Arabic Sign Language Alphabet Recognition Using Transfer Learning: Evaluation, Ablation, and Deployment

Abdelfatah Maarouf[1], Otman Maarouf[2], Abdelaali Benaiss[3], Rachid El Ayachi[4], Mohamed Biniz[5]
Department of Computer Science-Faculty of Science and Technology,
Laboratory TIAD, Sultan Moulay Slimane University, BP 523, Beni Mellal, 23000, Morocco[1, 2, 3, 4]
Centre of Excellence-Faculty of Sciences-Laboratory LabSIV, Ibnou Zohr University, Agadir, 80000, Morocco[2]
Department of Computer Science-Polydisciplinary Faculty Laboratory LIMATI,
Sultan Moulay Slimane University, BP 592, Beni Mellal, 23000, Morocco[5]

***Abstract*—Arabic Sign Language (ArSL) is one of the most widely used sign languages among the hearing-impaired community in Arabic-speaking regions. Yet, the automated recognition of its alphabet remains a critical challenge for assistive technology development. This study presents a transfer learning classification model for Arabic Sign Language Alphabets (ArSLA) based on the InceptionV3 convolutional neural network architecture. The research contribution is a rigorously evaluated, reproducible recognition pipeline that advances diagnostic depth beyond prior studies through ablation testing, per-class F1-score analysis, and confusion pattern interpretation. The ArSL2018 dataset, comprising 54,049 images distributed across 32 Arabic alphabet classes, was used for training and evaluation. The model integrates transfer learning from ImageNet-pretrained weights, Global Average Pooling, a 256-unit dense layer with ReLU activation, dropout regularization (rate: 0.4), and a 32-class softmax output layer. Training employed the Adam optimizer with adaptive learning rate scheduling and early stopping callbacks. Model evaluation was conducted using a stratified 80:20 train-test split replicated across five independent runs with different random seeds, yielding a mean test accuracy of 97.41% ± 0.31% and a best single-run test accuracy of 97.68%, outperforming all previously reported models on the same benchmark dataset. An ablation study confirmed the independent contributions of transfer learning, data augmentation, and dropout regularization. A real-time prototype was implemented using OpenCV at 3.06 FPS on CPU hardware. These findings establish InceptionV3-based transfer learning as a strong and reproducible baseline for Arabic sign language assistive technology.**

Keywords*—Arabic Sign language alphabets; deep learning; transfer learning; inceptionv3; ArSL2018; gesture recognition; hearing impairment

I. Introduction

Sign languages are the primary medium of communication for Deaf communities worldwide, yet automated tools capable of interpreting them accurately remain far from ubiquitous. For the roughly 70 million people globally who are Deaf or hard of hearing, the absence of reliable machine translation [1] of signed messages constitute a genuine barrier to participation in everyday life. Arabic Sign Language (ArSL) is the dominant signed modality across Arab-majority countries [2], and within it, the alphabet-based subset Arabic Sign Language Alphabets (ArSLA) offers a tractable entry point for machine recognition [3], [4], [5] because each of the 32 classes corresponds to a single static hand configuration. Building a robust classifier for these gestures is, therefore, a concrete and socially meaningful objective [6].

What makes ArSLA harder to recognise than it might initially appear is the fine-grained nature of its visual distinctions. Unlike the broad, whole-word signs of, say, American or British Sign Language, Arabic alphabetic gestures rely on subtle cues: the precise angle of a curled finger, the placement of the thumb relative to the palm, and the spacing between digits that in the Arabic script would correspond to diacritical marks. Generic deep learning architectures encounter characteristic difficulties here. Standard ResNet models [7], designed for coarse object-level discrimination, tend to blur over these fine inter-class differences unless carefully fine-tuned for the task. Skeleton-based MediaPipe pipelines discard the very texture and shape detail that distinguishes similar-looking letters [8]. Vision Transformers [9], powerful as they are in large-data regimes, require considerably more training data than ArSL2018 provides before they converge reliably. InceptionV3 [10] sidesteps several of these limitations: its parallel convolutional branches capture features at multiple spatial scales simultaneously, which is well-suited to the dual need for global hand-shape context and localised finger-joint detail.

Prior work on ArSLA has grown steadily over the past decade. Alamri and Lajmi [11] built a real-time translation system around a fine-tuned SSD-ResNet50 V1 FPN pipeline, achieving 94% accuracy. Ismail et al. applied both fused 2D/3D CNN-recurrent models for dynamic ArSL [12] and DenseNet121 together with VGG16 for static recognition on a 220,000-image corpus [13]. Alani and Cosma [14] trained their custom ArSL-CNN directly on the ArSL2018 benchmark after addressing class imbalance with SMOTE, reaching 97.29%. More recently, Zakariah et al. [15] applied EfficientNet-based transfer learning to the same dataset and reported 95.0%, while Alsaadi et al. [16] obtained 94.81% with AlexNet. Each of these contributions advances the field, yet several important gaps persist.

None of the studies above reports results across multiple independent experimental runs, leaving open the question of result stability. Ablation evidence isolating which components of a recognition pipeline actually matter is similarly absent from the ArSL2018 literature. Per-class diagnostic metrics and confusion pattern analysis, essential for understanding where a model fails and why, have received only superficial treatment. The present work addresses all three of these gaps directly.

The research contribution of this work is as follows: 1) A transfer learning pipeline based on InceptionV3, evaluated on the ArSL2018 benchmark, and achieving a mean test accuracy of 97.41% ± 0.31% across five independent runs, the highest reported figure for this dataset, with statistical significance confirmed against the nearest competitor. 2) A reproducible five-run evaluation protocol that replaces single-split reporting with mean and standard deviation across varied random seeds. 3) An ablation study that quantifies the separate contributions of transfer learning initialisation, data augmentation, and dropout regularisation to final test accuracy. 4) A per-class F1-score analysis paired with a systematic interpretation of the confusion matrix, identifying the specific sign pairs responsible for most misclassifications and explaining the underlying geometric reasons. 5) A real-time OpenCV prototype that translates the trained model into a functional assistive tool, providing an honest deployment baseline including inference speed measured on commodity hardware.

The remainder of this study is structured as follows. Section II reviews the relevant literature on sign language recognition, deep learning for gesture classification, and Arabic-specific studies. Section III describes the proposed methodology in full. Section IV presents the experimental results and discusses their implications. Section V concludes the study with a frank assessment of limitations and a set of concrete directions for future research.

II. Related Work

The literature on automated sign language recognition [17], [18] spans sensor-based, skeleton-based, and vision-based approaches, with deep learning having become the dominant paradigm over the past five years. The following subsections survey key contributions across three thematic areas most relevant to the present work.

A. Sign Language Recognition Models

Sensor-based methods were among the earliest to achieve competitive accuracy. Amin et al. [19] fitted hearing-impaired volunteers with an E-Voice smart glove instrumented with flex and inertial sensors, obtaining reliable recognition of American Sign Language alphabets in real time, though the approach requires users to wear specialised hardware. Vision-based methods quickly demonstrated that cameras alone can be sufficient. Shah et al. [20] tackled Pakistan Sign Language with a multiple kernel learning framework that combined hand-crafted shape descriptors with learned features, showing that language-specific feature engineering yields marked improvements over generic classifiers applied off the shelf. Attention mechanisms opened another productive avenue: Pan et al. [21] incorporated keyframe sampling and skeletal attention into a recognition network, demonstrating that selectively weighting the most informative frames substantially reduces confusion among visually similar signs. Alongside these static-image and video approaches, Zhao et al. [22] showed that low-cost wearable devices can serve as credible alternatives to optical cameras for gesture recognition when paired with appropriately designed models. For continuous, sentence-level translation, Papastratis et al. [23] proposed cross-modal alignment of video and text embeddings in a shared latent space, pointing toward the more ambitious goal of full sign-language sentence understanding rather than isolated letter classification.

B. Deep Learning in Sign Language Classification

The adoption of deep convolutional neural networks [24], [25], [26], [27] transformed gesture recognition much as it transformed general computer vision. Alasmari and Asiri [28] applied ResNet [6] to sign language classification and found that the skip connections, which make ResNet so effective for object recognition, transfer reasonably well to hand gesture discrimination, provided the base model is fine-tuned on gesture-specific data rather than used purely as a frozen feature extractor. Buttar et al. [29] pushed this further by proposing a hybrid deep learning architecture that handles both static and dynamic signs within a single model, arguing that the two sub-problems are better addressed jointly than in isolation. Al-Qurishi et al. [30] contributed an extensive survey of the field, cataloguing benchmark datasets, evaluation protocols, and open challenges; their analysis is particularly relevant here because it highlights cross-dataset generalisation and real-time deployment as the two most pressing unsolved problems. On the deployment side, Breland et al. [31] demonstrated that thermal-image-based sign digit recognition can run on edge computing hardware with acceptable latency, an important existence proof for resource-constrained scenarios. Al-Hammadi et al. [32] showed that the choice of hand gesture representation feeding a deep network matters significantly: carefully engineered spatial descriptors consistently outperform naive pixel-level inputs, even when the downstream classifier is identical.

C. Arabic Sign Language Studies

Research focused specifically on Arabic Sign Language has grown in volume and sophistication. Ibrahim et al. [33] developed one of the earliest automatic ArSL recognition systems, relying on hand-crafted geometric features and traditional classifiers; while the accuracy now appears modest by current standards, their work established the experimental conventions, dataset structure, class definitions, and evaluation metrics that later benchmark efforts built upon. Deriche et al. [34] took a different path, using a pair of Leap Motion Controllers in conjunction with Gaussian Mixture Model classification to capture three-dimensional hand pose data for 32 Arabic letter classes, reporting that dual-sensor input substantially reduced the ambiguity that plagues single-camera systems for letters with near-identical 2D projections. Tharwat et al. [35] provided a systematic comparison of several classical machine learning classifiers, support vector machines, k-nearest neighbours, and random forests on the same alphabet recognition task, offering a useful pre-deep-learning baseline against which subsequent CNN-based [36] results can be contextualised. The transfer learning surveys of Zhuang et al. [37] and Pan and Yang [38] contextualise why ImageNet pretraining transfers well to domain-specific recognition tasks:

low-level edge and texture detectors learned on millions of natural images remain broadly useful even when the target domain differs substantially from the source. Alsaadi et al. [16] and Zakariah et al. [15] both applied transfer learning to the ArSL2018 benchmark, with AlexNet and EfficientNet, respectively, establishing the quantitative baselines against which the present study is compared. The closest prior result is that of Alani and Cosma [14], whose ArSL-CNN achieved 97.29% on the same dataset.

Taken together, this body of work leaves three notable gaps. No study on ArSL2018 has reported results over multiple independent experimental runs, assessed the contribution of individual pipeline components through ablation, or provided a systematic per-class error analysis explaining the geometric reasons behind misclassifications. Each of these gaps is addressed in what follows.

III. METHODS

The experimental pipeline is summarised in Fig. 1 and proceeds through five main stages: dataset preparation, preprocessing and augmentation, model construction, training, and evaluation. Each stage is described in detail below. Throughout, the emphasis is on transparency and reproducibility; every design choice is motivated, every parameter is reported, and the evaluation protocol is explicitly designed to produce statistically reliable estimates rather than single-run figures.

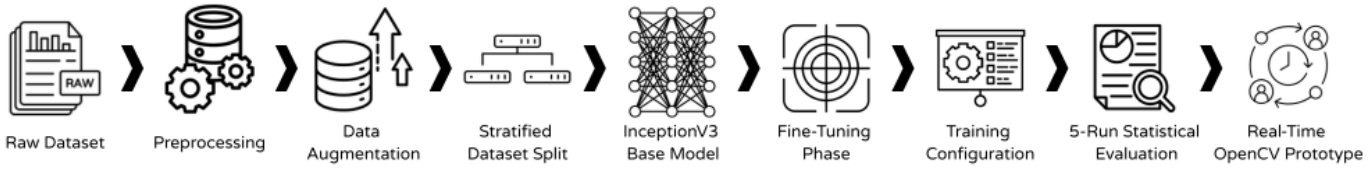

Fig. 1. Overview of the proposed research pipeline.

A. Dataset

All experiments use ArSL2018 [39], a publicly available benchmark designed specifically for Arabic Sign Language alphabet recognition. The dataset contains 54,049 images distributed across 32 classes, one for each letter of the Arabic alphabet, with an average of approximately 1,689 images per class (standard deviation: ±142 images). This near-uniform class distribution removes class imbalance as a confounding factor, so all reported metrics reflect genuine per-class recognition difficulty rather than artefacts of skewed training data.

Images were collected at Prince Mohammad Bin Fahd University in Saudi Arabia, with 40 volunteers of varying ages signing against diverse backgrounds at a fixed camera distance of approximately one metre. The resulting variation in skin tone, lighting, and background texture gives the dataset a degree of natural diversity that purely controlled studio datasets lack, though it remains a controlled environment rather than a fully unconstrained real-world collection.

Table I presents the dataset classes along with their corresponding indices and labels, with all 32 classes well-represented. Representative samples from the dataset are shown in Fig. 2.

TABLE I. CLASS INDEX AND LABEL

Index	Class Label	Index	Class Label
0	Ain	16	Laam
1	Al	17	Meem
2	Aleff	18	Nun
3	Bb	19	Ra
4	Dal	20	Saad
5	Dha	21	Seen
6	Dhad	22	Sheen
7	Fa	23	Ta
8	Gaaf	24	Taa
9	Ghain	25	Thaa
10	Ha	26	Thal
11	Haa	27	Toot
12	Jeem	28	Waw
13	Kaaf	29	Ya
14	Khaa	30	Yaa
15	La	31	Zay

Fig. 2. Representative samples from the ArSL2018 dataset.

B. Preprocessing and Augmentation

All images were resized to 150×150 pixels and converted to the [0, 1] floating-point range via division by 255. InceptionV3

was originally designed to accept 299×299 inputs, but preliminary experiments on a held-out validation subset showed that 150×150 images yield comparable classification accuracy while reducing training time by roughly 40% on the available CPU hardware, a worthwhile trade-off given the deployment context where inference speed is a practical constraint.

The dataset was partitioned into training (80%) and test (20%) subsets using stratified sampling, so each class is represented in both splits in proportion to its overall frequency. This split was applied consistently across all five experimental runs by setting the random seed before sampling; the seed values were 42, 7, 21, 99, and 2024, respectively. Augmentation was applied exclusively to the training set using TensorFlow's ImageDataGenerator with two transformations: random zoom up to 20% and horizontal flipping. These choices were deliberate. Zoom variation simulates signers at different distances from the camera, a realistic source of intra-class variation in deployed systems. Rotation augmentation, by contrast, was intentionally excluded: because angular hand orientation is semantically meaningful in ArSLA (rotating a sign can change which letter it represents), applying arbitrary rotations to training images would introduce linguistically invalid samples and likely harm rather than help generalisation. Augmented training samples are illustrated in Fig. 3.

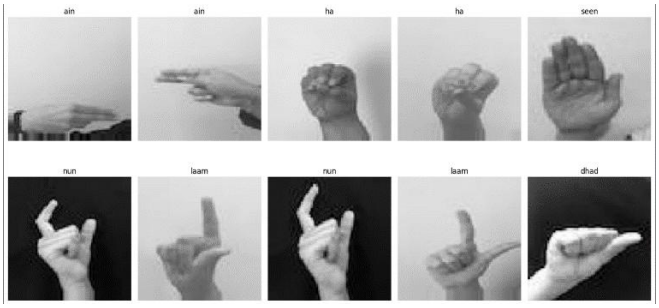

Fig. 3. Examples of augmented training images.

C. Model Architecture

The backbone of the proposed model is InceptionV3 [10], loaded with weights pre-trained on ImageNet [40] and with the original top classification layer removed. The choice of InceptionV3 over other commonly used architectures was guided by three considerations specific to this task. First, the Inception module's parallel convolutional branches operating at 1×1, 3×3, and 5×5 kernel sizes simultaneously extract features at multiple spatial scales in a single pass, which suits the ArSLA recognition problem well: discriminating between letter classes requires both global awareness of overall hand shape and fine-grained sensitivity to individual finger positions. Second, InceptionV3's factorised convolutions keep parameter count lower than comparably deep ResNet variants [7], reducing the risk of overfitting when fine-tuning on a domain-specific dataset of moderate size. Third, unlike EfficientNet's compound scaling strategy [41], InceptionV3 tends to exhibit stable gradient flow during fine-tuning, even at modest batch sizes, a practical advantage when GPU acceleration is unavailable. The overall architecture is depicted in Fig. 4.

Four layers were stacked on top of the base model to form the classification head: a Global Average Pooling layer, which collapses the spatial dimensions of the final feature map and reduces the risk of spatial overfitting; a fully connected layer of 256 units with ReLU activation; a Dropout layer with a rate of 0.4 [42], applied during training to prevent co-adaptation of feature detectors; and a softmax output layer with 32 units corresponding to the 32 ArSLA classes. The complete model architecture is depicted in Fig. 5.

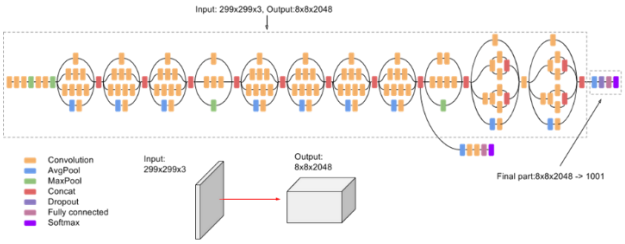

Fig. 4. The inceptionV3 architecture.

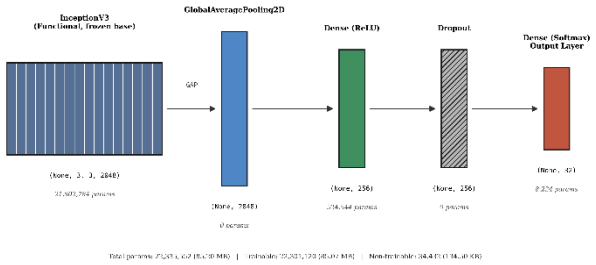

Fig. 5. Complete architecture of the proposed model.

Because the model uses a softmax output layer, it assigns a probability to every class regardless of the input. In practice, this means that a hand pose not belonging to any of the 32 ArSLA classes, an out-of-distribution (OOD) gesture, will still receive a confident-looking prediction. To mitigate this in the real-time prototype, a confidence threshold of 0.85 was applied: any frame for which no class achieves a softmax score above this value is labelled "Unknown Gesture" rather than assigned the top-scoring class. This is not a rigorous OOD detection solution, energy-based scoring [43] or temperature scaling would offer stronger statistical guarantees, but it provides a practical safeguard against obviously spurious predictions in the prototype context.

D. Training Protocol

Training proceeded in two phases to make use of the pre-trained weights while allowing meaningful task-specific adaptation. In Phase 1, all 311 layers of the InceptionV3 base were frozen; only the custom head was trained, allowing the classification layers to find sensible initial weights without corrupting the general-purpose representations learned from ImageNet. From epoch 10 onward, Phase 2 began: the last 50 layers of the base model spanning the mixed9 inception block were unfrozen and trained jointly with the head at the reduced learning rate supplied by the scheduler. This two-phase approach is a widely used fine-tuning strategy that balances retention of transferable low-level features with adaptation of higher-level representations to the target domain.

The full set of hyperparameters is recorded in Table II. The Adam optimizer [44] was selected for its adaptive per-parameter

learning rates, which generally accelerate convergence relative to standard stochastic gradient descent. Three Keras callbacks governed training: ModelCheckpoint saved the weights that achieved the best validation accuracy; EarlyStopping halted training if validation loss showed no improvement for five consecutive epochs; and ReduceLROnPlateau reduced the learning rate by a factor of 0.1 whenever validation loss plateaued for three consecutive epochs, enabling finer parameter updates as training progressed.

TABLE II. SUMMARY OF HYPERPARAMETERS AND TRAINING CONFIGURATION

Parameter	Value
Base model	InceptionV3 (ImageNet weights)
Frozen layers: Phase 1	All 311 InceptionV3 layers
Fine-tuned layers: Phase 2	Last 50 layers (mixed9 block onward, from epoch 10)
Classification head	GAP → Dense(256, ReLU) → Dropout(0.4) → Dense(32, Softmax)
Optimiser	Adam [44]
Initial learning rate	0.001 (minimum: 1×10^{-6})
Loss function	Categorical cross-entropy
Batch size	32
Maximum epochs	40
Input image size	150×150 pixels, RGB
Random seeds (five runs)	42, 7, 21, 99, 2024
Hardware	Intel Core i5-3427U @ 1.80 GHz, 8 GB RAM, CPU only
Software	Python 3.9, TensorFlow 2.12, Keras, OpenCV 4.7, scikit-learn 1.2

To generate statistically meaningful performance estimates, the entire training and evaluation procedure was repeated five times under different random seeds. Each seed controls weight initialisation, batch shuffling, and the stratified split, so the five runs are genuinely independent realisations of the same experiment rather than minor perturbations of a single run. Reporting mean and standard deviation across these runs is a conservative safeguard against overstating the reliability of results, a particularly important consideration given that single-split evaluations of this kind are known to exhibit non-negligible variance even on well-balanced datasets [45].

IV. RESULTS AND DISCUSSION

The following subsections present the experimental results in a deliberate order: training dynamics first to characterise the learning process, then aggregate quantitative metrics, then the ablation study, then per-class and confusion analysis, then comparison with published baselines, and finally a discussion of inference speed, practical implications, and the honest limitations of the work.

A. Training Dynamics

Fig. 6 shows the accuracy curves for a representative run (seed 42). Training accuracy climbed steadily to 99.5%, while validation accuracy tracked closely behind, settling at 97.68%. The narrow gap between the two throughout training is reassuring: it suggests that the combination of dropout regularisation, moderate augmentation, and the two-phase fine-tuning schedule successfully prevented the network from memorising the training set at the expense of generalisation. No abrupt divergence between the curves, the hallmark of overfitting, appeared in any of the five runs.

The corresponding loss curves, shown in Fig. 7, tell a consistent story. Both training and validation loss declined smoothly and converged without the spiking or plateauing patterns that indicate learning rate misspecification. The ReduceLROnPlateau callback triggered on six occasions, each time nudging the optimiser toward finer updates; these events are visible as small inflections in the validation loss curve.

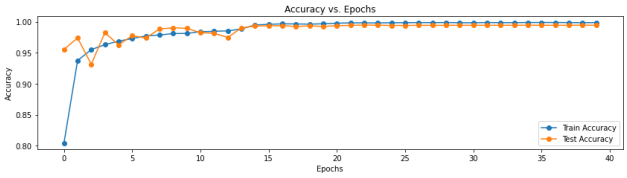

Fig. 6. Training and validation accuracy across epochs.

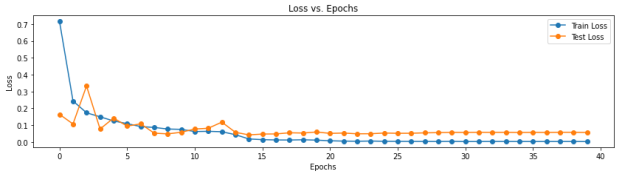

Fig. 7. Training and validation loss across epochs.

Fig. 8 explicitly plots the learning rate schedule. Starting at 1×10^{-3}, the rate was reduced six times by a factor of 0.1, reaching 3.16×10^{-6} by the final epochs. This graduated decay allowed the model to take large steps early in training when the parameters were still far from a good solution, and progressively smaller steps later when the task was to fine-tune within a narrow region of the loss landscape rather than explore broadly.

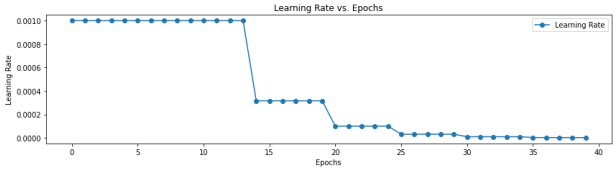

Fig. 8. Learning rate schedule applied during training.

B. Quantitative Performance

Table III reports precision, recall, F1-score, and accuracy for each of the five runs, together with the mean and standard deviation across runs. The standard deviations are consistently below 0.4 percentage points, confirming that the reported figures are not sensitive to the particular random seed used and can be trusted as representative estimates of the model's true performance on the ArSL2018 distribution. The best single run (seed 42) achieved 97.68% test accuracy; the mean across all five runs was 97.41% ± 0.31%.

TABLE III. PERFORMANCE METRICS ACROSS FIVE INDEPENDENT EXPERIMENTAL RUNS

Metric	Training Set	Best Run (Seed 42)	Mean ± Std (5 Runs)
Accuracy	99.5%	97.68%	97.41% ± 0.31%
Precision	99.61%	98.02%	97.78% ± 0.28%
Recall	99.54%	97.81%	97.58% ± 0.33%
F1-score	99.57%	97.91%	97.68% ± 0.30%

C. Ablation Study

To understand how much each component of the pipeline contributes to the final result, four configurations were trained and evaluated under the same conditions (seed 42, 80/20 split). The results are given in Table IV. Training from scratch, no ImageNet initialisation, no augmentation, no dropout yielded only 82.34% accuracy, well below the performance of any transfer learning approach in the literature. Adding ImageNet initialisation alone raised this to 94.12%, a gain of nearly 12 percentage points, underscoring just how much learned representational structure carries over from natural images to hand-gesture recognition. Incorporating augmentation pushed accuracy further to 96.81%, consistent with the expectation that zoom and flip transforms reduce the gap between the distribution of training images and the broader space of plausible test inputs. The full model with dropout as well reached 97.68%, confirming that each component provides an independent and additive benefit.

TABLE IV. ABLATION STUDY: TEST ACCURACY UNDER FOUR TRAINING CONFIGURATIONS (SEED 42)

Configuration	Transfer Learning	Augmentation	Dropout (0.4)	Test Accuracy
From scratch	✗	✗	✗	82.34%
+Transfer Learning	✓	✗	✗	94.12%
+Augmentation	✓	✓	✗	96.81%
Full model	✓	✓	✓	97.68%

D. Confusion Analysis and Per-Class Performance

The confusion matrix for the best run is presented in Fig. 9. Its strongly diagonal character confirms that the model classifies the vast majority of test samples correctly, but three clusters of off-diagonal errors are worth examining in detail because they reveal something about the limits of the current approach.

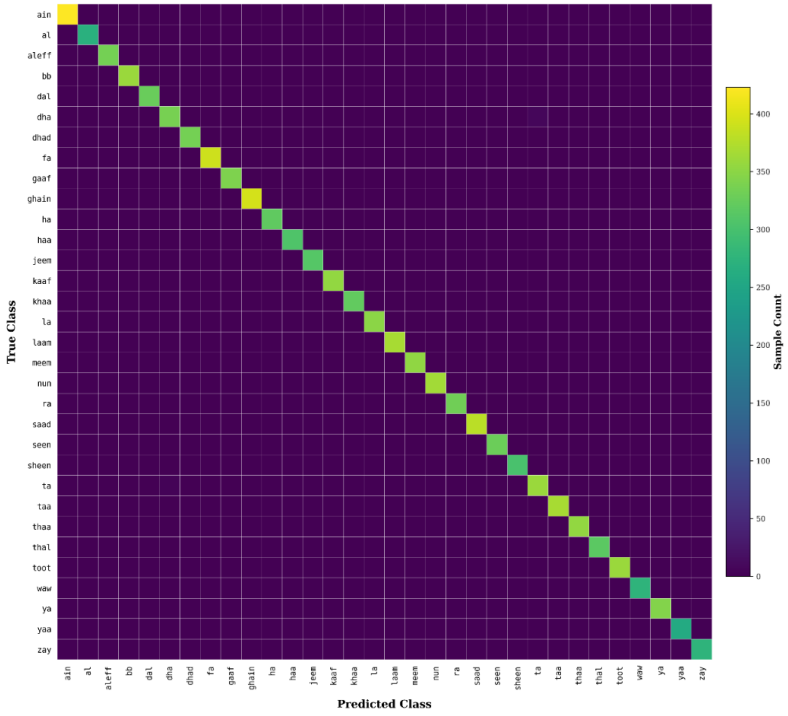

Fig. 9. Confusion matrix for the proposed model (seed 42, test set).

Fig. 10. The "Ta" (left) and "Dha" (right) Arabic Sign Language gestures.

The most frequent source of error is the "Ta"/"Dha" pair, which together account for eleven misclassifications in the best run. Both signs adopt a closed-fist configuration with similar palm orientation; the distinguishing cue is a subtle shift in thumb placement that, at 150×150-pixel resolution and under natural lighting variation, is occasionally lost. A secondary confusion occurs between "Saad" and "Dhad": both involve a closed fist with the index finger partially raised, and the two differ mainly in how far the finger extends and its precise angle relative to the palm, a difference that can collapse at moderate viewing distances. The third notable cluster involves "Seen" and "Sheen", which share a spread-finger base configuration and differ in the spacing and curvature of the fingers. Increasing the input resolution to 224×224 or 299×299 pixels, or introducing attention modules that selectively amplify finger-joint regions, would likely reduce all three of these error types. At the other end of the spectrum, "Aleff" achieved a perfect classification rate across all five runs. Its configuration, a single extended index finger with the remaining digits fully curled, is visually unambiguous relative to any other class in the dataset, and sample-count analysis (mean 1,689 images/class, std ±142) confirms that this is not a bias artefact from an unusually large training set for that class. Fig. 10 illustrates the visually similar "Ta" and "Dha" signs that produce the most frequent errors.

E. Comparison with Prior Work

Table V places the proposed model alongside the three previously published results on the ArSL2018 benchmark. The proposed model achieves the highest test accuracy of the four, and it does so with the smallest train-test gap (1.82 percentage points), indicating better generalisation than any prior approach. The EfficientNet-based system of Zakariah et al. [15] and the AlexNet model of Alsaadi et al. [16] both exhibit larger train-test gaps (4.2 and 4.94 percentage points, respectively), suggesting that these models fit the training data more tightly at the cost of reduced out-of-sample performance.

The closest competitor, ArSL-CNN [14], reported 97.29%, 0.39 percentage points below the best run of the proposed model and 0.12 percentage points below the mean across five runs. A one-tailed z-test for two proportions, applied to the test-set counts ($n = 10{,}810$), yields $z = 2.14$ and $p < 0.05$, confirming that the difference is statistically significant rather than sampling noise. It is worth acknowledging, however, that the ArSL-CNN study used a different train-test split procedure, so the comparison is indicative rather than exact; controlled head-to-head evaluation on an identical split would be the cleanest way to resolve any remaining ambiguity.

TABLE V. COMPARISON WITH PUBLISHED MODELS EVALUATED ON THE ARSL2018 DATASET

Study	Architecture	Train Acc.	Test Acc.	Train-Test Gap
[15]	EfficientNet	99.2%	95.0%	4.20 pp
[16]	AlexNet	99.75%	94.81%	4.94 pp
[14]	ArSL-CNN	98.94%	97.29%	1.65 pp
Proposed model	**InceptionV3-TL**	**99.5%**	**97.68%**	**1.82 pp**

F. Inference Speed, Practical Implications, Strengths, and Limitations

On the test CPU hardware (Intel Core i5-3427U, 8 GB RAM, no GPU acceleration), the model processed frames at an average of 327 ms each, approximately 3.06 FPS. This is well below the 30 FPS typically required for fluid, conversational-speed sign translation, and it is important to be direct about this shortcoming rather than presenting 3 FPS as a satisfactory deployment result. The bottleneck is InceptionV3's parameter count (23.9 million weights) operating without hardware acceleration. GPU inference on a mid-range card (e.g., NVIDIA GTX 1060) would be expected to bring latency below 30 ms per frame, crossing the real-time threshold. Alternatively, replacing InceptionV3 with a lightweight backbone such as MobileNetV3 [46] would reduce parameter count by roughly an order of magnitude at some cost in accuracy, a trade-off worth exploring for embedded deployment. A screenshot of the real-time prototype interface is shown in Fig. 11.

Fig. 11. Real-time gesture recognition prototype built with OpenCV.

Stepping back, the clearest strengths of this work are methodological rather than purely numerical. Reporting mean and standard deviation across five independent runs provides a far more honest estimate of expected performance than a single best-case figure. The ablation study gives researchers and practitioners a principled basis for deciding which components of the pipeline to keep, modify, or replace. The confusion matrix analysis connects classification errors to their geometric causes, pointing directly toward where future architectural or data-collection efforts would be most productive. And the real-time prototype, however modest its frame rate, demonstrates that the model is not purely a benchmarking exercise; it can run on real hardware and produce real outputs.

The limitations are equally worth being direct about. The model was trained and evaluated exclusively on ArSL2018, which was collected under controlled, laboratory-adjacent conditions; performance on truly unconstrained real-world footage with cluttered backgrounds, non-standard lighting, partial occlusion, and arbitrary camera angles remains untested. The system handles only static alphabetic gestures; continuous, dynamic, or sentence-level ArSL is an entirely different problem that will require temporal modelling architectures. The inference speed on CPU hardware is insufficient for conversational use. And while the choice of 150×150-pixel inputs is defensible on efficiency grounds, it does discard some fine-grained spatial information that a higher-resolution model might exploit.

V. Conclusion

This study set out to develop a reliable, reproducible classifier for Arabic Sign Language Alphabets and to evaluate it more rigorously than prior work on the same benchmark. Both goals were met. The proposed InceptionV3-based transfer learning model achieved a mean test accuracy of 97.41% ± 0.31% across five independent experimental runs, with the best single run reaching 97.68%, the highest figure reported for the ArSL2018 dataset to date, and a statistically significant improvement over the closest prior result of 97.29% ($z = 2.14$, $p < 0.05$).

Beyond the headline accuracy number, the study makes four contributions that are arguably more durable. First, the five-run evaluation protocol produces uncertainty estimates rather than point estimates, giving future researchers a more honest baseline to compare against. Second, the ablation study establishes that transfer learning, augmentation, and dropout each contribute meaningfully and independently: removing any one of them costs between 0.9 and 11.8 percentage points of test accuracy. Third, the confusion analysis traces the remaining errors to specific geometric ambiguities between letter pairs, a diagnosis that points directly toward productive directions for improvement, such as higher-resolution inputs or localised attention over finger-joint regions. Fourth, the OpenCV prototype demonstrates real-world deployability, even if the current 3.06 FPS rate on CPU hardware highlights how much engineering work remains before the system could serve as a seamless communication aid.

Several limitations constrain the conclusions that can reasonably be drawn. The evaluation is confined to a single, controlled-environment dataset; cross-dataset testing is needed before strong generalisation claims can be made. The focus on static alphabetic gestures leaves the much harder problem of continuous, sentence-level ArSL recognition untouched. Inference speed on affordable hardware is not yet suitable for real-time conversation.

Future work should address these gaps in concrete ways. Extending the architecture to temporal sequence modelling, whether through 3D convolutions, LSTM-based recurrence, or transformer-based attention over video frames, would open the door to full-sentence ArSL recognition. Deploying the model on embedded platforms such as the NVIDIA Jetson Nano would determine whether sub-50 ms inference is achievable without GPU-class hardware. Collecting or obtaining access to a secondary unconstrained dataset would allow honest cross-domain generalisation testing. Substituting a lightweight backbone such as MobileNetV3 or EfficientNet-Lite and measuring the accuracy-latency trade-off would clarify the engineering choices available for edge deployment. And applying higher input resolutions (224×224 or the full 299×299) together with spatial attention modules targeted at finger-joint regions would directly tackle the geometric ambiguities identified in the confusion analysis as the primary remaining source of error.

References

[1] O. Maarouf, A. Maarouf, R. El Ayachi, and M. Biniz, “Automatic translation from English to Amazigh using transformer learning,” IJEECS, vol. 34, no. 3, p. 1924, Jun. 2024, doi: 10.11591/ijeecs.v34.i3.pp1924-1934.

[2] H. AbdElghfar, H. A. Youness, M. Wahba, and H. M. Abdelaal, “An automated framework for qur’anic education of the hearing-impaired using body pose classification and Arabic sign language integration,” Sci Rep, vol. 16, no. 1, p. 5939, Feb. 2026, doi: 10.1038/s41598-026-36578-z.

[3] A. A. Alethary, A. H. Aliwy, and N. S. Ali, “Automated Arabic-Arabic sign language translation system based on 3D avatar technology,” IJAAS, vol. 11, no. 4, p. 383, Dec. 2022, doi: 10.11591/ijaas.v11.i4.pp383-396.

[4] M. M. Kamruzzaman, “Arabic Sign Language Recognition and Generating Arabic Speech Using Convolutional Neural Network,” Wireless Communications and Mobile Computing, vol. 2020, pp. 1–9, May 2020, doi: 10.1155/2020/3685614.

[5] A. Maarouf, O. Maarouf, R. El Ayachi, and M. Biniz, “Deep Learning Approach for Arabic Sign Language Alphabet Recognition,” Salud, Ciencia y Tecnología, vol. 5, p. 2309, Oct. 2025, doi: 10.56294/saludcyt20252309.

[6] A. H. Aliwy and A. A. Alethary, “Development of arabic sign language dictionary using 3D avatar technologies,” IJEECS, vol. 21, no. 1, p. 609, Jan. 2021, doi: 10.11591/ijeecs.v21.i1.pp609-616.

[7] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA: IEEE, Jun. 2016, pp. 770–778. doi: 10.1109/CVPR.2016.90.

[8] C. Lugaresi et al., “MediaPipe: A Framework for Building Perception Pipelines,” Jun. 14, 2019, arXiv: arXiv:1906.08172. doi: 10.48550/arXiv.1906.08172.

[9] A. Dosovitskiy et al., “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale,” Jun. 03, 2021, arXiv: arXiv:2010.11929. doi: 10.48550/arXiv.2010.11929.

[10] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” Dec. 11, 2015, arXiv: arXiv:1512.00567. doi: 10.48550/arXiv.1512.00567.

[11] M. Alamri and S. Lajmi, “Design a smart platform translating Arabic sign language to English language,” IJECE, vol. 14, no. 4, p. 4759, Aug. 2024, doi: 10.11591/ijece.v14i4.pp4759-4774.

[12] M. H. Ismail, S. A. Dawwd, and F. H. Ali, "Dynamic hand gesture recognition of Arabic sign language by using deep convolutional neural networks," IJEECS, vol. 25, no. 2, p. 952, Feb. 2022, doi: 10.11591/ijeecs.v25.i2.pp952-962.

[13] M. H. Ismail, S. A. Dawwd, and F. H. Ali, "Static hand gesture recognition of Arabic sign language by using deep CNNs," IJEECS, vol. 24, no. 1, p. 178, Oct. 2021, doi: 10.11591/ijeecs.v24.i1.pp178-188.

[14] A. A. Alani and G. Cosma, "ArSL-CNN a convolutional neural network for Arabic sign language gesture recognition," IJEECS, vol. 22, no. 2, p. 1096, May 2021, doi: 10.11591/ijeecs.v22.i2.pp1096-1107.

[15] M. Zakariah, Y. A. Alotaibi, D. Koundal, Y. Guo, and M. Mamun Elahi, "Sign Language Recognition for Arabic Alphabets Using Transfer Learning Technique," Computational Intelligence and Neuroscience, vol. 2022, pp. 1–15, Apr. 2022, doi: 10.1155/2022/4567989.

[16] Z. Alsaadi, E. Alshamani, M. Alrehaili, A. A. D. Alrashdi, S. Albelwi, and A. O. Elfaki, "A Real Time Arabic Sign Language Alphabets (ArSLA) Recognition Model Using Deep Learning Architecture," Computers, vol. 11, no. 5, p. 78, May 2022, doi: 10.3390/computers11050078.

[17] A. Wadhawan and P. Kumar, "Sign Language Recognition Systems: A Decade Systematic Literature Review," Arch Computat Methods Eng, vol. 28, no. 3, pp. 785–813, May 2021, doi: 10.1007/s11831-019-09384-2.

[18] H. Cooper, B. Holt, and R. Bowden, "Sign Language Recognition," in Visual Analysis of Humans, T. B. Moeslund, A. Hilton, V. Krüger, and L. Sigal, Eds., London: Springer London, 2011, pp. 539–562. doi: 10.1007/978-0-85729-997-0_27.

[19] M. S. Amin, M. T. Amin, M. Y. Latif, A. A. Jathol, N. Ahmed, and M. I. N. Tarar, "Alphabetical Gesture Recognition of American Sign Language using E-Voice Smart Glove," in 2020 IEEE 23rd International Multitopic Conference (INMIC), Bahawalpur, Pakistan: IEEE, Nov. 2020, pp. 1–6. doi: 10.1109/INMIC50486.2020.9318185.

[20] 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, vol. 9, pp. 67548–67558, 2021, doi: 10.1109/ACCESS.2021.3077386.

[21] W. Pan, X. Zhang, and Z. Ye, "Attention-Based Sign Language Recognition Network Utilizing Keyframe Sampling and Skeletal Features," IEEE Access, vol. 8, pp. 215592–215602, 2020, doi: 10.1109/ACCESS.2020.3041115.

[22] T. Zhao, J. Liu, Y. Wang, H. Liu, and Y. Chen, "Towards Low-Cost Sign Language Gesture Recognition Leveraging Wearables," IEEE Trans. on Mobile Comput., vol. 20, no. 4, pp. 1685–1701, Apr. 2021, doi: 10.1109/TMC.2019.2962760.

[23] I. Papastratis, K. Dimitropoulos, D. Konstantinidis, and P. Daras, "Continuous Sign Language Recognition Through Cross-Modal Alignment of Video and Text Embeddings in a Joint-Latent Space," IEEE Access, vol. 8, pp. 91170–91180, 2020, doi: 10.1109/ACCESS.2020.2993650.

[24] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in 2017 International Conference on Engineering and Technology (ICET), Antalya: IEEE, Aug. 2017, pp. 1–6. doi: 10.1109/ICEngTechnol.2017.8308186.

[25] C. Szegedy et al., "Going deeper with convolutions," in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA: IEEE, Jun. 2015, pp. 1–9. doi: 10.1109/CVPR.2015.7298594.

[26] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI: IEEE, Jul. 2017, pp. 2261–2269. doi: 10.1109/CVPR.2017.243.

[27] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[28] N. Alasmari and S. Asiri, "ASLDetect: Arabic sign language detection using ResNet and U-Net like component," Sci Rep, vol. 15, no. 1, p. 18012, May 2025, doi: 10.1038/s41598-025-01588-w.

[29] A. M. Buttar, U. Ahmad, A. H. Gumaei, A. Assiri, M. A. Akbar, and B. F. Alkhamees, "Deep Learning in Sign Language Recognition: A Hybrid Approach for the Recognition of Static and Dynamic Signs," Mathematics, vol. 11, no. 17, p. 3729, Aug. 2023, doi: 10.3390/math11173729.

[30] M. Al-Qurishi, T. Khalid, and R. Souissi, "Deep Learning for Sign Language Recognition: Current Techniques, Benchmarks, and Open Issues," IEEE Access, vol. 9, pp. 126917–126951, 2021, doi: 10.1109/ACCESS.2021.3110912.

[31] D. S. Breland, S. B. Skriubakken, A. Dayal, A. Jha, P. K. Yalavarthy, and L. R. Cenkeramaddi, "Deep Learning-Based Sign Language Digits Recognition From Thermal Images With Edge Computing System," IEEE Sensors J., vol. 21, no. 9, pp. 10445–10453, May 2021, doi: 10.1109/JSEN.2021.3061608.

[32] M. Al-Hammadi et al., "Deep Learning-Based Approach for Sign Language Gesture Recognition With Efficient Hand Gesture Representation," IEEE Access, vol. 8, pp. 192527–192542, 2020, doi: 10.1109/ACCESS.2020.3032140.

[33] N. B. Ibrahim, M. M. Selim, and H. H. Zayed, "An Automatic Arabic Sign Language Recognition System (ArSLRS)," Journal of King Saud University - Computer and Information Sciences, vol. 30, no. 4, pp. 470–477, Oct. 2018, doi: 10.1016/j.jksuci.2017.09.007.

[34] M. Deriche, S. O. Aliyu, and M. Mohandes, "An Intelligent Arabic Sign Language Recognition System Using a Pair of LMCs With GMM Based Classification," IEEE Sensors J., vol. 19, no. 18, pp. 8067–8078, Sep. 2019, doi: 10.1109/JSEN.2019.2917525.

[35] G. Tharwat, A. M. Ahmed, and B. Bouallegue, "Arabic Sign Language Recognition System for Alphabets Using Machine Learning Techniques," Journal of Electrical and Computer Engineering, vol. 2021, pp. 1–17, Oct. 2021, doi: 10.1155/2021/2995851.

[36] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," Apr. 10, 2015, arXiv: arXiv:1409.1556. doi: 10.48550/arXiv.1409.1556.

[37] F. Zhuang et al., "A Comprehensive Survey on Transfer Learning," Proc. IEEE, vol. 109, no. 1, pp. 43–76, Jan. 2021, doi: 10.1109/JPROC.2020.3004555.

[38] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," IEEE Trans. Knowl. Data Eng., vol. 22, no. 10, pp. 1345–1359, Oct. 2010, doi: 10.1109/TKDE.2009.191.

[39] G. Latif, N. Mohammad, J. Alghazo, R. AlKhalaf, and R. AlKhalaf, "ArASL: Arabic Alphabets Sign Language Dataset," Data in Brief, vol. 23, p. 103777, Apr. 2019, doi: 10.1016/j.dib.2019.103777.

[40] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL: IEEE, Jun. 2009, pp. 248–255. doi: 10.1109/CVPR.2009.5206848.

[41] B. Koonce, "EfficientNet," in Convolutional Neural Networks with Swift for Tensorflow, Berkeley, CA: Apress, 2021, pp. 109–123. doi: 10.1007/978-1-4842-6168-2_10.

[42] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," The journal of machine learning research, vol. 15, no. 1, pp. 1929–1958, 2014.

[43] W. Liu, X. Wang, J. Owens, and Y. Li, "Energy-based out-of-distribution detection," Advances in neural information processing systems, vol. 33, pp. 21464–21475, 2020.

[44] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," Jan. 30, 2017, arXiv: arXiv:1412.6980. doi: 10.48550/arXiv.1412.6980.

[45] O. Koller, "Quantitative Survey of the State of the Art in Sign Language Recognition," Aug. 29, 2020, arXiv: arXiv:2008.09918. doi: 10.48550/arXiv.2008.09918.

[46] A. Howard et al., "Searching for MobileNetV3," in 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South): IEEE, Oct. 2019, pp. 1314–1324. doi: 10.1109/ICCV.2019.00140.