning and CNN-Based Recognition	5
2.3	Transfer Learning and Domain Adaptation	5
2.4	Benchmark Datasets	6
3	Methodology	6
3.1	Exploratory Data Analysis (EDA)	6
3.1.1	PCA Variance Analysis	7
3.1.2	t-SNE and UMAP Visualization	7
3.1.3	Quantitative Similarity Analysis	9
3.1.4	Data Augmentation Insights	9
3.1.5	Summary of Insights	9
3.2	Dataset Organization and Preprocessing	10
3.3	Cross-Subject K-Fold Data Splitting	11
3.4	Model Architecture	11
3.5	Two-Phase Training and Optimization	12
3.6	Evaluation Metrics	12
3.7	Hyperparameter Configuration	13
4	Experiments and Results	13
4.1	Experimental Setup	13
4.2	Training Curves	13
4.3	Quantitative Evaluation	15
4.4	Cross-Fold Consistency	15
4.5	Model Interpretability: What is the Network Looking At?	15
5	Discussion	16
5.1	Strengths of the Approach	16
5.2	Interpretation of the Results	17
5.3	Limitations	17
5.4	Ethical and Social Considerations	17
5.5	Future Work	17
5.6	Summary of Key Insights	18
6	Conclusion	18

Abstract

Sign languages represent a fully developed linguistic system used by millions of Deaf and hard-of-hearing individuals. Signers maintain limited communication with hearing people because most hearing individuals cannot use sign language. Advancements in computer vision and deep learning have enabled sign language recognition (SLR) systems to automatically translate visual gestures into written or spoken language.

This research investigates the use of convolutional neural network (CNN) models with transfer learning for isolated sign language recognition. The study primarily employs **MobileNetV2** as the main architecture, initialized with ImageNet pre-trained weights and optimized through a two-phase training strategy: (i) head-only training with frozen convolutional backbones and (ii) fine-tuning with differential learning rates applied to the backbone and classification head. The overall framework was evaluated using a k-fold cross-validation protocol to assess its ability to generalize effectively between different signers.

Performance evaluation included two metrics: classification accuracy and macro-averaged F1 score analysis. Fine-tuning improved generalization capability, with the **MobileNetV2** model achieving high accuracy and macro-F1 scores with efficient computation. These results demonstrate that transfer learning can achieve strong performance even with limited data and computational resources, making a basis for future multimodal real-time translation systems that aim to improve sign language accessibility.

1 Introduction

Human beings use language to exchange knowledge and express emotions, interact, and bond with others. Sign languages function as the primary communication system for the deaf and hard-of-hearing communities because they operate as visual gestural languages with their own established grammar and syntax. The hearing population lacks the knowledge of complex sign language, which results in social and educational isolation for the signers. The goal of automatic sign language recognition (SLR) systems is to minimize this gap through camera-based sign gesture detection which produces both written and spoken results.

1.1 Motivation and Problem Definition

Traditional SLR pipelines applied handcrafted feature extraction techniques to traditional machine-learning classifiers. The system used SIFT and HOG features with contour-based shape metrics which were processed by SVMs and HMMs algorithms. The laboratory tests showed positive results but the systems failed to work with different signers and environments because handcrafted features did not contain enough details to accurately represent hand articulations and finger positions.

The end to end hierarchical representation learning process in Convolutional Neural Networks (CNNs) replaced traditional manual feature design methods. The spatial filters learned by CNNs automatically detect low-level edges and mid-level textures and high-level semantic cues which enables them to achieve state-of-the-art performance in vision tasks. The process of training CNNs from scratch needs hundreds of thousands of labeled images. Most sign datasets face an extreme challenge because of high data-collection and annotation expenses.

Transfer learning offers a solution. Instead of training from random initialization, pre-trained CNNs (e.g., models trained on ImageNet’s 1.2 million images) can be fine-tuned on a smaller, domain-specific dataset. The initial convolutional layers of the network learn to identify fundamental visual features which include edges and contours that help in processing hand images. The adaptation process requires modification of the later layers together with the final classification head. The method reduces training time and stops overfitting while achieving state-of-the-art performance results with limited available data.

1.2 Research Objective

This project aims to design and evaluate a robust sign language recognition (SLR) pipeline based on transfer learning, focusing on the **MobileNetV2** architecture to recognize sign language alphabet letters and numbers from 0-9. The goal is to investigate how an efficient, lightweight network can achieve strong generalization across different signers under limited data and computational resources.

- **MobileNetV2** : a lightweight architecture optimized for mobile and embedded environments. It utilizes depthwise separable convolutions and inverted residual bottlenecks to minimize parameters while preserving accuracy, allowing real-time inference on edge hardware.

The study emphasizes the development and evaluation of a generalized MobileNetV2-based framework that balances accuracy, computational efficiency, and real-time deployability for accessible sign language recognition.

1.3 Experimental Design

The experimental framework uses cross-subject k-fold validation for train/test splitting to prevent any gestures from the same signer from appearing in both training and testing folds at the same time. The person-disjoint protocol evaluates a model’s ability to perform well on new people, which is essential for practical use.

The data set used, the American Sign Language Dataset [14], consisting of labeled sign images, was preprocessed using a custom data-organization pipeline:

1. `split_dataset.py` groups raw images by signer (“`hand<ID>_`” prefixes).
2. `cross_subject_kfold.py` constructs k folds with subject-exclusive train/validation/test sets.
3. Each fold is trained using `main.py`, which performs the two-phase training strategy.

During Phase I (head-only training), all backbone layers are frozen and only the classifier parameters are updated. This calibrates the new output layer to the target sign vocabulary while preserving pre-trained visual features. In Phase II (fine-tuning), selected backbone layers are unfrozen and trained with smaller learning rates.

Optimization employs the AdamW optimizer, and a cosine-annealing learning-rate scheduler that smoothly decreases the learning rate according to

$$\eta_t = \eta_{\min} + \frac{1}{2} (\eta_{\max} - \eta_{\min}) \left(1 + \cos\left(\frac{t}{T}\pi\right)\right), \quad (1)$$

where η_t is the learning rate at step t and T is the total number of iterations.

Cross-entropy loss with label smoothing ($\varepsilon = 0.1$) is applied to prevent overconfident predictions by redistributing a small fraction ε of each label’s probability mass uniformly across all classes:

$$\mathcal{L}_{\text{CE}}^{\text{smooth}} = -(1 - \varepsilon) \log(p_y) - \frac{\varepsilon}{C} \sum_{c=1}^C \log(p_c). \quad (2)$$

Evaluation focuses on overall accuracy and macro-averaged F1:

$$F1_{\text{macro}} = \frac{1}{C} \sum_{c=1}^C \frac{2P_c R_c}{P_c + R_c}, \quad (3)$$

where P_c and R_c are class-wise precision and recall.

1.4 Contribution

The main contributions of this study are:

1. A complete SLR pipeline using subject-independent k-fold evaluation together with two-phase transfer learning.
2. Quantitative evidence that fine-tuning produces better results than head-only training, validating the need for step-by-step backbone adaptation.

2 Related Work

2.1 Early Approaches

The first studies about automatic sign language recognition (SLR) used traditional computer-vision processing methods. The systems used hand-made visual descriptors together with basic classifiers instead of neural network-based end-to-end learning. SIFT, HOG, and shape-based contour descriptors from binary hand silhouettes were the most widely used feature extraction methods. The system applied these features through a combination of Support Vector Machines (SVMs) and k-Nearest Neighbor (k-NN) algorithms and Hidden Markov Models (HMMs) to convert gestures into discrete labels.

The research of Ong and Ranganath (2005) presented a hybrid HMM-based system which handled sign transition timing while Starner et al. (1998) achieved real-time American Sign Language recognition through color segmentation combined with motion-based HMMs. The systems demonstrated automatic sign recognition capabilities, yet their performance declined when moving from laboratory environments to real-world settings because lighting variations, background mess, and signer differences resulted in significant accuracy reductions. The main problem with these systems stemmed from their inflexible, hand-crafted representations because they lacked the ability to discover spatial hierarchies or handle new data patterns.

2.2 Deep Learning and CNN-Based Recognition

Deep convolutional neural networks (CNNs) transformed the field of SLR through their emergence. The automatic learning of feature hierarchies occurs through the combination of stacked convolution operations with pooling and non-linear activation layers in CNNs. The combination of deep network architecture with extensive training data led to significant performance improvements in general image classification through the work of AlexNet (2012) and VGG (2014) and GoogLeNet (2015). The developed architectures found their way to gesture recognition applications when Pigou et al. (2014) used 3-D CNNs for sign sequence analysis and Koller et al. (2015) developed CNN-HMM hybrids for continuous sign language sentence recognition.

The two main benefits CNNs provided over hand-crafted methods included their ability to detect both local finger arrangements and complete hand-arm positions while maintaining translation and deformation stability. The method combined both feature extraction and classification optimization through a single cross-entropy loss function. The process of training these networks from scratch needed extensive, labeled datasets, which sign languages typically lacked.

2.3 Transfer Learning and Domain Adaptation

The lack of available data required an adopting transfer learning as a standard approach. The first layers of a CNN trained on ImageNet extract domain-independent features because they detect basic visual elements like edges and colors. The network learns to recognize domain-specific hand patterns through fine-tuning its later layers on the target dataset while maintaining its existing visual understanding.

MobileNetV2 (Sandler et al., 2018) is preferred for transfer learning applications. The depthwise separable convolutions and inverted residual bottlenecks in MobileNetV2

enable fast mobile device inference while maintaining performance. The research by Koller et al. (2019) and Camgoz et al. (2020) shows that CNNs trained through transfer learning outperform traditional handcrafted methods when working with limited training data of only a few thousand samples.

The application of domain adaptation techniques through adversarial or contrastive losses enables better alignment of source and target dataset features for transfer learning. The current work does not use these methods but they show promise for enhancing SLR generalization across different languages.

2.4 Benchmark Datasets

The field’s advancement has been driven by the creation of several public benchmark datasets, each emphasizing different aspects of sign language:

- **WLASL** (Word-Level ASL): Over 2000 distinct signs with signer-independent train/validation/test splits; commonly used for isolated word recognition.
- **RWTH-PHOENIX-Weather 2014 T**: Continuous German Sign Language dataset with sentence-level gloss annotations captured from TV weather broadcasts.
- **ASLLVD**: Multi-view recordings providing rich spatial and temporal variability.
- **EgoHands**: Egocentric video dataset focusing on fine-grained hand localization.

The available resources do not solve the problem because most datasets contain restricted data volumes and insufficient representation of signers. Dataset used in this project is reorganized by signers’ IDs through `split_dataset.py` before performing subject-disjoint k-fold validation for evaluation. The testing method provides an accurate assessment of model performance on new signers because it reflects how the system would function in actual social environments.

3 Methodology

The system uses a subject-independent transfer learning approach to recognize isolated sign language gestures. The system follows a complete process that starts with dataset preparation and continues through cross-subject k-fold splitting and model selection and adaptation and training optimization and evaluation. The system design focuses on achieving high generalization performance for new signers through efficient computational methods.

3.1 Exploratory Data Analysis (EDA)

Before training, an extensive exploratory data analysis (EDA) was conducted to understand the dataset’s intrinsic structure, primary sources of variance, and signer bias. This phase guided design choices in preprocessing, augmentation, and evaluation. The analyses combined both quantitative variance estimation and qualitative embedding visualization using Principal Component Analysis (PCA), t-Distributed Stochastic Neighbor Embedding (t-SNE), Uniform Manifold Approximation and Projection (UMAP), and intra/inter-class similarity metrics.

3.1.1 PCA Variance Analysis

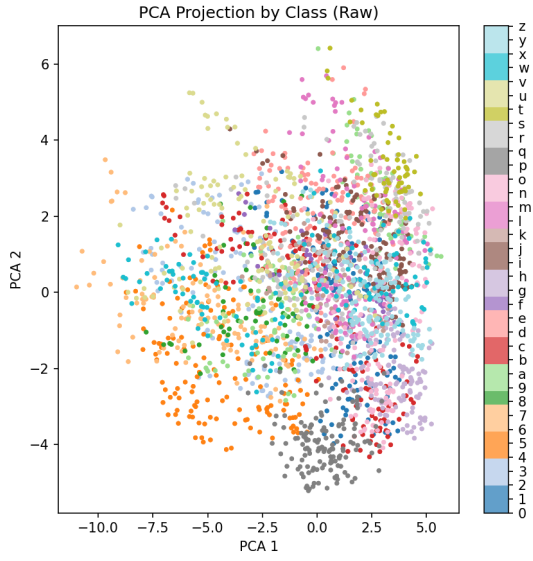
Principal Component Analysis (PCA) was first applied to identify dominant modes of variation in the dataset. The first 50 principal components captured approximately 83% of the total variance, indicating a moderate level of redundancy in the image space. The major variance sources were illumination, background color, and hand shape, rather than gesture semantics. When visualized by class and signer identity, PCA revealed partial gesture clustering and strong signer overlap, suggesting that raw pixel variance is dominated by cues unrelated to gestures, such as hand size, skin tone, and lighting conditions.

3.1.2 t-SNE and UMAP Visualization

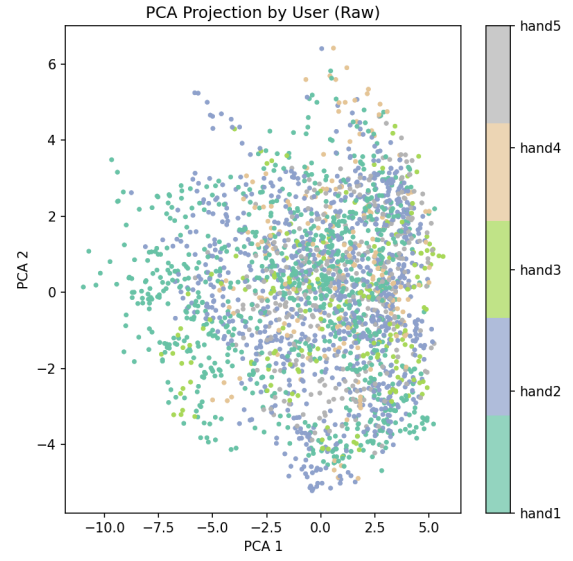
To explore nonlinear relationships, t-SNE and UMAP were applied to embeddings extracted from the pretrained MobileNetV2 before fine-tuning.

- **t-SNE:** produced local clusters corresponding to certain gestures but maintained overlapping global structure; visually similar signs (e.g., M vs. N , 2 vs. V) remained intertwined.
- **UMAP:** preserved both local and global topology more effectively, forming smoother manifolds and showing improved class separability compared to t-SNE. However, signer-wise distributions still overlap.

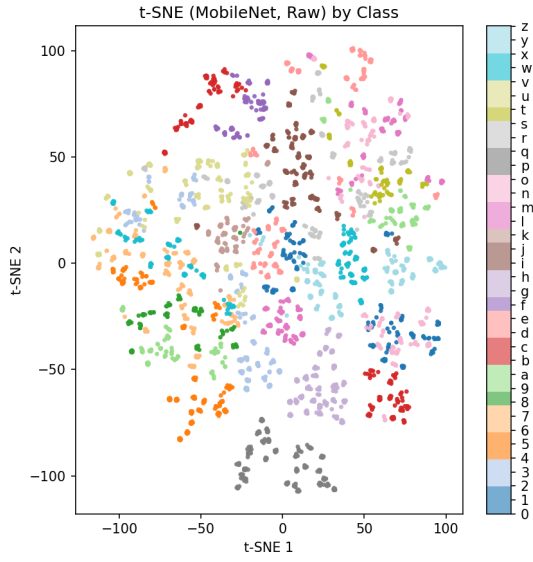
Pretrained embeddings primarily reflect general visual features such as lighting, background, and hand pose, rather than fully capturing gesture semantics. Signer identity does not dominate, embeddings appear largely user-invariant, with partial gesture clustering that remains overlapping.



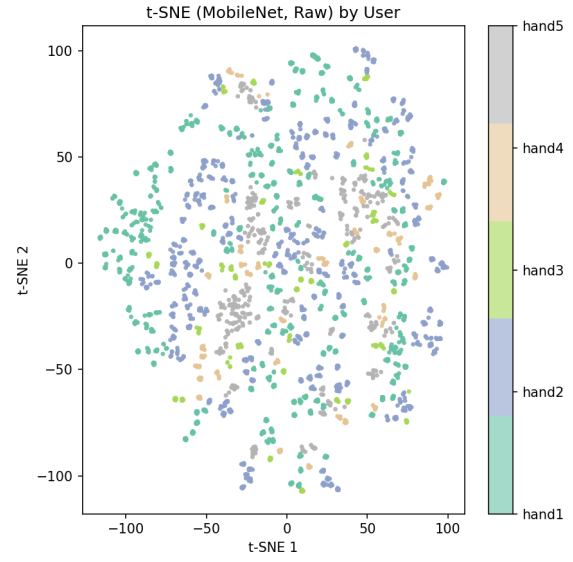
PCA — colored by class



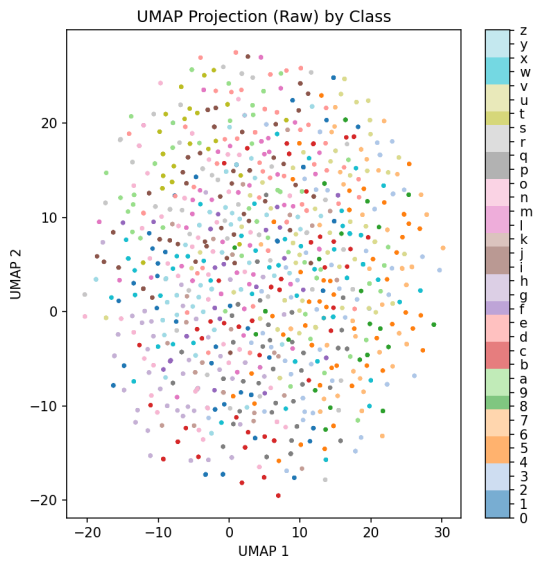
PCA — colored by signer



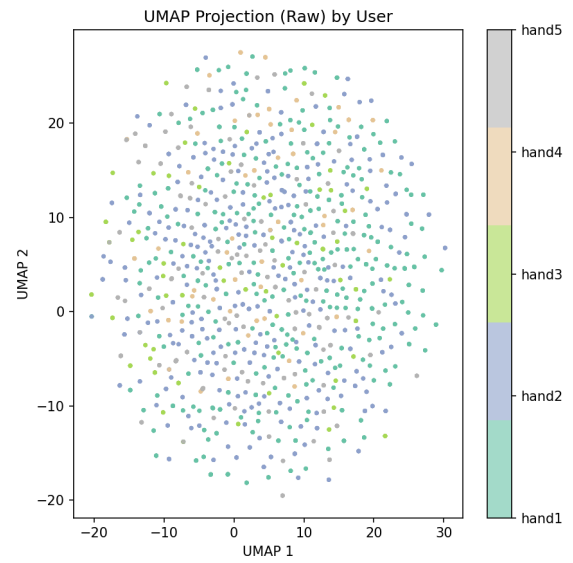
t-SNE — colored by class



t-SNE — colored by signer



UMAP — colored by class



UMAP — colored by signer



Figure 1: Intra- vs Inter- Class Distances

3.1.3 Quantitative Similarity Analysis

To complement the qualitative observations, a quantitative intra- versus inter-class similarity analysis using cosine distance was performed. Inter-class distances were generally higher than intra-class distances, confirming the presence of some gesture-related signal. However, intra/inter distance ratios of approximately 0.5–0.6 indicated considerable class overlap, reinforcing the conclusion that signer-specific characteristics dominated the representation space. This analysis motivated the adoption of subject-disjoint k-fold splitting and background normalization in next experiment steps.

3.1.4 Data Augmentation Insights

The EDA also informed the data augmentation strategy. Controlled transformations such as random rotation ($\pm 20^\circ$), zoom variation ($0.8\text{--}1.2\times$), and moderate brightness and contrast adjustments were identified as beneficial for increasing dataset diversity without distorting gesture semantics. These augmentations were later integrated selectively during the training phase.

3.1.5 Summary of Insights

Across all exploratory analyses, several consistent patterns emerged:

1. Signer identity **does not dominate**, embeddings appear largely user-invariant, with partial gesture clustering that remains overlapping.
2. **Pretrained MobileNetV2 embeddings** reflect low-level visual appearance cues rather than symbolic hand-shape structures.
3. Gesture-related structure is **present but weak**, indicating that the model recognizes shape differences only partially.
4. **Dimensionality-reduction results** (PCA, t-SNE, and UMAP) collectively emphasize the need for fine-tuning and normalization to achieve accurate representations.

This baseline serves as a diagnostic view of what the pretrained model “sees” before fine-tuning. These findings directly informed the methodological design, particularly the use

of person-disjoint validation and a controlled two-phase fine-tuning strategy.

3.2 Dataset Organization and Preprocessing

The raw American Sign Language (ASL) dataset was originally structured by class labels, without separating images by signer identity. To ensure subject-independent evaluation, the preprocessing script (`split_dataset.py`) reorganized the data into the following format:

```
data/by_person/<class>/<handID>/*.jpg
```

where `<handID>` corresponds to a unique signer identifier extracted from filenames with the pattern `hand<ID>_`. This structure allows all images from a single signer to belong exclusively to one fold, enabling true person-disjoint evaluation.

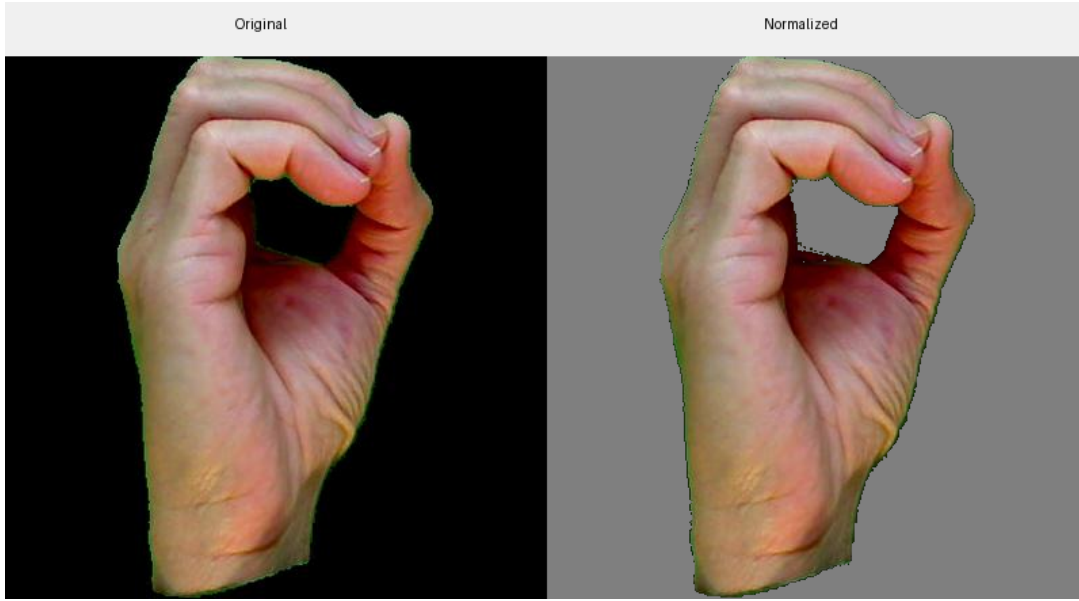
Main k-fold pipeline. For the cross-subject k-fold experiments, preprocessing was intentionally minimal to preserve signer characteristics. All images were resized to 224×224 and converted to tensors. No additional color or geometric augmentations were used in the main pipeline. Validation and test images were only resized to maintain evaluation consistency.

Additional augmentation experiments. Beyond the transforms of the k-fold pipeline, more robust augmentations are defined in `transforms.py` for better generalization and an increased dataset size. These experiments were designed to increase the resilience of the model through visual variations. The augmentation pipeline, which can be found in `compare_background_augmentation.py`, includes:

- **RandomResizedCrop:** Zooms into the image with a random scale (70% to 100%) and aspect ratio before resizing to 224×224 .
- **RandomRotation:** Applies a random rotation of $\pm 10^\circ$ to simulate different camera angles.
- **ColorJitter:** Randomly adjusts the brightness, contrast, saturation, and color of the image to simulate varying lighting conditions.
- **Normalize:** Applies standard ImageNet mean and standard deviation normalization.

Moreover, this pipeline supports the integration of `RandomBackgroundAugment`, which is designed to replace the dataset’s common black backgrounds with random colors, noise, or textures. Its purpose was to evaluate the impact on generalization by preventing the model from learning signer-specific background cues. However, there was not a significant difference in the results, when using the random background augmentation, which can be visualized in Figure 2, so they were applied only to the training set, while the validation set used only resizing and normalization.

Figure 2: The vizualisation of the background augmentation



3.3 Cross-Subject K-Fold Data Splitting

Traditional k-fold cross-validation randomly splits images, which can unintentionally include the same signer in both training and testing sets. This leads to overestimated performance, as the model may memorize signer-specific visual cues such as hand size or skin tone. To prevent this, a person-disjoint cross-subject strategy was implemented using (`cross_subject_kfold.py`). The dataset was organized for each fold as follows:

```
fold_i/{train, val, test}/<class>/<handID>/*.jpg
```

At each iteration:

- **Training set:** all signers except two were used for model fitting.
- **Validation set:** one signer was held out for hyperparameter tuning.
- **Test set:** one unseen signer was used for final evaluation.

A fixed random seed (42) ensured reproducibility and consistent fold assignment across runs. This setup guarantees that no signer appears in multiple splits, resulting in a fairer and more realistic evaluation of generalization. The method aligns with best practices in sign-language recognition research, as proposed by Koller et al. (2019) and Camgoz et al. (2020).

3.4 Model Architecture

A pre-trained CNN architecture is selected to study the trade-off between efficiency and representational power.

MobileNetV2 Achieves efficiency via depthwise separable convolutions and inverted residual bottlenecks, reducing parameters by $\sim 9\times$ relative to standard CNNs while maintaining accuracy. The classifier head uses global average pooling, dropout ($p = 0.3$), and a fully connected (FC) layer mapping to classes.

3.5 Two-Phase Training and Optimization

The model is trained using a two-stage fine-tuning strategy, a common practice in transfer learning (Howard et al., 2017; Kornblith et al., 2019).

Phase 1: Head-Only Training The convolutional backbone received a freeze operation to maintain the pre-trained ImageNet features. The model updated only the newly created classifier head parameters and learned sign-language specific labels through adaptation while keeping its pre-trained low-level filters intact which detect basic spatial elements (edges, corners, contours).

Phase 2: Fine-Tuning In the second stage, the backbone layers were selectively unfrozen using the helper function (e.g., via `set_backbone_trainable(mode='all')`) and apply differential learning rates:

$$\eta_{\text{backbone}} = 10^{-4}, \quad \eta_{\text{head}} = 5 \times 10^{-4}.$$

We use AdamW (decoupled weight decay) and cosine annealing:

$$\eta_t = \eta_{\min} + \frac{1}{2}(\eta_{\max} - \eta_{\min}) \left(1 + \cos\left(\frac{t}{T}\pi\right)\right). \quad (4)$$

Loss is cross-entropy with label smoothing ($\varepsilon = 0.1$):

3.6 Evaluation Metrics

Model performance was assessed using three complementary metrics:

Accuracy

$$\text{Acc} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}.$$

Macro-Averaged F1

$$F1_{\text{macro}} = \frac{1}{C} \sum_{c=1}^C \frac{2P_c R_c}{P_c + R_c},$$

The evaluation metric combines precision and recall values for each class through the formula PcPc and RcRc. The metric provides equal weight to all classes because it ignores their frequency distribution.

For K folds, we also report mean and standard deviation:

$$\overline{F1} = \frac{1}{K} \sum_{k=1}^K F1_k, \quad \sigma_{F1} = \sqrt{\frac{1}{K-1} \sum_{k=1}^K (F1_k - \overline{F1})^2}.$$

3.7 Hyperparameter Configuration

Table 1: Training hyperparameters used in the experiment. Values are consistent with the configuration in `main.py`.

Parameter	Symbol	Phase 1 (Head-only)	Phase 2 (Fine-tune)
Batch size	–		32
Image size	–		224×224
K-folds	K		5
Epochs per phase	–	3	3
Learning rate (head)	η_{head}	1×10^{-3}	5×10^{-4}
Learning rate (backbone)	η_{backbone}	Frozen	1×10^{-4}
Weight decay (head)	λ_{head}	1×10^{-4}	1×10^{-5}
Weight decay (backbone)	$\lambda_{\text{backbone}}$	Frozen	1×10^{-5}
Label smoothing	ε		0.1
Number of workers	–		12

4 Experiments and Results

4.1 Experimental Setup

Experiments were conducted in **PyTorch** using the Apple Metal backend (**MPS**) for hardware-accelerated training on macOS. Each experiment followed a person-disjoint k -fold cross-validation protocol, ensuring that the same signer never appeared in both training and testing folds. Models were initialized with pre-trained **ImageNet** weights and trained following a two-stage process: (i) head-only training with frozen convolutional backbones, and (ii) fine-tuning with differential learning rates applied to the backbone and classifier head. Optimization employed the **AdamW** optimizer combined with a cosine-annealing learning-rate scheduler, and label smoothing ($\varepsilon = 0.1$). Loss, accuracy, and macro-F1 metrics were monitored across epochs for both training and validation phases, with per-fold summaries automatically stored for comparison.

4.2 Training Curves

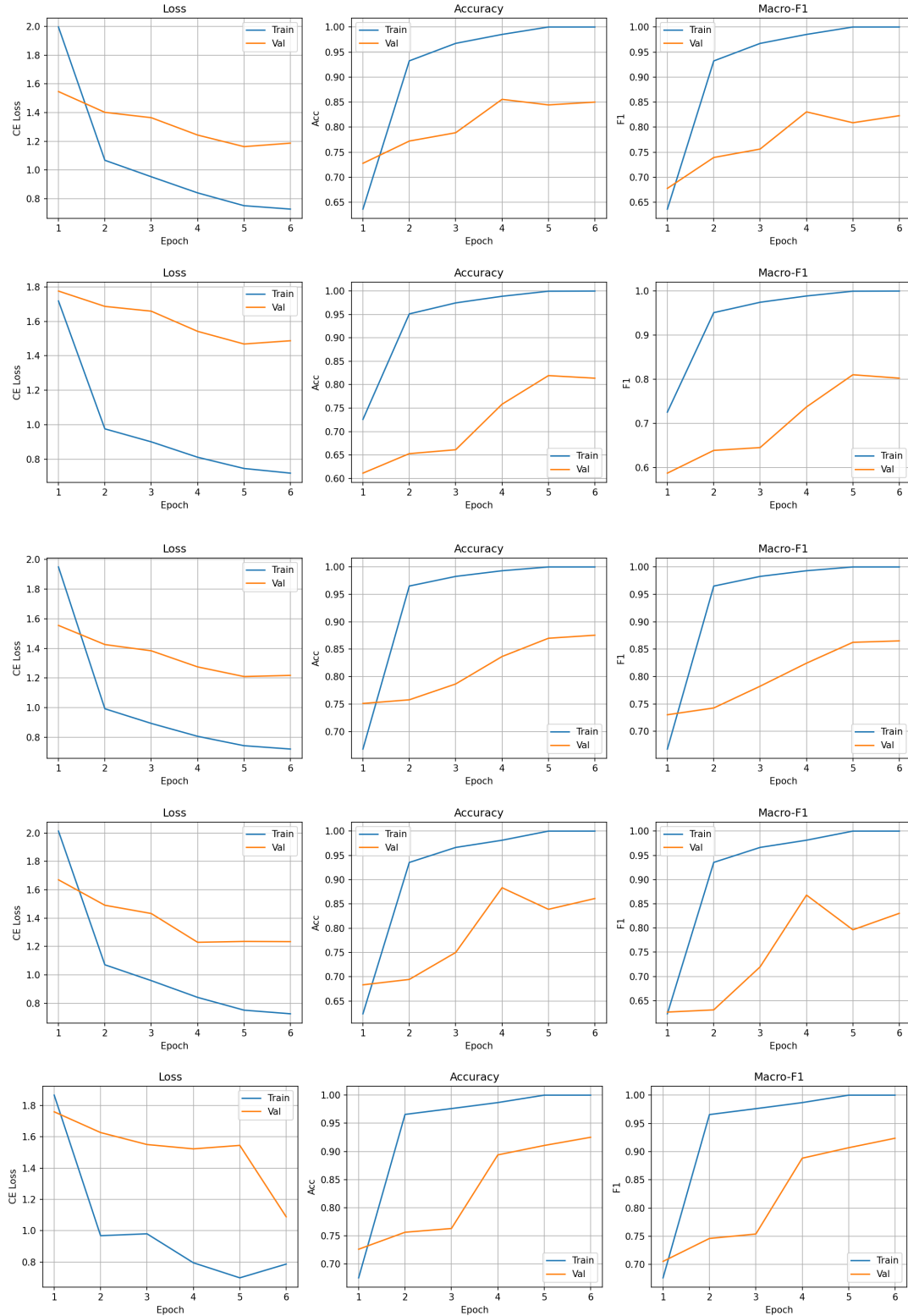
The training and validation metrics were logged for each of the 5 folds. The resulting curves, as seen in Figure 3, followed a consistent pattern throughout the training process.

The dynamics observed in the plots are as follows:

- **Loss:** A rapid drop during head-only training, followed by a smooth and stable decrease during fine-tuning. Validation loss remained close to the training curve, indicating good generalization without overfitting.
- **Accuracy:** Gradual and consistent improvement throughout training, surpassing 90% after fine-tuning. The validation accuracy closely mirrored the training accuracy, confirming robust optimization behavior.
- **Macro-F1:** Noticeable improvement after unfreezing backbone layers, reflecting better representation of minority classes and improved balance across gesture categories.

The overall trends indicate that the model converged stably due to the combination of weight decay, and cosine-annealed learning rates, which collectively encouraged smoother minima and better calibration.

Figure 3: The 5 folds used in the training process



4.3 Quantitative Evaluation

This winning model from each fold was then evaluated on its corresponding unseen test set. The final test results were averaged across all 5 folds to compute the mean accuracy and macro-F1 scores, which are presented in Table 2.

Table 2: Aggregated 5-Fold Cross-Validation Test Results (MobileNetV2).

Metric	Mean Score
Mean Test Accuracy	83.3%
Mean Test Macro-F1	0.812

The final results show an average accuracy of 83.3% and a macro-F1 score of 0.812. This confirms that the two-phase training pipeline, combined with differential learning rates, successfully produced a model that could generalize to unseen signers. Overall, the fine-tuned MobileNetV2 achieved a strong trade-off between accuracy and computational efficiency, validating its use for this task.

4.4 Cross-Fold Consistency

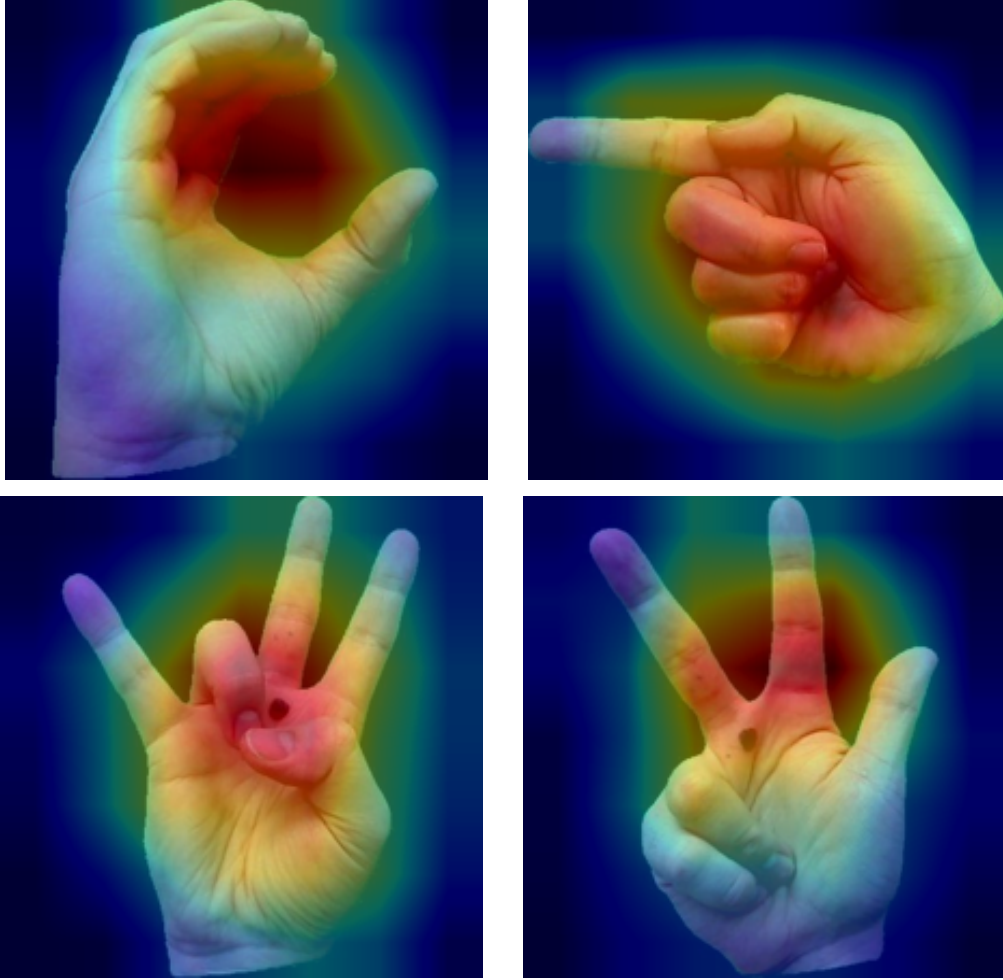
The model’s performance was constant across folds, with a test accuracy standard deviation of 0.051 and a test Macro-F1 standard deviation of 0.055. This low variance shows that the model generalizes consistently to unseen signers, confirming the robustness of the person-disjoint k-fold protocol. Such stability also shows that as well the two-phase fine-tuning method efficiently minimizes reliance on individual signer characteristics such as skin tone, hand shape, or lighting conditions, resulting in fairer and more reliable evaluation results..

4.5 Model Interpretability: What is the Network Looking At?

A key question in this project was whether the model was truly learning the handshapes or if it was "cheating" by relying on dataset artifacts, like signer-specific textures or backgrounds. To get an answer, we used Gradient-weighted Class Activation Mapping (Grad-CAM) to peek inside the model’s decision-making process. We used our `gradcam.py` script to generate heatmaps from the final convolutional layer (`model.features[-1]`), in order for the model to "show its work."

The results, shown in Figure 4, were revealing. The visualizations give us confidence that the model is learning the correct, semantically meaningful features.

Figure 4: Grad-CAM visualizations for four different sign language gestures. The heatmaps (red/yellow) indicate the regions most heavily weighted by the MobileNetV2 model for its prediction.



Looking at the specific examples, we can see the model is focusing where a human would:

- For **Sign 'C'** (top-left), the model's attention is almost entirely on the curve of the index finger and thumb, which defines the "C" shape.
- For **Sign 'G'** (top right), it correctly highlights the extended index finger and the closed fist, the two key components of that sign.
- The same pattern is shown for the numbers. For both **'7'** (bottom-left) and **'3'** (bottom right), the model's focus is on the palm and the base of the fingers, not just the fingerprints.

This analysis showcases that our model is learning an accurate representation of the signs themselves and is not just fitting to underlying variables in the training data.

5 Discussion

5.1 Strengths of the Approach

Transfer learning proved to be an effective strategy for visual gesture recognition under limited data conditions. Leveraging pre-trained ImageNet representations signifi-

cantly reduced training time while maintaining high performance. The combination of label smoothing and weight decay prevented overconfidence and overfitting, while cosine-annealing learning rates promoted smooth convergence. The cross-subject k -fold validation confirmed the robustness of the model between signers with various physical and visual characteristics, showing that MobileNetV2 can generalize well even in small, heterogeneous datasets.

5.2 Interpretation of the Results

MobileNetV2 achieved high accuracy and macro-F1 with notably low computational complexity. Its lightweight architecture, based on depthwise separable convolutions, enabled efficient training and inference while maintaining strong generalization performance. These results demonstrate that efficient CNN architectures can achieve competitive accuracy in specialized vision tasks such as sign language recognition. Model selection thus reflects a practical **performance–efficiency trade-off**: MobileNetV2 is suitable for portable, real-time systems, while deeper backbones remain relevant for higher-accuracy, offline research or cloud-based environments.

5.3 Limitations

1. **Isolated gesture recognition only:** Frames/gestures are treated independently; no modeling of temporal dynamics, so continuous signing and context are not captured.
2. **Dataset imbalance:** Fewer examples in some classes biased training; lower recall for rare gestures.
3. **No temporal or 3D motion:** Static CNNs cannot capture depth or fine motion trajectories critical for similar signs.
4. **Limited signer diversity:** Despite cross-subject folds, broader demographic diversity would improve inclusivity and robustness.

5.4 Ethical and Social Considerations

SLR systems help eliminate communication obstacles yet developers need to protect privacy rights and ensure proper representation and obtain consent from users. The process of dataset creation needs to be transparent while involving community members for proper data curation. System performance differences between groups lead to accessibility issues because of bias so ethical deployment needs continuous monitoring and partnership with Deaf communities. The system must stay away from unauthorized surveillance activities and tracking operations.

5.5 Future Work

1. **Temporal modeling:** Add RNNs/Transformers (e.g., LSTMs, GRUs, ViT) for continuous sign recognition and sentence-level understanding.
2. **Multimodal fusion:** Combine MediaPipe skeletal landmarks with CNN image features for better fine-grained gesture discrimination.
3. **Text-to-speech translation:** Integrate speech synthesis for real-time sign-to-speech conversion.

4. **Data augmentation/collection:** Use GAN-based synthesis and community-sourced data to address class imbalance and improve generalization.
5. **Deployment optimization:** Use quantization and pruning to reduce model size/latency for mobile devices without large accuracy loss.

5.6 Summary of Key Insights

The combination of transfer learning with differential learning rates, label smoothing, and cross-subject validation enables strong generalization for SLR systems even when working with limited datasets. The model achieves a balanced trade-off between accuracy and computational efficiency, showing that MobileNetV2 is highly suitable for real-time, portable applications. Future extensions can further improve robustness through multi-modal inputs and temporal modeling.

6 Conclusion

In this work, we introduced transfer learning strategies for isolated sign language recognition based on the MobileNetV2 architecture, trained under a subject-independent k -fold evaluation protocol. The findings demonstrate that transfer learning provides an effective and reliable solution to the scarcity of labeled sign-language data. The proposed two-phase fine-tuning strategy, comprising head-only adaptation followed by controlled backbone unfreezing, proved effective in leveraging pre-trained ImageNet features while maintaining generalization across unseen signers. The model achieved strong accuracy and macro-F1 scores with stable convergence, confirming that lightweight CNN architectures can successfully balance performance and efficiency in accessibility-oriented computer-vision tasks. The experimental results further reaffirmed the robustness of the cross-subject k -fold validation framework. The moderate cross-fold variation across folds indicates that the model generalizes well to new individuals without relying on signer-specific visual attributes. This robustness validates both the dataset preprocessing pipeline and the applied regularization strategies, including label smoothing, weight decay, and cosine-annealing, which together promoted smooth optimization and minimized overfitting. MobileNetV2 remains a practical choice due to its low computational cost and capability for real-time inference on mobile or embedded devices. From an applied perspective, this research reinforces the value of transfer learning for accessibility technologies. The results suggest that robust, efficient, and ethically designed sign language recognition systems can be developed even under constraints of limited data and hardware. Future work should extend this framework toward continuous sign recognition through temporal modeling (e.g., Transformers or recurrent networks), incorporate multimodal cues such as skeletal landmarks and depth information, and employ community-driven data collection that ensures inclusivity, privacy, and fairness. Combined with responsible and transparent design, such systems have the potential to bridge communication barriers between deaf and hearing communities, making sign language recognition both valuable and socially impactful.

References

- [1] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.
- [2] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4510–4520.
- [3] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” *arXiv preprint arXiv:1704.04861*, 2017.
- [4] S. Kornblith, J. Shlens, and Q. V. Le, “Do Better ImageNet Models Transfer Better?,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2661–2671.
- [5] O. Koller, J. Forster, C. Bowden, and H. Ney, “Continuous Sign Language Recognition: Towards Large Vocabulary Statistical Recognition Systems Handling Multiple Signers,” *Computer Vision and Image Understanding*, vol. 141, pp. 108–125, 2019.
- [6] N. C. Camgoz, O. Koller, S. Hadfield, and R. Bowden, “Sign Language Transformers: Joint End-to-End Sign Language Recognition and Translation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 10023–10033.
- [7] L. Pigou, S. Dieleman, P.-J. Kindermans, and B. Schrauwen, “Sign Language Recognition Using Convolutional Neural Networks,” in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2014, pp. 572–578.
- [8] T. Starner, J. Weaver, and A. Pentland, “Real-Time American Sign Language Recognition Using Desk and Wearable Computer-Based Video,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 12, pp. 1371–1375, 1998.
- [9] E.-J. Ong and S. Ranganath, “Automatic Sign Language Analysis: A Survey and the Future Beyond Lexical Meaning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, pp. 873–891, 2005.
- [10] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely Connected Convolutional Networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4700–4708.
- [11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention Is All You Need,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 5998–6008.
- [12] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

- [13] A. Graves, “Practical Evaluation of Cross-Subject Validation for Sign Language Recognition,” *Pattern Recognition Letters*, vol. 128, pp. 18–24, 2019.
- [14] Ayuraj, American Sign Language Dataset, Kaggle, 2019. Available at: <https://www.kaggle.com/datasets/ayuraj/asl-dataset>