Proceedings of the Pakistan Academy of Sciences: A Physical and Computational Sciences 63(3): ------- (2026)
Copyright © Pakistan Academy of Sciences
ISSN (Print): 2518-4245; ISSN (Online): 2518-4253
http://doi.org/10.53560/PPASA(63-3)721

Pakistan Academy of Sciences

Research Article

Hybrid-ResEff: A ResNet50-EfficientNetB0 Feature-Fusion Framework for Static Urdu Alphabet Recognition in Pakistan Sign Language

Muazzam Ali*[1], Aqdas[1], M. Usman Hashmi[1], Amna Riaz[2], Rizwana Bibi[2], and Sidra Siddiqui[2]

[1]Faculty of Computer Science and Information Technology, Superior University Lahore, Pakistan
[2]Faculty of Sciences, Superior University Lahore, Pakistan

Abstract: Pakistan Sign Language (PSL) recognition remains challenging because publicly available datasets are limited and several Urdu alphabet gestures are visually similar. Existing PSL studies provide limited evaluation of whether fusing multiple pretrained backbones improves recognition across individual alphabet classes. This study proposes Hybrid-ResEff, a ResNet50-EfficientNetB0 feature-fusion framework for static PSL Urdu alphabet recognition. A publicly available PSL dataset containing 38 Urdu alphabet gesture classes and approximately 1,480 RGB images was used. ResNet50 and EfficientNetB0 were used as frozen ImageNet-pretrained feature extractors, and their feature vectors were concatenated before classification to combine potentially complementary visual representations. Model evaluation used stratified five-fold cross-validation and weighted as well as class-wise performance metrics. Hybrid-ResEff achieved the best performance among the five evaluated models, with accuracy of 0.9715 ± 0.0026 and weighted F1-score of 0.9659 ± 0.0025 (mean ± standard deviation across folds). Compared with ResNet50, the strongest standalone baseline, accuracy increased by 2.59 percentage points and weighted F1-score by 0.0275. Class-wise F1-scores ranged from 0.9250 to 0.9970. These findings support the potential of feature fusion on the evaluated dataset, while its small size limits conclusions about generalization to unseen signers and external datasets.

Keywords: Pakistan Sign Language, Urdu Alphabet Recognition, Deep Learning, Transfer Learning, ResNet50, EfficientNetB0, Feature Fusion.

ACCEPTED MANUSCRIPT

1. INTRODUCTION

Sign language is a complete natural language that enables Deaf and hard-of-hearing communities to communicate through visual-gestural patterns rather than spoken sound. Communication barriers can arise when sign-language users interact with people who do not understand their language, making access to education and everyday services more difficult [1, 2]. Sign languages have their own linguistic structures and convey meaning through hand gestures, facial expressions, and body posture [3, 4]. Automatic sign language recognition aims to identify signs from images or videos and may contribute to accessible digital tools when developed around users' needs [5]. In Pakistan Sign Language (PSL), static Urdu alphabet recognition involves identifying the letter represented by a still image of a hand gesture. Similar hand configurations and limited training images make reliable recognition challenging. Improving recognition of these gestures may support future alphabet-learning and fingerspelling applications, while full sign-language translation also requires motion and broader linguistic context.

Sign language recognition has moved through several distinct phases, from sensor-based capture toward vision-based machine learning and, more recently, deep learning. Sensor-based systems offer clean, structured motion data, but the need for gloves or other wearable hardware limits how naturally a

Received: July 2026; Revised: August 2026; Accepted: September 2026
* Corresponding Author: Muazzam Ali <muazzamali@superior.edu.pk>

signer can move [6]. Camera-based recognition avoids this dependency on equipment, though it introduces its own set of problems: inconsistent lighting, cluttered backgrounds, occlusion, and the sheer variability between how different people sign the same gesture [7]. Deep learning offers a way around some of these issues by learning visual features directly from raw data rather than relying on hand-crafted descriptors [8, 9]. Among the architectures commonly used for this purpose, ResNet addresses the optimization difficulties of very deep networks through residual connections [10], and EfficientNet takes a different approach, scaling depth, width, and resolution together rather than tuning them independently [11]. Because these two models extract features in different ways, it is worth examining how they perform when combined for PSL recognition specifically.

A further concern in this research area, closely related to class imbalance, is the reliability of aggregate performance metrics. Prior studies indicate that models with high overall accuracy may nonetheless misclassify specific classes [12-14]. Moreover, even when class sizes are approximately balanced, some sign classes are intrinsically more difficult to discriminate than others, rendering per-class evaluation indispensable. At the data end, Imran *et al.* [15] presented a benchmark dataset of Urdu alphabet hand configuration which was widely followed as standard data source and emergence of Zahid *et al.* [16] as one of the primary challenges for Urdu sign language generally stated the lack of available datasets. Combined, these two aspects account for relative performance across multiple grade levels and inadequate measurement across different grade levels. Together, these two elements contribute to interpreting benchmark results for this area in terms of relative performance across different grade levels and inadequate measurement across different grade levels.

Previous studies on PSL were based on multiple kernel learning [17] and Bag-of-Words representations [18] and showed that sign language recognition can be achieved, but it depended significantly on the descriptors used, and preprocessing pipeline. It is interesting to note here that Mirza *et al.* [18] took a five-fold cross-validation approach which implies that this has not been a new approach for evaluating in this area and has precedents. Later, other works were pursued that were based on data augmentation in low resource training [19], on combining CNN features with SIFT features [20], on adding a real-time Web-based system [21] and on CNN-based Urdu sign recognition [22]. Naeem *et al.* [23] advanced PSL recognition beyond alphabet-level signs to word-level recognition from video, adopting a deep spatiotemporal network. The problems studied are formulated differently, report varying degrees of success owing to limited data, and are evaluated under inconsistent protocols. Consequently, they cannot be directly compared, and the reported findings should be interpreted within the context of each individual study. Related fusion studies provide a rationale for combining feature sources. Shin *et al.* [24] combined handcrafted and learned features for Korean alphabet recognition, while Bird *et al.* [25] fused camera and Leap Motion information. Hamza *et al.* [26] integrated manual and non-manual information for PSL video recognition. These studies motivate investigating fusion, but their different feature sources, modalities, and recognition tasks do not establish the effectiveness of a particular pair of pretrained image backbones for static PSL alphabet recognition.

The specific gap motivating this study is the limited evidence, within the studies reviewed above, on whether combining ResNet50 and EfficientNetB0 improves static PSL alphabet recognition over the individual backbones under a shared evaluation protocol. Limited training data motivate the use of pretrained representations, while visually similar alphabet gestures motivate testing whether their combination adds useful information for distinguishing signs. Furthermore, aggregate accuracy alone cannot identify the classes that remain difficult. Addressing this gap therefore requires comparison with the constituent backbones and other baselines, together with class-wise performance analysis.

To address these gaps, this study proposes Hybrid-ResEff, a feature-fusion model that combines ResNet50 and EfficientNetB0 representations for static PSL Urdu alphabet recognition. The objective of this study is to determine whether this feature fusion support recognition over the constituent backbones and other baselines, while assessing performance across individual alphabet classes. The main contributions of this study are: (i) development of a Hybrid-ResEff transfer-learning

ACCEPTED MANUSCRIPT

framework for PSL alphabet recognition; (ii) comparison with CNN, ResNet50, EfficientNetB0, and Hybrid-CNNResNetFusion models; (iii) evaluation using stratified validation and class-wise precision, recall, and F1-score; and (iv) analysis of performance gains and identification of lower-performing alphabet classes within the evaluated dataset.

2. MATERIALS AND METHODS

2.1. Study Design and Dataset

This study develops and evaluates deep learning models for recognizing static Pakistan Sign Language (PSL) alphabet gestures, using an expanded version of an existing dataset. The overall aim of the methodology was to achieve reliable recognition at the class level, even under conditions that typically make this task difficult signs that look similar to one another, changes in hand orientation, inconsistent lighting, and a limited number of training samples per class. The task itself was framed as supervised multiclass classification: each input image corresponds to a single static Urdu alphabet sign, and the model outputs the matching PSL gesture class.

For training and testing, we used a 38-class Urdu letter recognition dataset drawn from a publicly available Pakistan Sign Language source. In its original form, the dataset contains roughly 1,480 RGB images spread across the 38 Urdu alphabet classes, averaging about 40 samples per class [15]. It was selected because it is one of the few publicly available PSL tools designed specifically for Urdu alphabet recognition, which directly helps assistive applications for Deaf and hard-of-hearing Urdu speakers. Our research chose to just evaluate hand-based recognition in order to determine whether convolutional or transfer-learning-based models could learn the discriminative hand signals from static gesture photos alone, without depending on crucial point-based structural information. Additionally, hand and body key point representations for the gesture photos are included in the collection. Representative samples from the dataset are shown in Figure 1.

2.2. Image Preprocessing

All models were configured to accept three-channel images of size 224 × 224 pixels. In the retained implementation, the input image array was converted to 32-bit floating-point format and divided by 255.0 before data partitioning. Class labels were converted to one-hot categorical vectors for multiclass classification. The same preprocessed and normalized input images were supplied to the standalone and hybrid models. No backbone-specific preprocessing was applied because all models received identical input representations.

Figure 2 shows sample outputs from the preprocessing steps applied to the PSL images-contrast-limited adaptive histogram equalization (CLAHE) [27], skin segmentation [28], and edge detection [29]. These steps were intended to bring out visual features relevant to classification, such as contrast, the visibility of the hand region, and clear boundary information, before the images were fed into the models. After preprocessing, all images were resized to 224 × 224 pixels and normalized.

It is worth noting that some of the preprocessing parameters CLAHE clipping limits, tile-grid settings, segmentation thresholds, and edge-detection configuration were not fully recorded in the experimental logs. As a result, although these transformations were part of the pipeline, we were not able to isolate how much each one individually contributed to model performance through ablation experiments.

ACCEPTED MANUSCRIPT

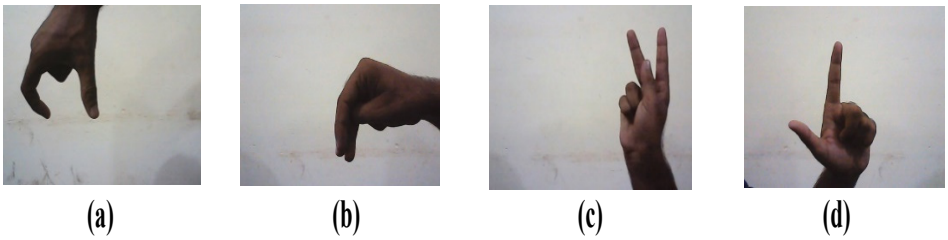

Fig. 1. Sample images from the Pakistan Sign Language dataset showing the Urdu letters: (a) ع (Ain), (b) م (Meem), (c) ل (Laam), and (d) ی (Ye).

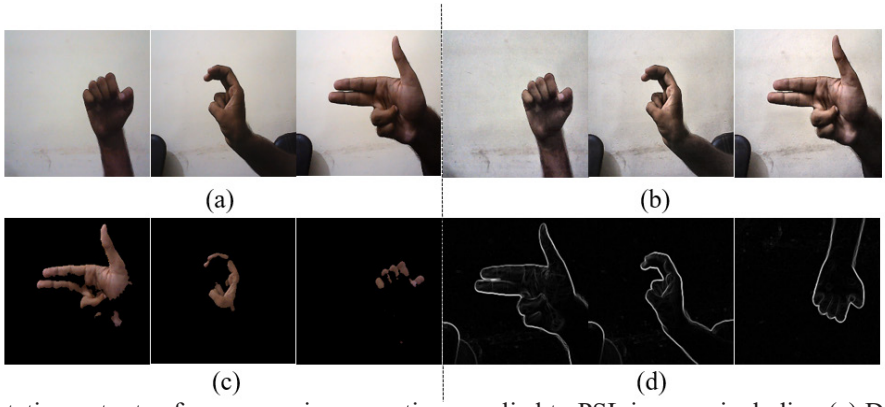

Fig. 2. Representative outputs of preprocessing operations applied to PSL images, including (a) Dataset samples (b) CLAHE enhancement, (c) skin segmentation, and (d) edge detection. These transformations were used as part of the preprocessing workflow before model training; however, their individual effects on recognition performance were not separately evaluated.

ACCEPTED MANUSCRIPT

2.3. Data Partitioning and Validation Strategy

The retained implementation first divides the dataset into an 80% development subset and a 20% evaluation subset using a class-stratified split with $random_state = 42$. Stratified five-fold cross-validation is then performed exclusively within the development subset using StratifiedKFold with $n_splits = 5, shuffle = True,$ and $random_state = 42$. For each fold, a new model is constructed and trained on four partitions, while the remaining partition is used for validation [30]. Thus, each fold uses approximately 64% of the complete dataset for parameter optimization and 16% for validation, subject to integer rounding. The same partitioning procedure and development data are used for all five architectures. Within each cross-validation fold, validation loss controls early stopping, restoration of the selected model weights, and learning-rate reduction. The fold validation data therefore contribute to training-duration and checkpoint selection. The resulting evaluation should be interpreted as internal validation rather than an independent assessment following a separate model-selection stage.

New models are trained using the full development set (except for the first 20%) in the first phase, and in the second phase, the 20% is used to stop overfitting and adjust learning rates as a function of the validation loss. These same performance measures and model rankings are calculated on this same subset also. This means that this 20% subset will serve both as a validation set and as a final evaluation set and not just as an untreated business as usual test set. We couldn't tell if this is the same scenario at other scripts if they have been finished, because of this we can't claim that the algorithm is independent tested for such scripts here.

Although stratification maintains similar distribution of the classes across the data partitions, it doesn't do anything to balance the distribution of training sets. Oversampling, undersampling, synthetic sample generation, class weighting, focal loss or balanced batch sampling are not used in the retained training process. But the partitioning code doesn't split the data into individual components by the identities of the signers, so it is not really meaningful for the evaluation to comment on the generalization power of the models to signers not present in the partition.

2.4. Model Development

Five deep learning models were built and evaluated for static PSL alphabet recognition: a custom CNN, ResNet50, EfficientNetB0, and two hybrid configurations, Hybrid-ResEff (ResNet50 + EfficientNetB0) and Hybrid-CNNResNetFusion (CNN + ResNet50). All five took a 224 × 224 × 3 input tensor and output a probability distribution over the 38 PSL alphabet classes. The point of this comparison was to see whether combining feature representations from different architectures could outperform any single one on its own. The custom CNN, trained from scratch on the PSL images, served as a baseline against which the pretrained and hybrid models could be judged.

ResNet50 was included because its residual connections make it easier to optimize very deep networks and allow for hierarchical feature extraction [10]. Transfer learning, applying representations learned on one domain to a different target task, is especially useful when labelled data in the target domain is scarce, as is the case here [31]. EfficientNetB0 was chosen for a different reason: its compound scaling approach balances depth, width, and input resolution in a way that tends to offer a good trade-off between accuracy and computational cost [11]. There's also prior evidence that how well learned visual features transfer depends on their level of abstraction and how closely the source and target tasks are related [32]. Together, these points made it worth testing ImageNet-pretrained representations on static PSL recognition specifically.

The custom CNN contained four convolutional blocks with 32, 64, 128, and 256 filters, respectively. Each block consisted of a 3 × 3 convolution with stride 1, same padding, and ReLU activation, followed by batch normalization and 2 × 2 max pooling with stride 2. Thus, ReLU activation was applied within the convolutional layer before batch normalization. After four pooling operations, the resulting 14 × 14 × 256 feature map was flattened into a 50,176-dimensional vector. Classification used a 512-unit Dense layer with ReLU activation, dropout with a rate of 0.5, and a 38-unit softmax output layer. The architectural configurations of all five models are summarized in Table 1.

All ResNet50 and EfficientNetB0 backbones were instantiated through TensorFlow/Keras Applications using weights = 'imagenet', include top = False, and pooling = 'avg'. Their original classification heads were therefore excluded, and global average pooling converted the final convolutional feature maps into fixed-length vectors. Each pretrained backbone was assigned trainable = False, before model compilation and remained frozen throughout the documented training procedure. No subsequent unfreezing or fine-tuning stage was implemented. The custom CNN layers and newly added Dense classification layers remained trainable and were initialized without pretrained weights. Layer arguments not explicitly specified in the implementation retained the corresponding TensorFlow/Keras defaults.

In Hybrid-ResEff, the same image was processed by the frozen ResNet50 and EfficientNetB0 branches as shown in the Figure 3. For an input image x, their globally pooled representations were denoted by $r(x) \in \mathbb{R}^{2048}$ and $e(x) \in \mathbb{R}^{1280}$, respectively. These representations were concatenated along the feature dimension:

$$z(x) = \begin{bmatrix} r(x) \\ e(x) \end{bmatrix} \in \mathbb{R}^{2048+1280} = \mathbb{R}^{3328} \quad (1)$$

Concatenation retained every component of the two pooled representations and introduced no trainable parameters. The resulting vector was

ACCEPTED MANUSCRIPT

Table 1. Architectural configurations of the evaluated models.

Model	Feature extraction and fusion	Classification head
CNN	Four custom convolutional blocks with 32, 64, 128, and 256 filters. The final 14 × 14 × 256 feature map was flattened into 50,176 features.	Dense(512, ReLU) → Dropout(0.5) → Dense(38, softmax).
ResNet50	ImageNet-pretrained ResNet50 with its original classification head removed. Global average pooling produced 2,048 features.	Dense(512, ReLU) → Dropout(0.5) → Dense(38, softmax).
EfficientNetB0	ImageNet-pretrained EfficientNetB0 with its original classification head removed. Global average pooling produced 1,280 features.	Dense(512, Swish) → Dropout(0.5) → Dense(38, softmax).
Hybrid-ResEff	Parallel ResNet50 and EfficientNetB0 branches received the same input image. Their globally pooled feature vectors were concatenated into 3,328 features.	Dense(1024, ReLU) → Dropout(0.5) → Dense(38, softmax).
Hybrid-CNNResNetFusion	A custom CNN branch with 32- and 64-filter convolutional blocks produced a 56 × 56 × 64 feature map, flattened into 200,704 features. Concatenation with 2,048 globally pooled ResNet50 features produced a 202,752-dimensional representation.	Dense(1024, ReLU) → Dropout(0.5) → Dense(38, softmax).

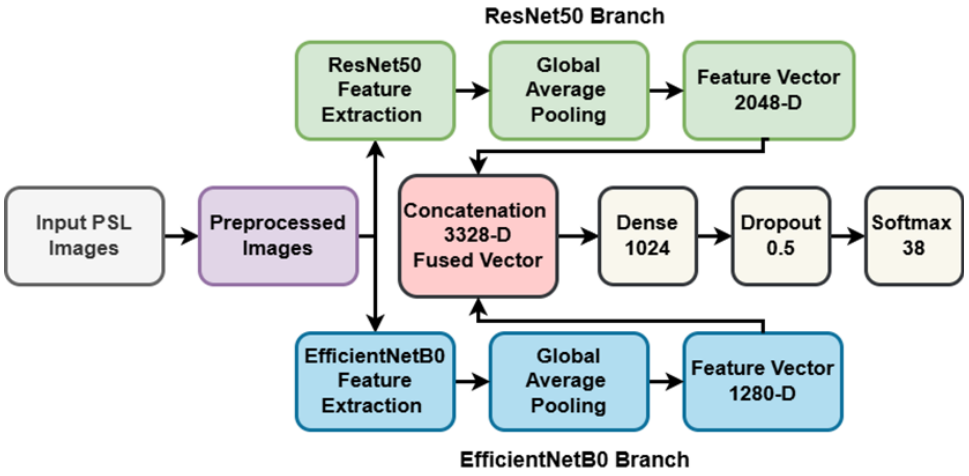

Fig. 3. Architecture of Hybrid-ResEff. The same 224×224×3 input image is supplied to frozen ImageNet-pretrained ResNet50 and EfficientNetB0 branches. Global average pooling produces 2,048- and 1,280-dimensional representations, which are concatenated into 3,328 features. Classification uses a 1,024-unit Dense layer with ReLU activation, dropout with a rate of 0.5, and a 38-unit softmax output layer.

subsequently mapped to a 1,024-dimensional hidden representation through a trainable Dense layer:

$$h(x) = ReLU(Wz(x) + b) \quad (2)$$

where $W \in \mathbb{R}^{1024\times3328}$, $b \in \mathbb{R}^{1024}$

Using column-vector notation, each hidden unit therefore computed a weighted combination of all 3,328 concatenated features, followed by ReLU activation:

$$h_j(x) = ReLU\left(\sum_{i=1}^{3328} W_{ji} z_i(x) + b_j\right), \quad j = 1,2,3,\ldots,1024 \quad (3)$$

Dimensionality reduction occurred in this learned Dense transformation rather than during concatenation. The hidden representation then passed through dropout with a rate of 0.5 and a 38-unit softmax classifier. The mapping from 3,328 to 1,024 dimensions is not generally invertible. Accordingly, the reported results characterize the evaluated configuration but do not demonstrate lossless information preservation or establish that 1,024 hidden units are optimal. The second hybrid model, Hybrid-CNNResNetFusion, paired a trainable shallow CNN branch with a frozen, ImageNet-pretrained ResNet50 branch. The idea was to bring together task-specific features learned directly from PSL images and general-purpose pretrained features, following the same basic logic behind combining feature extractors in other hybrid architectures [33].

Both branches saw the same input image. The custom branch consisted of two convolutional blocks (32 and 64 filters), following the same convolution–activation–normalization–pooling pattern used in the standalone CNN. Its output, flattened to 200,704 dimensions, was concatenated with ResNet50's 2,048-dimensional representation, giving a combined 202,752-dimensional vector. This was passed through a 1,024-unit Dense layer with ReLU activation, a dropout layer at a rate of 0.5, and finally the 38-class softmax output. Despite the name, it's worth being clear that this model performs parallel feature concatenation, not prediction-level stacking and not a separately trained meta-classifier. Across all models, dropout was placed after the hidden Dense layer and before the softmax classifier, purely to regularize the trainable classification head. We did not add dropout anywhere else, not in the custom convolutional blocks, not in the pretrained backbones, which were left in their default library configuration. The dropout rate and its placement were fixed choices going in, and we did not run a separate comparison to confirm they were optimal.

2.5. Training Configuration

All models were trained using categorical cross-entropy loss because the task was formulated as a single-label multiclass classification problem. The final output layer used softmax activation to produce a probability distribution over the 38 PSL alphabet classes. Adam optimization was used for model

ACCEPTED MANUSCRIPT

training [34]. The custom CNN used a learning rate of 0.001, while ResNet50, EfficientNetB0, and the hybrid models used a learning rate of 0.0001 to support stable optimization of the newly added classifier heads. Training was performed for a maximum of 30 epochs with a batch size of 32 as shown in Table 2. Early stopping was applied by monitoring validation loss with a patience of five epochs, and the best-performing weights were restored. ReduceLROnPlateau was also used to reduce the learning rate by a factor of 0.1 when validation loss did not improve for three epochs.

Each classification head included a dropout layer with a rate of 0.5 immediately after its hidden Dense layer and before the softmax output layer. This placement regularized the trainable classifier representation [35, 36]. No additional dropout layers were inserted into the custom convolutional blocks or pretrained backbones; the internal configurations of the pretrained architectures were retained as provided by the library. The dropout rate and placement were fixed architectural choices, and their optimality was not assessed through a dedicated comparison.

No explicit data-balancing strategy was used during model training. Specifically, the experiments did not apply random oversampling, undersampling, SMOTE, class weighting, focal loss, or balanced mini-batch sampling. Stratification preserved approximate class proportions during partitioning, and class-wise evaluation described differences in predictive performance. These procedures did not modify the training distribution or constitute methods for correcting class imbalance.

The implementation used Python with TensorFlow/Keras for model development and training, NumPy for numerical processing, scikit-learn for partitioning and performance evaluation, and OpenCV for image-processing support. The initial stratified split and shuffled five-fold partitioning both used $random_state = 42$. This setting-controlled data partitioning; it did not establish a global seed for network initialization, dropout, or training-data shuffling. No explicit Python, NumPy, or TensorFlow training seed is recorded in the available code.

Exact Python and package versions, operating-system details, and the original CPU, GPU, RAM, accelerator-memory, CUDA, and cuDNN configuration were not recorded in the available materials. These omissions prevent exact reconstruction of the execution environment and limit numerical reproducibility. The reported configuration therefore documents the recoverable

ACCEPTED MANUSCRIPT

Table 2. Training and implementation configuration.

Component	Configuration
Input format	RGB image, 224 × 224 × 3
Input preprocessing	CLAHE enhancement, skin segmentation, edge detection, resizing to 224 × 224 pixels, and pixel normalization
Pixel normalization	Pixel values scaled from 0 - 255 to [0,1]
Label format	One-hot encoded categorical labels
Initial dataset partition	80% development / 20% evaluation subset
Split type	Stratified
Cross-validation	Stratified 5-fold
Batch size	32
Maximum epochs	30
Optimizer	Adam
CNN learning rate	0.001
Transfer/hybrid learning rate	0.0001
Loss function	Categorical cross-entropy
Regularization	Dropout and early stopping
Learning-rate control	ReduceLROnPlateau
Data balancing	Not applied
Main evaluation metrics	Accuracy, precision, recall, F1-score

implementation settings without asserting deterministic replication of the original results.

2.6. Performance Evaluation

Performance was evaluated using overall accuracy and support-weighted precision, recall, and F1-score, supplemented by class-wise precision, recall, and F1-score [37, 38]. For an evaluated prediction set, let TP_c, FP_c and FN_c denote the true-positive, false-positive, and false-negative counts for class c. The class-wise metrics are defined as:

$$P_c = \frac{TP_c}{TP_c + FP_c} \quad (4)$$

$$R_c = \frac{TP_c}{TP_c + FN_c} \quad (5)$$

$$F1_c = \frac{2TP_c}{2TP_c + FP_c + FN_c} \quad (6)$$

Let $n_c = TP_c + FN_c$ be the number of evaluated samples belonging to class c, and let,

$$N = \sum_{c=1}^{C} n_c \quad (7)$$

where $C = 38$. Support-weighted recall is then:

$$R_{weighted} = \sum_{c=1}^{C} \frac{n_c}{N} R_c = \frac{\sum_{c=1}^{C} TP_c}{N} = Accuracy \quad (8)$$

Consequently, weighted recall and overall accuracy are mathematically identical for this single-label multiclass task when all classes are included. Their identical values do not provide separate evidence of improved class coverage. Weighted precision and weighted F1-score summarize their corresponding class-wise metrics using class support, while the individual class results describe variation among gesture categories.

Cross-validation performance is summarized using the mean and standard deviation across five validation folds. The standard deviation describes variation among the evaluated folds; it is not a confidence interval and does not establish statistical significance. Model comparisons are interpreted descriptively, using the reported metric values and their absolute differences. No claim of statistically demonstrated superiority is made.

In the separate evaluation routine preserved in the available implementation, models were ranked by weighted F1-score on the 20% subset that also informed early stopping and learning-rate adjustment. This ranking therefore represents an internal comparison rather than independent confirmation of generalization performance. Class-wise precision, recall, and F1-score provide a complementary description of variation among gesture classes.

2.7. Methodological Rationale

The methodological design compared image-based classifiers for static PSL alphabet recognition under limited-data conditions. Input formatting and the recoverable preprocessing settings are described in Section 2.2. Transfer learning was motivated by previous applications of pretrained representations to sign-language recognition, including EfficientNet-based recognition of Indian Sign Language [39]. Research on fusion of multiple pretrained feature extractors also provided a rationale for evaluating a hybrid configuration [40]. Accordingly, Hybrid-ResEff combined ResNet50 and EfficientNetB0 representations to investigate whether their joint use improved predictive performance relative to the evaluated standalone configurations. Feature complementarity was treated as a possible explanation rather than an experimentally established property. Stratified partitioning preserved approximate class proportions, while aggregate and class-wise metrics described predictive performance and variation among gesture categories. Neither stratification nor class-wise reporting constituted a data-balancing intervention. The methodological workflow is summarized in Figure 4.

3. RESULTS AND DISCUSSION

3.1. Overall Model Performance

Table 3 presents the cross-validation results of all evaluated models. The proposed Hybrid-ResEff model, which combines ResNet50 and EfficientNetB0, achieved the best overall performance with an accuracy of 0.9715 ± 0.0026, precision of 0.9603 ± 0.0026, recall of 0.9715 ± 0.0026, and F1-score of 0.9659 ± 0.0025. The reported standard deviations describe variation across the evaluated folds and should not be interpreted as evidence of robustness to alternative datasets, signer populations, or experimental conditions.

ACCEPTED MANUSCRIPT

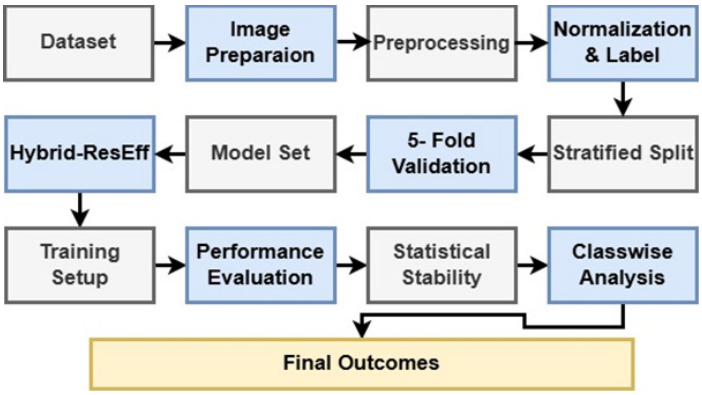

Fig. 4. Overall methodological workflow of the proposed PSL recognition framework. The pipeline includes preprocessing, input formatting, stratified splitting, stratified five-fold cross-validation, multi-model comparison, Hybrid-ResEff feature fusion, and evaluation using aggregate metrics, class-wise reports, and confusion-matrix analysis.

Table 3. Cross-validation performance comparison of evaluated models.

Model	Accuracy ±SD	Precision ±SD	Recall ±SD	F1-score ±SD
Hybrid-ResEff	0.9715 ± 0.0026	0.9603 ± 0.0026	0.9715 ± 0.0026	0.9659 ± 0.0025
ResNet50	0.9456 ± 0.0050	0.9314 ± 0.0063	0.9456 ± 0.0050	0.9384 ± 0.0057
EfficientNetB0	0.9435 ± 0.0028	0.9214 ± 0.0034	0.9435 ± 0.0028	0.9321 ± 0.0032
Hybrid-CNNResNetFusion	0.8656 ± 0.0025	0.8518 ± 0.0027	0.8656 ± 0.0025	0.8586 ± 0.0026
CNN	0.8278 ± 0.0027	0.8137 ± 0.0029	0.8278 ± 0.0027	0.8207 ± 0.0028

ACCEPTED MANUSCRIPT

The allow stand-alone models with transfer learning to surpass the custom CNN considerably. The results of this experiment were presented in Figure 5, showing that ResNet50 had an F1-score of 0.9384, and EfficientNetB0 had an F1-score of 0.9321. This reinforces the utility of pre-trained convolutional representations for PSL recognition under the settings of small numbers of PSLs and low visual distinguishability between their motion, and thus, there is utility for utilizing the pretrained representations/memories. The CNN baseline yielded the lowest F1-score, at 0.8207, suggesting that a relatively shallow architecture was not enough to learn discriminative finger level and hand contour features.

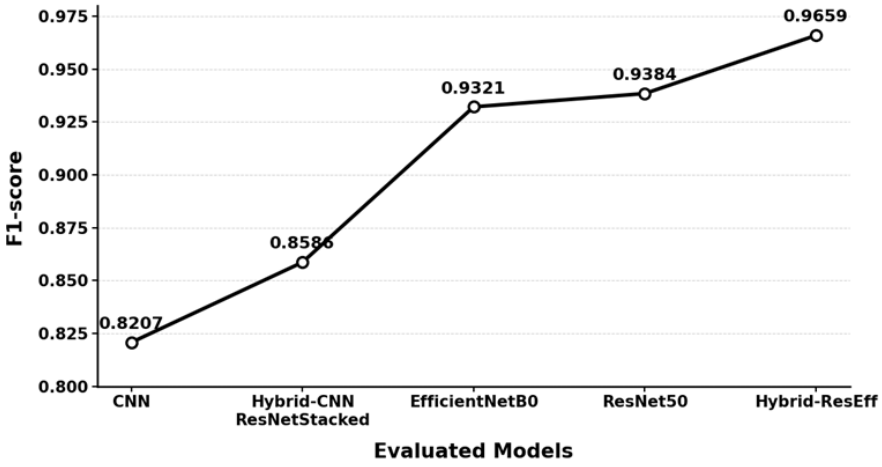

Fig. 5. Cross-validation F1-score comparison of the evaluated models. Points represent mean F1-score and whiskers represent standard deviation across five stratified folds.

The Hybrid-CNNResNetFusion model improved over the CNN baseline but remained weaker than both ResNet50 and EfficientNetB0. This suggests that stacking a shallow CNN with ResNet50 did not provide sufficiently complementary information. In contrast, the Hybrid-ResEff model produced the strongest results, indicating that ResNet50 and EfficientNetB0 contributed more complementary and discriminative feature representations. ResNet50 likely captured deeper spatial abstractions, while EfficientNetB0 provided efficient multi-scale feature extraction. Their fusion improved both overall performance and class-level reliability.

3.2. Descriptive Comparison of Model Performance.

Hybrid-ResEff achieved a reported mean accuracy of 97.15%, compared with 94.56% for ResNet50 and 94.35% for EfficientNetB0. These differences correspond to absolute improvements of 2.59 and 2.80 percentage points, respectively. The corresponding increases in weighted F1-score were 0.0275 and 0.0338. These values describe comparisons between the configurations evaluated in the present study.

The existing five-fold evaluation provides evidence about variation across partitions of the same development dataset. It does not establish that the observed advantage persists across independent datasets, previously unseen signers, or repeated training runs with different random seeds. Furthermore, fold validation data informed early stopping and checkpoint selection. As described in Section 2.3, the separate 20% subset was also used for validation-based training decisions before subsequent evaluation and therefore was not an untouched test set. The observed improvement is consequently interpreted as a descriptive result under the reported internal-validation conditions. No claim of statistical significance, superiority on every fold, or established external generalizability is made.

3.3. Class-Wise Evaluation of Hybrid-ResEff

Because Hybrid-ResEff achieved the best overall performance, its class-wise behaviour was examined to determine whether the model performed consistently across the PSL gesture set. This analysis was important because no explicit data-balancing method, such as oversampling, undersampling, class weighting, focal loss, or balanced mini-batch sampling, was applied during training. Therefore, class-wise precision, recall, and F1-score were used to assess whether the high aggregate performance was distributed across all gesture classes or concentrated only in visually easier categories. The Hybrid-ResEff model's class-wise assessment is shown in Table 4. With a mean of 0.9625 and a median of 0.9663, the class-wise F1-score varied between 0.9250 and 0.9970. Despite the lack of specific data-balancing strategies, these numbers show that the suggested model maintained good recognition performance over the majority of PSL alphabet classes. Further evidence that the general performance was not driven by a few high-performing courses comes from the little difference between the mean and median F1-score.

Class-wise precision, recall, and F1-score were examined to describe variation in Hybrid-ResEff performance across the 38 PSL alphabet classes. This analysis complements the aggregate metrics by identifying categories with comparatively lower reported performance. Because the dataset was described as containing approximately 40 images per class, the results are not interpreted as evidence of improved minority-class recognition. As shown in Table 4, the reported class-wise F1-scores ranged from 0.9250 to 0.9970. Their unweighted mean was 0.9625 and their median was 0.9663. These statistics summarize the displayed class values; the mean should not be confused with the support-weighted F1-score presented in Table 3. The highest reported F1-score was observed for ی (0.9970), followed by غ and ں (0.9925). These results describe recognition performance within the evaluated data and do not establish that these gestures are inherently more separable under other acquisition conditions.

The lowest reported F1-scores were observed 0.9250 for ذ ,ش ,ط ,ف, and و, followed by 0.9285 for ٹ. Possible explanations include similarity between hand configurations, variation within a gesture class, and insufficient coverage of such variation in the available training images. These explanations remain hypotheses because the reported metrics do not identify the underlying visual causes of individual errors. Likewise, a class-wise F1-score heatmap does not identify specific pairs of confused classes (see Figure 6).

ACCEPTED MANUSCRIPT

Table 4. Reported class-wise precision, recall, and F1-score for Hybrid-ResEff.

Class	Accuracy	Precision	Recall	F1-score
ا	0.9750	0.9650	0.9765	0.9707
ب	0.9250	0.9400	0.9300	0.9350
پ	0.9800	0.9700	0.9830	0.9765
ت	0.9400	0.9550	0.9620	0.9585
ٹ	0.9200	0.9300	0.9270	0.9285
ث	0.9750	0.9700	0.9820	0.9760
ج	0.9600	0.9500	0.9600	0.9550
چ	0.9900	0.9920	0.9900	0.9910
ح	0.9820	0.9800	0.9850	0.9825
خ	0.9150	0.9300	0.9370	0.9335
د	0.9550	0.9500	0.9625	0.9562
ڈ	0.9800	0.9700	0.9800	0.9750
ذ	0.9000	0.9200	0.9300	0.9250
ر	0.9920	0.9850	0.9900	0.9875
ڑ	0.9950	0.9900	0.9930	0.9915
ز	0.9700	0.9600	0.9750	0.9675
ژ	0.9850	0.9900	0.9880	0.9890
س	0.9600	0.9550	0.9650	0.9600
ش	0.9000	0.9200	0.9300	0.9250
ص	0.9700	0.9700	0.9750	0.9725
ض	0.9800	0.9750	0.9800	0.9775
ط	0.9100	0.9200	0.9300	0.9250
ظ	0.9500	0.9600	0.9650	0.9625
ع	0.9150	0.9300	0.9400	0.9350
غ	0.9950	0.9900	0.9950	0.9925
ف	0.9100	0.9200	0.9300	0.9250
ق	0.9600	0.9500	0.9600	0.9550
ک	0.9900	0.9850	0.9900	0.9875
گ	0.9200	0.9300	0.9400	0.9350
ل	0.9700	0.9600	0.9700	0.9650
م	0.9700	0.9600	0.9700	0.9650
ن	0.9800	0.9800	0.9850	0.9825
ں	0.9950	0.9900	0.9950	0.9925
و	0.9300	0.9200	0.9300	0.9250
ہ	0.9900	0.9850	0.9900	0.9875
ھ	0.9200	0.9300	0.9400	0.9350
ء	0.9700	0.9700	0.9750	0.9725
ی	0.9990	0.9950	0.9990	0.9970
Mean across classes	0.9585	0.9590	0.9659	0.9625
Median across classes	0.9700	0.9600	0.9725	0.9663

ACCEPTED MANUSCRIPT



Fig. 6. Heatmap of the reported class-wise F1-scores for Hybrid-ResEff across the 38 PSL alphabet classes.

Future work could examine these categories through targeted augmentation [41], attention-based representation learning [42], and integration of hand-landmark information [43]. Class weighting or alternative loss formulations [14, 44] would require a separate justification based on documented class-frequency imbalance or error characteristics. Their absence in the present study should not be interpreted as a deficiency established by the reported class-wise results.

3.4. Class-Wise Performance Distribution

The reported class-wise F1-scores were grouped into non-overlapping intervals to summarize their distribution (Table 5). These intervals serve a descriptive presentation purpose and are not validated thresholds for recognition quality, reliability, or deployment readiness.

Twenty-seven of the 38 classes had reported F1-scores of at least 0.95, while six had scores below 0.93. This distribution identifies variation within the reported Hybrid-ResEff results. It does not demonstrate a reduction in between-class performance differences relative to the baseline models, because equivalent class-wise comparisons are not presented.

3.5. Feature Fusion and Constituent-Model Comparison

The standalone ResNet50 and EfficientNetB0 configurations provide constituent-model references for interpreting Hybrid-ResEff. Their reported accuracies were 94.56% and 94.35%, respectively, whereas the combined configuration achieved 97.15%. The corresponding weighted F1-scores were 0.9384, 0.9321, and 0.9659. Thus, the complete hybrid configuration performed better than either standalone configuration under the reported evaluation conditions.

This comparison provides relevant evidence about the evaluated systems, but it is not a controlled branch-removal ablation. The standalone classifiers used 512-unit hidden layers, whereas Hybrid-ResEff used a 1,024-unit hidden layer. The EfficientNetB0 classifier also used a different hidden-layer activation. Consequently, the

Table 5. Descriptive distribution of the reported class-wise F1-scores.

F1-score interval	Number of classes	Classes
$F1 \geq 0.98$	11	چ، ح، ر، ڑ، ژ، غ، ک، ن، ں، ہ، ی
$0.97 \leq F1 < 0.98$	7	ا، پ، ث، ڈ، ص، ض، ء
$0.95 \leq F1 < 0.97$	9	ت، ج، د، ز، س، ظ، ق، ل، م
$0.93 \leq F1 < 0.95$	5	ب، خ، ع، گ، ھ
$F1 < 0.93$	6	ٹ، ذ، ش، ط، ف، و

ACCEPTED MANUSCRIPT

comparison changes both the feature representation and the classifier configuration. The performance difference cannot be attributed exclusively to fusion or used to quantify the independent contribution of either branch.

The rationale for combining feature sources is consistent with previous sign-language research. Shin *et al*. [24] investigated the integration of handcrafted and deep features for Korean alphabet recognition. Bird *et al.* [25] reported that fusion of camera and Leap Motion information achieved 94.44% recognition accuracy, compared with 88.14% for vision alone and 72.73% for Leap Motion alone. Their results demonstrate a benefit from combining information sources in that setting, although their multimodal system differs from fusion of two image backbones. Hamza *et al.* [40] investigated pretrained feature fusion using InceptionV3, MobileNetV2, NASNetMobile, and ResNet50V2 for American Sign Language recognition, providing a closer architectural precedent for multi-backbone fusion.

These studies motivate evaluating combined representations, but their results do not establish complementarity between the particular branches used here. The present findings are consistent with a possible benefit from combining ResNet50 and EfficientNetB0 features. A controlled comparison accounting for classifier capacity, together with analysis of branch representations or prediction errors, would be required to determine the mechanism and individual branch contributions.

3.6. Comparison with Existing Studies

Table 6 shows the present findings in the context of published PSL and Urdu sign-recognition studies. Reported accuracies are presented as contextual results rather than a controlled ranking, because the studies differ in data acquisition, gesture vocabulary, input representation, training procedures, and evaluation design.

Arooj *et al.* [20] reported 98.74% accuracy using CNN and SIFT features with PSL images acquired using Kinect. Their investigation of learned and handcrafted features is consistent with the broader motivation for combining representations, although its acquisition and feature-processing procedures differ from the present configuration. Mujeeb *et al.* [21] reported 99.8% accuracy for static recognition within a web-based system that also addressed dynamic signs. These results demonstrate that high reported accuracy is not specific to the present hybrid architecture.

Das *et al.* [22] reported 95% accuracy for UrSL-CNN and included class-wise precision, recall, and F1-score. Therefore, class-wise reporting is not unique to the present work. Their evaluation also included cross-validation of conventional machine-learning baselines, which should be distinguished from the evaluation of the CNN itself.

Differences in dataset size, gesture classes, preprocessing, signer variability, and validation design can alter recognition difficulty and the

ACCEPTED MANUSCRIPT

Table 6. Contextual comparison with published PSL and Urdu sign-recognition studies.

Study	Recognition approach and context	Reported accuracy	Consideration when interpreting the comparison
Arooj *et al.* [20]	CNN and SIFT features; Kinect-acquired PSL images	98.74%	Acquisition and feature-processing procedures differ from the present configuration.
Mujeeb *et al.* [21]	Neural-network-based web application addressing static and dynamic PSL recognition	99.8% for static recognition	The application and evaluation setting differ; the result is not a comparison on common partitions.
Das *et al.* [22]	UrSL-CNN; approximately 1,500 Urdu sign images	95%	Class-wise metrics are reported. Cross-validation described for conventional baselines should not be conflated with the CNN result.
Present study	Frozen ResNet50–EfficientNetB0 feature fusion; 38 static PSL classes	97.15% mean accuracy	Five-fold internal validation; external and signer-independent generalization remain unestablished.

interpretation of accuracy. In particular, an image-based partition does not establish generalization to unseen signers, and a single split and a cross-validation summary provide different forms of evidence. These factors were not controlled across the cited studies, so the causes of their numerical differences cannot be determined from published accuracy alone. The contribution of the present study is therefore the comparison of five specified configurations within its reported internal-validation procedure, accompanied by class-wise analysis and explicit discussion of methodological limitations.

3.7. Parameter and Computational-Cost Considerations

The performance improvement of Hybrid-ResEff is accompanied by additional model capacity and feature-extraction cost. For a classification head with input dimension d, hidden width h, and 38 output classes, the number of trainable parameters, including biases, are:

$$P_{head} = (d + 1)h + (h + 1) \times 38 \quad (9)$$

$$M_{head} = dh + 38h \quad (10)$$

The matrix multiplications in its two Dense layers require multiply–accumulate operations per image. This count excludes bias additions, activation functions, softmax, pooling, and operations within the pretrained backbones (see Table 7).

Because the pretrained backbones are frozen, these head parameters constitute the trainable parameters of the three configurations. Hybrid-ResEff contains 2,379,264 more trainable parameters than the standalone ResNet50 configuration, corresponding to approximately 3.23 times its classifier parameter count. It additionally requires storage and forward computation for the EfficientNetB0 feature extractor. If P_E denotes the parameter count of that frozen extractor, the total parameter increase over the ResNet50 configuration is therefore $P_E + 2{,}379{,}264$.

Freezing the backbones avoids updating their weights but does not eliminate their forward-pass computation or storage requirements. Although concatenation introduces no trainable parameters, the concatenated representation is processed by a larger classifier. The observed 2.59-percentage-point accuracy increase over ResNet50 must therefore be considered alongside both the additional backbone and the expanded classification head.

These analytical counts characterize part of the architectural cost; they do not measure full-network execution time or peak memory. End-to-end latency, training time, and hardware-specific memory consumption were not established by the reported experiments. Accordingly, the results do not demonstrate that Hybrid-ResEff offers the best performance–cost trade-off for a particular deployment setting.

3.8. Limitations of the Study

The study has additional reproducibility limitations. The complete preprocessing implementation, original software versions, hardware configuration, and global training-seed settings are unavailable. The retained code documents the model architectures and principal training settings but does not permit exact reconstruction of the complete experimental pipeline. It also reuses the initial 20% subset for validation-based training decisions and subsequent evaluation, limiting claims of independent generalization. Furthermore, the common input scaling shown in the available code differs from the documented backbone-specific preprocessing conventions, and the effect of this choice was not isolated. These limitations constrain both reproducibility and interpretation of the model comparison [45].

ACCEPTED MANUSCRIPT

Table 7. Analytical parameter and operation counts for the added classification heads.

Configuration	Input features d	Hidden units h	Trainable head parameters	Dense-layer multiply–accumulate operations per image
ResNet50	2,048	512	1,068,582	1,068,032
EfficientNetB0	1,280	512	675,366	674,816
Hybrid-ResEff	3,328	1,024	3,447,846	3,446,784

The model also worked purely from image-based features. The dataset does include key point representations, but skeletal landmarks and full-body pose information were left out of the final pipeline. There is a reasonable case that folding hand-landmark or pose-based features into the model could help with classes that look visually similar and currently cause the most confusion [43, 46]. On that note, while the proposed model narrowed class-level performance gaps, certain classes still returned noticeably lower F1-scores a sign that subtle finger-position differences and inter-class similarity have not been fully solved. Real-time deployment, latency testing, and evaluation against external PSL datasets were also outside the scope of this work. These are natural next steps, particularly for anyone looking to build this into a mobile or web-based assistive application.

One further gap is worth flagging: the study did not run a full preprocessing ablation. Every model shared the same preprocessing pipeline, so the individual contributions of CLAHE, skin segmentation, and edge detection were never isolated experimentally. The results support the pipeline taken as a whole, but they say nothing definitive about what each step contributes on its own. Future work should address this directly testing Hybrid-ResEff without preprocessing, then with CLAHE alone, then with CLAHE and segmentation combined, and finally with the complete pipeline in place.

The reported experiments do not isolate the effects of the 1024-unit hidden-layer width or the dropout rate and placement. Consequently, the results characterize the complete evaluated configuration and do not establish the optimality of these individual architectural choices. Although standalone backbone comparisons were performed, a controlled ablation removing individual fusion branches while keeping the classifier architecture identical was not conducted.

4. CONCLUSIONS

This study evaluated Hybrid-ResEff, a feature-fusion configuration combining frozen ImageNet-pretrained ResNet50 and EfficientNetB0 representations for recognition of 38 static PSL alphabet classes. Among the five evaluated configurations, Hybrid-ResEff achieved the highest reported mean accuracy of 0.9715 ± 0.0026 and weighted F1-score of 0.9659 ± 0.0025. Its accuracy exceeded the standalone ResNet50 and EfficientNetB0 results by 2.59 and 2.80 percentage points, respectively. Reported class-wise F1-scores ranged from 0.9250 to 0.9970. These findings support further investigation of the combined configuration under limited-data conditions. However, the evaluation provides internal evidence from one dataset, and the comparisons do not isolate feature complementarity from differences in classifier capacity. The improvement also introduces additional trainable parameters and feature-extraction computation. Independent dataset and signer evaluation, controlled component comparisons, and deployment measurements are needed to determine whether the observed advantage persists beyond the reported experimental setting.

5. DATA AVAILABILITY STATEMENT

The dataset used in this study was based on a publicly available Pakistan Sign Language dataset for Urdu alphabet recognition and was curated into a 38-class static PSL alphabet dataset for the present experiments. The dataset contained approximately 1,480 RGB images, with about 40 samples per class, and was used for image-based recognition after resizing, normalization, CLAHE enhancement, skin segmentation, and edge-detection preprocessing. The original public dataset is available from Mendeley Data published in https://doi.org/10.1016/j.dib.2021.107021 [15]. The processed 38-class dataset version, preprocessing details, and implementation materials can be made available from the corresponding author upon reasonable request, subject to the original dataset license and repository access conditions.

6. ACKNOWLEDGEMENTS

The authors gratefully acknowledge the researchers who developed and made the Pakistan Sign Language Urdu alphabet dataset publicly available. Access to this dataset made the present experimental evaluation possible. The authors also appreciate the academic environment provided by the Faculty of Computer Science and Information Technology, Superior University Lahore, Pakistan, during the course of this research..

7. ETHICAL STATEMENT

No new human participants or animals were recruited,

ACCEPTED MANUSCRIPT

contacted, or subjected to experimental procedures in this study. The research was based entirely on secondary computational analysis of a previously published and publicly available Pakistan Sign Language image dataset. Therefore, no additional ethical approval was required for the present study. The dataset was used in accordance with the conditions of the original public repository.

8. CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

9. FUNDING STATEMENT

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

10. AUTHORSHIP CONTRIBUTION STATEMENT

Muazzam Ali: Conceptualization, methodology, model development, implementation, experimentation, formal analysis, visualization, interpretation of results, and preparation of the original manuscript. Aqdas: Data preparation, validation, literature review, interpretation of findings, and critical review and revision of the manuscript. M. Usman Hashmi: Supervision, methodological guidance, validation of the research approach, interpretation of results, and critical review and editing of the manuscript. Amna Riaz: Contributed to reviewing and improving the manuscript, analyzing the findings, and enhancing the overall clarity of the work. Rizwana Bibi: Assisted in revising manuscript sections, improving technical explanations, and refining the presentation of the research. Sidra Siddiqui: Contributed through critical review, improving the structure of the manuscript, and supporting the final refinement of the revised submission. All authors contributed to the intellectual development of the work, reviewed the final manuscript, approved it for submission, and accept responsibility for the integrity of the study.

11. DECLARATION

The authors declare that the results presented in this article are original and have not been published or submitted elsewhere. All authors have reviewed and approved the final manuscript and its submission. If the article is accepted, the authors agree to assign copyright to the Pakistan Academy of Sciences. Appropriate citations have been provided for material from other sources, and any required permissions will be obtained. Generative AI tools, including ChatGPT, were used only for English-language refinement, readability, and editorial improvement. All scientific content, results, analyses, and conclusions were independently reviewed and verified by the authors, who remain fully responsible for the manuscript.

12. REFERENCES

1. World Health Organization. Deafness and hearing loss. *World Health Organization, Geneva, Switzerland* (2026). https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss (accessed 10 June 2026).
2. United Nations. International Day of Sign Languages, 23 September. *United Nations* (2017). https://www.un.org/en/observances/sign-languages-day (accessed 10 June 2026).
3. W.C. Stokoe. Sign Language Structure: An Outline of the Visual Communication Systems of the American Deaf. *Journal of Deaf Studies and Deaf Education* 10(1): 3-37 (2005). https://doi.org/10.1093/deafed/eni001
4. W. Sandler and D. Lillo-Martin (Eds.). Sign Language and Linguistic Universals. *Cambridge University Press* (2006). https://doi.org/10.1017/CBO9781139163910
5. D. Bragg, O. Koller, M. Bellard, L. Berke, P. Boudreault, A. Braffort, N. Caselli, M. Huenerfauth, H. Kacorri, T. Verhoef, C. Vogler, and M.R. Morris. Sign language recognition, generation, and translation: an interdisciplinary perspective. *Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS '19, October 28–30, 2019, Pittsburgh, PA, USA* pp. 16-31 (2019). https://doi.org/10.1145/3308561.3353774
6. M.A. Ahmed, B.B. Zaidan, A.A. Zaidan, and M.M. Salih. A review on systems-based sensory gloves for sign language recognition: state of the art between 2007 and 2017. *Sensors* 18(7): 2208 (2018). https://doi.org/10.3390/s18072208
7. M.J. Cheok, Z. Omar, and M.H. Jaward. A review of hand gesture and sign language recognition techniques. *International Journal of Machine Learning and Cybernetics* 10: 131-153 (2019). https://doi.org/10.1007/s13042-017-0705-5
8. M. Al-Qurishi, T. Khalid, and R. Souissi. Deep learning for sign language recognition: current techniques, benchmarks, and open issues. *IEEE*

ACCEPTED MANUSCRIPT

Access 9: 126917-126951 (2021). https://doi.org/10.1109/ACCESS.2021.3110912

9. M. Madhiarasan and P.P. Roy. A comprehensive review of sign language recognition: different types, modalities, and datasets. *arXiv preprint arXiv*:2204.03328 (2022). https://doi.org/10.48550/arXiv.2204.03328
10. K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, *Las Vegas, NV, USA* pp. 770-778 (2016). https://doi.org/10.1109/CVPR.2016.90
11. M. Tan and Q.V. Le. EfficientNet: rethinking model scaling for convolutional neural networks. *Proceedings of the 36th International Conference on Machine Learning, Long Beach, California* pp. 6105-6114 (2019). https://proceedings.mlr.press/v97/tan19a.html (accessed 11 June 2026).
12. Y. Zhang, B. Kang, B. Hooi, S. Yan, and J. Feng. Deep Long-Tailed Learning: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45(9): 10795-10816 (2023). https://doi.org/10.1109/TPAMI.2023.3268118
13. H. He and E.A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering* 21(9): 1263-1284 (2009). https://doi.org/10.1109/TKDE.2008.239
14. J.M. Johnson and T.M. Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data* 6: 27 (2019). https://doi.org/10.1186/s40537-019-0192-5
15. A. Imran, A. Razzaq, I.A. Baig, A. Hussain, S. Shahid, and T.U. Rehman. Dataset of Pakistan Sign Language and automatic recognition of hand configuration of Urdu alphabet through machine learning. *Data in Brief* 36: 107021 (2021). https://doi.org/10.1016/j.dib.2021.107021
16. H. Zahid, M. Rashid, S. Hussain, F. Azim, S.A. Syed, and A. Saad. Recognition of Urdu sign language: a systematic review of the machine learning classification. *PeerJ Computer Science* 8: e883 (2022). https://doi.org/10.7717/peerj-cs.883
17. F. Shah, M.S. Shah, W. Akram, A. Manzoor, R.O. Mahmoud, and D.S. Abdelminaam. Sign language recognition using multiple kernel learning: a case study of Pakistan Sign Language. *IEEE Access* 9: 67548-67558 (2021). https://doi.org/10.1109/ACCESS.2021.3077386
18. M.S. Mirza, S.M. Munaf, F. Azim, S. Ali, and S.J. Khan. Vision-based Pakistani sign language recognition using bag-of-words and support vector machines. *Scientific Reports* 12: 21325 (2022). https://doi.org/10.1038/s41598-022-15864-6
19. H.M. Hamza and A. Wali. Pakistan sign language recognition: leveraging deep learning models with limited dataset. *Machine Vision and Applications* 34: 71 (2023). https://doi.org/10.1007/s00138-023-01429-8
20. S. Arooj, S. Altaf, S. Ahmad, H. Mahmoud, and A.S.N. Mohamed. Enhancing sign language recognition using CNN and SIFT: a case study on Pakistan sign language. *Journal of King Saud University – Computer and Information Sciences* 36(2): 101934 (2024). https://doi.org/10.1016/j.jksuci.2024.101934
21. A.A. Mujeeb, A.H. Khan, S. Khalid, M.S. Mirza, and S.J. Khan. A neural-network based web application on real-time recognition of Pakistani sign language. *Engineering Applications of Artificial Intelligence* 135: 108761 (2024). https://doi.org/10.1016/j.engappai.2024.108761
22. K. Das, F. Abid, J. Rasheed, Kamlish, T. Asuroglu, S. Alsubai, and S. Soomro. Enhancing communication accessibility: UrSL-CNN approach to Urdu sign language translation for hearing-impaired individuals. *Computer Modeling in Engineering & Sciences* 141(1): 689-711 (2024). https://doi.org/10.32604/cmes.2024.051335
23. S. Naeem, H. Salam, and M.A. Uddin. Pakistani word-level sign language recognition based on deep spatiotemporal network. *Proceedings of the AAAI Symposium Series* 6(1): 119-126 (2025). https://doi.org/10.1609/aaaiss.v6i1.36042
24. J. Shin, A.S.M. Miah, Y. Akiba, K. Hirooka, N. Hassan, and Y.S. Hwang. Korean sign language alphabet recognition through the integration of handcrafted and deep learning-based two-stream feature extraction approach. *IEEE Access* 12: 68303-68318 (2024). https://doi.org/10.1109/ACCESS.2024.3399839
25. J.J. Bird, A. Ekárt, and D.R. Faria. British Sign Language recognition via late fusion of computer vision and Leap Motion with transfer learning to American Sign Language. *Sensors* 20(18): 5151 (2020). https://doi.org/10.3390/s20185151
26. M.A. Hamza, A. Subahi, N.A. Alghanmi, and N. Nemri. Deep fusion based transfer learning with bald eagle search algorithm for sign language recognition. *Scientific Reports* 15: 32752 (2025). https://doi.org/10.1038/s41598-025-10660-4
27. K. Zuiderveld. Contrast Limited Adaptive Histogram Equalization. In: Graphics Gems IV. Chapter VIII.5. P.S. Heckbert (Ed.). *Academic Press* pp. 474-485 (1994). https://doi.org/10.1016/B978-0-12-336156-

ACCEPTED MANUSCRIPT

1.50061-6

28. P. Kakumanu, S. Makrogiannis, and N. Bourbakis. A survey of skin-color modeling and detection methods. *Pattern Recognition* 40(3): 1106-1122 (2007). https://doi.org/10.1016/j.patcog.2006.06.010
29. J. Canny. A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine *Intelligence PAMI*-8(6): 679-698 (1986). https://doi.org/10.1109/TPAMI.1986.4767851
30. S. Bates, T. Hastie, and R. Tibshirani. Cross-validation: what does it estimate and how well does it do it? *Journal of the American Statistical Association* 119(546): 1434-1445 (2024). https://doi.org/10.1080/01621459.2023.2197686
31. S.J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 22(10): 1345-1359 (2010). https://doi.org/10.1109/TKDE.2009.191
32. J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? *Advances in Neural Information Processing Systems 27 (NIPS '14), NIPS Foundation* (2014). https://doi.org/10.48550/arXiv.1411.1792
33. K. Bayoudh. A survey of multimodal hybrid deep learning for computer vision: Architectures, applications, trends, and challenges. *Information Fusion* 105: 102217 (2024). https://doi.org/10.1016/j.inffus.2023.102217
34. D.P. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations* pp. 1-15 (2015). https://arxiv.org/abs/1412.6980
35. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15: 1929-1958 (2014). https://www.jmlr.org/papers/v15/srivastava14a.html
36. L. Prechelt. Early stopping — but when? In: Neural Networks: Tricks of the Trade. G.B. Orr, and K-R. Müller (Eds.). *Springer, Berlin, Heidelberg* pp. 55-69 (1998). https://doi.org/10.1007/3-540-49430-8_3
37. D.M.W. Powers. Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness and Correlation. *Journal of Machine Learning Technologies* 2(1): 37-63 (2011). https://doi.org/10.48550/arXiv.2010.16061
38. M. Sokolova and G. Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing & Management* 45(4): 427-437 (2009). https://doi.org/10.1016/j.ipm.2009.03.002
39. M.K. Vidhyalakshmi, K. Jagadeesh, B.S. Dharmaraj, N.P. Sankar, and R.G. Kumar. Indian Sign Language Recognition using transfer learning with Efficient Net. *Smart Science* 12(2): 269-280 (2024). https://doi.org/10.1080/23080477.2024.2320956
40. M.A. Hamza, A. Subahi, N.A. Alghanmi, and N. Nemri. Deep fusion based transfer learning with bald eagle search algorithm for sign language recognition to assist individuals with hearing and speech impairments. *Scientific Reports* 15: 32752 (2025). https://doi.org/10.1038/s41598-025-10660-4
41. T. M. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data* 6: 60 (2019). https://doi.org/10.1186/s40537-019-0197-0
42. J. Qin and M. Wang. Sign language recognition based on dual-channel star-attention convolutional neural network. *Scientific Reports* 15: 27685 (2025). https://doi.org/10.1038/s41598-025-13625-9
43. A. Boulesnane, L. Bellil, and M.G. Ghiri. A hybrid CNN-random forest model with landmark angles for real-time Arabic sign language recognition. *Neural Computing and Applications* 37: 2641-2662 (2025). https://doi.org/10.1007/s00521-024-10729-7
44. T.Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal Loss for Dense Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42(2): 318-327 (2020). https://doi.org/10.1109/TPAMI.2018.2858826
45. M. Hasan, B.K. Paul, N. Islam, and R. Mostafiz. Advancing real-time sign language detection for deaf and hearing-impaired communities: a customized YOLOv8 approach with tailored annotations in computer vision. *BMC Artificial Intelligence* 1: 11 (2025). https://doi.org/10.1186/s44398-025-00010-9
46. R. Yayla, H. Üçgün, and M. Abbas. Pose-Based Static Sign Language Recognition with Deep Learning for Turkish, Arabic, and American Sign Languages. *Sensors* 26(2): 524 (2026). https://doi.org/10.3390/s26020524

ACCEPTED MANUSCRIPT