

Speech Recognition System Using Deep Learning Techniques for Arabic Sign Language

Sultan Ali Senan

ORCID ID: [\[ORCID ID\]](#)

Artificial Intelligence – College of Informatics – Midocean University – United Arab Emirates

Email: 253002563@st.midocean.edu.km

Dr. Amjad Qatish ([ORCID ID](#)) Artificial Intelligence, College of Informatics, Midocean University, United Arab Emirates, Email: [\[email\]](#)

Abstract:

Objective: Design, implementation and evaluation of deep learning based recognition of Arabic Sign Language, as well as a controlled benchmarking comparative study between conventional machine learning techniques and deep learning architectures on the same standardized dataset in order to pinpoint the most optimal architecture in terms of performance vs training cost. **Methodology:** The publicly available Arabic Alphabets Sign Language dataset, comprising 54,049 grayscale images across 32 alphabet classes, was adopted. Seven models were implemented: four conventional classifiers, such as Random Forest, Linear Support Vector Machine, HistGradientBoosting and Multilayer Perceptron, and three deep neural architectures, including a custom convolutional network, Light ResNet and MobileNetV2 transfer learning model. Stratified train/val split of 80%/20% with a fixed random seed was utilized. **Results:** The proposed convolutional network provided the highest validation accuracy of 94.88%, MobileNetV2 showed 93.64% and Light ResNet scored 80.97%. In turn, among the classifiers, Random Forest proved the most efficient reaching 80.25% accuracy and Linear SVM – the worst with 47.85%. Training time of the proposed model was 103.54 seconds vs 280.73 seconds for the transfer learning one. **Conclusion:** It is possible to achieve high performance with a purpose-built compact architecture which significantly outperforms larger pre-trained one on grayscale domain-specific imagery and requires much less computational power.

Keywords: Arabic Sign Language, Deep Learning, Convolutional Neural Networks, Model Benchmarking, Real-Time Recognition.

1. Introduction:

Communication is one of the basic needs for humankind, but millions of deaf and hard-of-hearing people keep on facing challenges in their daily lives due to the accessibility problems. Sign languages remain the most common mode of communication in deaf communities, however, automatic translation from sign language to written or spoken language is still an extremely complicated computational problem. Arabic Sign Language is employed in over twenty Arab

countries, but it is significantly understudied in artificial intelligence in comparison with widely studied sign languages, including American Sign Language and British Sign Language.

Unlike other sign languages, Arabic Sign Language is a full natural language, not the manual representation of spoken Arabic language. It has its grammar, and it conveys meanings through the combination of manual features, i.e., hand shape, orientation, location, and movement, as well as non-manual features such as facial expressions and head movements. Manual alphabet, consisting of 32 signs for all letters of the written Arabic, has a special place in Arabic Sign Language as it is mainly used for spelling proper nouns, technical terms, and words without existing sign. Thus, automatic detection of manual alphabet is a critical element of Arabic Sign Language recognition and the main objective of the present study.

The main peculiarity of Arabic Sign Language technology lies in the regional diversity. Due to the fact that there were no centrally organized efforts to develop Arabic Sign Language in contrast to some other sign languages, the variations of vocabulary and articulation occur in different Arab countries. It makes building datasets more complicated, and it partially explains the fact that there are fewer publicly available Arabic Sign Language resources than American or British Sign Language ones.

Modern deep learning methods brought a revolution to computer vision and pattern recognition. Convolutional Neural Networks proved to be very efficient in recognizing visual patterns, therefore, they seem to be a great solution for sign language recognition, which requires identification of hand shape, finger configuration, and spatial orientation. However, the actual question about the best architecture to be used with any particular sign language dataset is usually not solved through controlled experiments, and people often choose the architecture based on different results achieved under heterogeneous conditions.

1.1 Statement of the Problem

Even though there have been numerous advancements in the area of sign language recognition, the technological support for Arabic Sign Language is still relatively poor. The deaf people in the Arab region experience many issues with communication because of the lack of automatic recognition systems as well as a shortage of interpreters reported in the literature, which hinders access to public services, education, and job opportunities.

Moreover, there is a gap in the literature regarding systematic comparisons of traditional machine learning methods and various deep learning algorithms on the same standardized data set, under the same conditions. The comparison is necessary in order to evaluate whether one of them is better, and how much better in practice. The majority of previous studies do not include computational complexity in the process of evaluation and rarely implement models on practice.

1.2 Research Questions

Therefore, this research is aimed to answer the following research questions:

- Which algorithms perform better than others in terms of the recognition accuracy when evaluated on the same Arabic Sign Language data set under the same experimental conditions?
- Which algorithm is the most suitable choice for deployment purposes due to its high accuracy, acceptable training time, and implementation convenience?

- Does a purpose-built custom CNN algorithm perform better than transfer learning using large pre-trained models for grayscale images of signs used in sign language?

1.3 Objectives of the Study

- To collect, validate, and process the data set of images of the Arabic Sign Language alphabet.
- To develop and train a custom convolutional model for sign language recognition on a regular computer.
- To use conventional machine learning algorithms as well as alternative deep learning approaches for comparison.
- To compare all models according to a unified protocol with regard to validation accuracy and training time.
- To implement the optimal model as part of a real-time translation system.

1.4 Significance of the Study

Technologically, the significance of the study lies in providing a documented comparison of seven models on a publicly available large dataset for Arabic Sign Language, with empirical accuracy and computational performance results reported for each of the seven methods, thus providing an important benchmark which is currently rare for this sign language. Socially, it provides a practical tool designed to help Deaf and hard of hearing individuals access public services, education, and the job market, as well as advance digitization of Arabic Sign Language materials in an area where more attention is needed.

1.5 Scope and Delimitations

The scope of the study includes the recognition of 32 static signs corresponding to the letters of the Arabic manual alphabet. Recognition of dynamic signs at the word level and translation of continuous signing and complete sentences are out of scope of the current research. The evaluation procedure includes stratified random partitioning and not a signer-independent protocol. All deep learning models have been trained for a fixed number of five epochs using a central processing unit architecture. The implementation of the deployed system outputs recognized letters and translated words in text form on the screen; audible speech synthesis was not implemented.

2. Previous Studies:

The creation of the Arabic Alphabets Sign Language dataset is a crucial milestone in the development of Arabic Sign Language recognition technology (Latif, Mohammad, Alghazo, AlKhalaf and AlKhalaf, 2019). This dataset includes 54,049 grayscale images of 32 Arabic alphabet signs, obtained from over forty subjects and converted to a 64×64 pixel format. Before this dataset became available to the research community, the majority of studies had to rely on private datasets, which precluded comparison between different works. The publication of this dataset has provided a common benchmark for research in the area, and it is the one chosen for use in this study.

In his proposal, Noor (2024) created a hybrid system of Arabic Sign Language recognition based on convolutional and long short-term memory networks. Tested on a custom dataset of 4,000 images of ten dynamic and ten static gesture words, the convolutional network showed 94.40%

accuracy, while the recurrent network scored 82.70%. The main motivation behind this work was the shortage of certified interpreters in Saudi Arabia, an issue that is also a concern for the present study. The limitation of this work is the relatively small size of the dataset and the restriction to a vocabulary of twenty words.

Another relevant work by Podder et al. (2023) addresses the problem of signer-independent Arabic Sign Language recognition, where the system has to be able to recognize gestures performed by people not included in the training sample. Their solution combines either MobileNetV2 or ResNet18 backbone with long short-term memory and a custom self-attention mechanism and achieves 87.69% accuracy, 88.57% precision, and F1-score of 87.72% on RGB video with face and hand segmentation, which is almost eleven percentage points better than the state-of-the-art scores previously reported on signer-independent tasks. The limitation of this work is its multi-stage pipeline, which consumes considerable resources.

Bani Baker (2023) studied the impact of transfer learning on the Arabic alphabet sign recognition in the case of a limited training set, and found out that in this case the transfer learning performs better than training a model from scratch. In this study, we test the hypothesis that the advantage of transfer learning exists even when there are more than fifty thousand training samples and the images are single-channel grayscale rather than RGB photography.

In addition to the Arab context, we should consider the recent developments in the domain of American Sign Language recognition by Kumar (2023) in real-time with the help of hand landmarks and convolutional classifiers with accuracy up to 99.95%, and in the field of word-level recognition of Indian Sign Language by Das (2023) with the use of VGG-19 and a bidirectional long short-term memory with histogram-based keyframe selection and an average accuracy of 87.67%. According to Hussain (2023), the research and evaluation practice is not equally distributed between regional languages, which makes the accumulation of knowledge difficult.

2.1 Classical Approaches and the Value of Being a Baseline

Before the advent of deep learning, sign language recognition was performed through a combination of classical computer vision techniques and traditional machine learning classifiers. The typical workflow included segmentation of the hand region based on the skin colour, extracting handcrafted features such as the histograms of oriented gradients or the scale-invariant feature transforms, and classification through classifiers such as the support vector machines or decision tree ensembles. Such approaches have demonstrated reasonable performance for restricted vocabularies but poor generalizability for changes in the lighting conditions, backgrounds, and signers' appearances because the handcrafted features were designed instead of being learned.

Nonetheless, classical approaches still remain highly valuable as baselines for evaluation. Determining the performance possible with a well-tuned classical classifier is a good benchmark in assessing the actual performance gain from utilizing deep learning. The Random Forests classifier constructs an ensemble of decision trees trained on bootstrapped subsets, making it highly robust and requiring no hyperparameters' fine-tuning. The gradient boosting builds an ensemble of models sequentially, so each next model attempts to minimize the residuals made by previous models in the ensemble. The support vector machines attempt to find the maximum margin separating hyperplane in the feature space, and their performance crucially depends on the separability of the feature space.

A practical constraint associated with these approaches is the issue of dimensionality. An image with the size of 64x64 pixels yields a 4,096-dimensional feature vector, and utilizing such classifiers on tens of thousands of samples with that dimensionality places high demands on memory and computational power. Common strategies for dealing with such problems include stratified sampling of the training dataset and downsampling of the images. Both techniques are used in the experimental setup described in the Section 3. This particular constraint is useful to consider in itself as it reflects a true scalability limitation of classical approaches when applied to image data.

2.2 Theoretical Basis of Convolutional Architectures

The Convolutional Neural Network is particularly suited for image classification due to the two underlying structural assumptions that fit very well with the characteristics of visual data. First of all, it is the assumption of locality: pixels that are close to each other in space are more likely to have semantic connections than distant pixels, hence, convolutional filters work on small local neighbourhoods. The second assumption is the one of translation equivariance: a visual pattern does not change in terms of identity regardless of its position inside the frame, and the same filter can be applied to all spatial locations in order to find this pattern at any place. In total, these two assumptions lead to the reduction of the number of parameters in comparison to the fully-connected networks.

Both assumptions are extremely important in case of sign language recognition: the identity of a manual sign is determined by the relative position of fingers and palm, i.e. by a local spatial relationship. A sign is the same thing whether the hand position slightly varies from the centre of the captured frame towards the left or right side. However, a classifier working with the flattened vector of pixels would treat every single location as an independent feature and thus would ignore both assumptions, which explains the considerable empirical difference discussed in Section 4.

The regularization mechanism plays a similar role. The Dropout technique proposed by Srivastava et al. (2014) randomly drops some of the units of the network during the training process, so that the network cannot rely on any particular pathway and creates multiple pathways, making the representation distributed and thus helping in reducing overfitting. The network proposed in this work uses the moderate dropout rate inside the convolutional blocks and high dropout rate before the final layer.

2.3 Transfer Learning and Domain Similarity

The transfer learning technique utilizes the already trained model for solving some problem as a base and trains the classification head only, adapting this model for a new task. One of the most widely used architectures for such purposes is MobileNetV2, proposed by Sandler et al. (2018), and uses depthwise separable convolutions and inverted residual blocks for fast inference on hardware constraints.

Rationale for the Transfer Learning Approach Transfer learning is based on the theoretical idea that low-level visual features (edges, simple textures, etc.) can be effectively reused for multiple visual tasks. However, the amount of benefit that can be derived is strongly dependent on the similarity of the domains (source and target). If the target domain has significantly different imagery compared to the source one, part of the learned representation will become useless or even counter-productive. For example, grayscale sign language imagery at 64 by 64 resolution is quite different

from color photographs on ImageNet, and therefore performance of transfer learning on this dataset should be empirically tested.

2.4 Research Gap

Three gaps in the existing literature have been identified. First, the comparison of traditional and deep learning models on a standardized Arabic Sign Language dataset under the same conditions is rare, and it is difficult to judge whether a certain family of approaches is more appropriate than others. Second, the computation cost is not usually reported by researchers along with accuracy, although the time of training and inference, as well as hardware resources needed, are crucial factors in designing assistive technology for regular use. Third, much work ends up at the stage of model evaluation without creating a working system out of the trained classifier. The current research covers all three gaps.

3. Methodology:

3.1 Dataset and Data Integrity Verification

This research uses Arabic Alphabets Sign Language dataset containing 54,049 grayscale images belonging to 32 classes and provided in uniform resolution of 64 by 64 pixels. A data integrity verification procedure was employed before the experiments began: a recursive search through the dataset directory was done to associate each filename to its absolute path, and existence, non-zero size, and successful image decoding were checked for every record in the metadata file. It turned out that there are no missing and corrupt files, and a total of 54,049 images was successfully verified. Classes were then mapped to integers from 0 to 31.

Table (1): Summary of the Arabic Alphabets Sign Language Dataset

Property	Value
Dataset name	Arabic Alphabets Sign Language (ArASL)
Source	Latif et al. (2019), publicly available
Total images	54,049
Number of classes	32 Arabic alphabet signs
Image resolution	64 x 64 pixels
Colour format	Grayscale (single channel)
Number of signers	More than 40 participants
Integrity check result	54,049 valid; 0 missing; 0 corrupt

3.2 Dataset Characteristics and Capture Variability

Prior to the development of any model, an exploratory investigation of intensity statistics for various sample classes was made in order to get an understanding of the corpus visual characteristics. Mean pixel intensity turned out to be dependent on the class in question, the vast majority of sample classes having a mean intensity of approximately 170 to 192 on the regular eight-bit scale – thus being consistent with the imagery taken against a light background. There is one class that departs from this norm in terms of mean intensity, which turned out to have a

minority of images that were taken under different lighting with a mean intensity of approximately 66 compared with 191 in the rest of the class.

This means that there is some heterogeneity of lighting across and within the sample classes. As a consequence, there are two things worth noting. First of all, this naturally introduces variability that can potentially improve robustness of a trained model with respect to various lighting conditions, in effect working in a similar way to intentional augmentation of the corpus. On the other hand, if certain lighting conditions correlate with certain classes, there is a danger of partial exploitation of this fact by a model instead of focusing on hand configuration as the only discriminating parameter. While in the case of the current experiments, this danger did not appear to be realized, the fact is recorded in the spirit of transparency, and will be returned to as a limitation in section 4.8.

3.3 Preprocessing

There are two preprocessing pipelines used depending on the model family. In the case of deep learning architectures, the entire image set is preloaded to memory as an unsigned eight-bit array of approximately 211 MB, eliminating the need to repeatedly load and decode the images throughout the training process. Then a data pipeline converts the images to the floating-point format and normalizes their intensities to the unit interval, shuffling using a buffer of 10,000 samples, batching at 64, and prefetching in order to overlap data processing with computation.

The traditional classifiers require the images to be flattened into one-dimensional feature vectors, thus imposing a certain scaling constraint. At the native resolution, the full dataset will yield a matrix of approximately 221 million elements, surpassing practical memory capabilities for methods like SVM. Thus, two commonly accepted mitigations were applied: first of all, a randomized stratified subset of 5,000 training and 2,000 validation samples was extracted to preserve the original class proportions, and secondly, spatial downsampling was done from 64 by 64 to 32 by 32 pixels, reducing dimensionality from 4,096 to 1,024 features per sample. Pixel values were then scaled to the unit interval.

3.4 Proposed Architecture

The suggested convolutional neural network receives a grayscale image of size 64 by 64 and is made up of three convolutional layers sequentially and a fully-connected head. Convolution is performed in each layer using a filter size of 3 by 3 with a rectified linear unit activation function, then max pooling with a window of 2 by 2 and finally applying a dropout rate of 0.25. The number of filters is incremented progressively from 16 to 32 and eventually to 64 in the last block allowing to learn more abstract representations while downscaling spatial resolution.

After that, the output from the third block is flattened and sent through a dense layer with 128 neurons and rectified linear unit activation function. This is followed by another dropout layer with a dropout of 0.5 and finally a dense layer with 32 neurons and a softmax activation that produces a probability distribution of 32 classes. The gradual dropout is employed in order to apply more weight towards the regularization of the densely connected part of the model, which includes most of its parameters.

3.5 Baseline Models

In addition, six different baseline models are implemented for comparison, including four classical classifiers and two alternative deep architectures. The configuration of the models and the reasons behind their selection are given in Table 2.

Table (2): Configuration of Baseline Models

Model	Configuration	Rationale for Inclusion
Random Forest	100 estimators, parallel execution	Robust ensemble baseline requiring minimal tuning
Linear SVM	$C = 1.0$, squared hinge loss, dual = False	Tests linear separability of the raw pixel space
HistGradientBoosting	Maximum 100 boosting iterations	Represents modern gradient boosting methods
Multilayer Perceptron	Hidden layers of 128 and 64 units, Adam solver	Neural baseline lacking spatial inductive bias
Light ResNet	Five layers with two residual skip blocks	Tests whether residual depth benefits this dataset
MobileNetV2	Frozen ImageNet backbone, grayscale-to- RGB layer	Tests transferability of natural-image features

3.6 Training Protocol

A stratified split of the dataset has been performed in order to allocate 80% of samples for training and 20% for validation, thereby ensuring class frequencies are preserved. The train set contains 43,239 images and the validation set contains 10,810 images. In all experiments, the random seed value of 42 is chosen in order to ensure repeatability. Deep learning models are trained with Adam (Kingma and Ba, 2015) optimizer, categorical cross-entropy loss function, batch size of 64 and five epochs on CPU hardware.

More importantly, however, all seven models were trained and validated in a single automated execution with the same partition and evaluation code. This is a crucial methodology aspect: when results are aggregated from independent experiments, differences in accuracy can stem not from different capabilities of the architectures but from variations in data partitioning, preprocessing or metric implementation. In this way, training of all models in one procedure helps to eliminate such a confounding factor and ensure that differences in accuracy are indeed caused by different model architectures.

Table (3): Training Hyperparameters

Parameter	Value
Framework	TensorFlow / Keras
Optimizer	Adam
Loss function	Categorical cross-entropy
Batch size	64
Epochs (deep learning models)	5
Train / validation split	80% / 20%, stratified
Training samples	43,239
Validation samples	10,810

Parameter	Value
Random seed	42
Training hardware	CPU-based environment

3.7 Evaluation Metrics

Validation accuracy was chosen as the main performance measure: it is defined as the share of validation samples predicted to belong to the right class. Given that the dataset is nearly balanced among 32 classes and stratification maintains this property, accuracy is a reasonable and comprehensible way of quantifying the model's performance. Training time was considered as the secondary performance measure since, in practice, the computation complexity plays a role in deciding which architecture to choose when deploying it on a device.

In order to make the paper complete, the definitions of the standard classification metrics are given below. These metrics are computed for each class being the positive class in the one-versus-rest approach: TP, TN, FP and FN stand for true positives, true negatives, false positives and false negatives respectively. Accuracy is the ratio of true positives and true negatives to the number of predictions overall. Precision is the ratio of true positives to the number of true and false positives. Recall is the ratio of true positives to the number of true and false negatives. Finally, the F1-score is the harmonic mean of precision and recall.

3.8 Real-Time Translation System

The best-performing model found in the benchmark was implemented in the real-time translation system designed as the client-server application. The server-side is implemented using the FastAPI web framework and contains the prediction endpoint: it accepts the frame of an image uploaded to it, decodes it, makes it grayscale, resizes it to the 64 by 64 pixels required by the model as its input, normalizes the pixels' values, and returns the predicted class and the corresponding confidence. The model is loaded at server start-up and kept in memory, so that the cost of each request is reduced to the cost of inference rather than that of loading model weights every time.

The client-side is implemented as the web interface that does not require any installation by the user. It accesses the device camera, shows the live preview of the video feed, captures the frames in 350-millisecond intervals and sends them to the prediction endpoint. Predicted letters are consecutively displayed in the text area; controls for inserting a space, deleting the last character and clearing the current text are provided. The stabilization algorithm accepts the letter only after it is predicted in consecutive frames, thus avoiding transient mistakes made while moving the hand from one sign to another to corrupt the result.

3.9 Reproducibility Measures

Various methods were used in order to make sure that the results can be reproduced independently. Specifically, a deterministic seed of 42 is used to initialize randomness in all stochastic parts of the pipeline including stratified partitioning, training batch shuffling, bootstrapping of samples in the Random Forest ensemble and weights initialization of neural networks. As a result, running the benchmark multiple times on the same data results in identical partitions and comparable results.

The benchmark is implemented as a single script that trains and evaluates all seven models one after another in one execution rather than a set of scripts whose results are combined later. After its

completion, the script produces a report containing the accuracy and the time needed to train each model as well as a visual comparison of the results. Being generated by the program and not transcribed manually from the results, all figures reported in Section 4 can be traced back to the particular run of the experiment and thus transcription errors can be completely ruled out as a possible source of discrepancies.

The dataset used in the study is publicly available and preprocessing procedures applied to it (normalization, stratification and spatial downsampling) are deterministic due to the fixed seed. The trained model is saved to the disk with a mapping of indices to class labels being stored explicitly along with it, so that the order of classes used in the training process is preserved during the inference. The latter might sound trivial but is quite important, because any inconsistency in the class labeling would result in incorrect predictions without changing the training results at all.

4. Results and Discussion:

This section presents the results of the comparative benchmark and their interpretation with respect to the research questions raised in Section 1.2. Tables 4 and 5 summarize the data partition and validation accuracy and training time of all seven evaluated models respectively.

Table (4): Dataset Split Summary

Subset	Number of Images	Percentage	Number of Classes
Training	43,239	80%	32
Validation	10,810	20%	32
Total	54,049	100%	32

Table (5): Accuracy of Validation and Training Time of all Tested Models

Model	Model Type	Validation Accuracy	Training Time (s)
Custom CNN (proposed)	Deep Learning	94.88%	103.54
MobileNetV2 (transfer learning)	Deep Learning	93.64%	280.73
Light ResNet	Deep Learning	80.97%	152.44
Random Forest	Traditional ML	80.25%	1.90
HistGradientBoosting	Traditional ML	77.75%	237.27
Multilayer Perceptron	Traditional ML	61.60%	18.72
Linear SVM	Traditional ML	47.85%	210.60

The results show vast differences among all tested techniques. Custom convolutional network proposed showed the highest validation accuracy rate of 94.88%, followed by 93.64% of MobileNetV2 transfer learning model. Accuracy of Light ResNet was equal to 80.97%, which is lower both for deep learning architectures and barely better than the best traditional classifier. For traditional classifiers, Random Forest was the most accurate one with 80.25%, followed by

HistGradientBoosting with 77.75%, Multilayer Perceptron with 61.60% and Linear Support Vector Machine with 47.85%. Graphical presentation of the comparison can be seen on Figure 1.

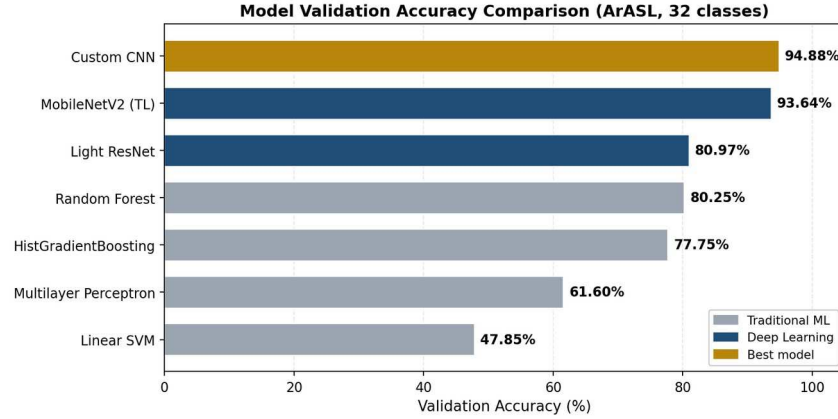


Figure (1): Comparison of Model Validation Accuracy

By model types, two top convolutional models are in one band above 93%, while all five remaining are in the cluster from 47% to 81%. Special attention should be paid to Light ResNet, which belongs to the deep learning models, but its accuracy is positioned between the tree-based ensemble models rather than with the rest of the convolutional models. It shows that being a deep learning model is not enough to achieve high accuracy, because architecture that matches the data characteristics is the key here.

At the same time, another significant difference is computational cost. The proposed network not only achieved the highest accuracy but did so in 103.54 seconds, compared with 280.73 seconds required by MobileNetV2, as detailed in Table 6. The proposed model therefore delivered superior accuracy while consuming approximately 37% of the training time of the transfer learning alternative.

Table (6): Performance vs Training Cost of Deep Learning Architectures

Model	Validation Accuracy	Training Time (s)	Relative Cost
Custom CNN (proposed)	94.88%	103.54	1.00x (reference)
Light ResNet	80.97%	152.44	1.47x
MobileNetV2 (transfer learning)	93.64%	280.73	2.71x

4.1 Deep Learning versus Machine Learning

The difference in the two categories is big; the proposed architecture performed better than the best machine learning model by 14.63 percentage points. This difference arises from the inherent structural difference in the models. The traditional classification methods work on flattened pixel vectors, which lose the spatial relationship of the pixels that define hand shapes, while the deep learning models use convolutional layers, which maintain the spatial relationship among neighboring pixels. Hence, the performance difference is as theorized in section 2.2.

Similarly, the difference within the family of machine learning models is very educative. While random forest performed better than Linear SVM by 32.40 percentage points when using the same input, the implication is that the choice of the algorithm within a family is very important just like the family itself. Indeed, the performance trend where the ensemble methods perform better than neural networks and neural networks perform better than linear classifiers is an indication of their capability to capture non-linear relationship in the pixel intensity space. The performance of the linear classifier is expected since the pixel intensity space is not linearly separable.

4.2 Custom Architecture versus Transfer Learning

The result of superiority of the proposed network in comparison with MobileNetV2 transfer learning is especially worth mentioning because usually transfer learning is considered an advantage. The thing is that domain mismatch occurs when pretraining a network. MobileNetV2 has been pretrained on ImageNet, which is a set of natural colour photos where some features like colour gradients and complex textures cannot be entirely used for the task of low-resolution monochrome pictures of hand shapes discrimination.

The result is also in line with the general rule that the benefit of transfer learning gets smaller when the distance between source and target domains becomes greater. In addition, there is more than fifty thousand training images in the dataset which is enough to train a good network without transfer learning at all. Therefore, the result contradicts the conclusions of Bani Baker (2023) who claims that there is some advantage of transfer learning for Arabic sign recognition because of insufficient data.

The poor performance of the Light ResNet with the 80.97% accuracy once again proves the idea that increased complexity is not always helpful. Residual connections have been introduced to enable training of very deep networks by improving gradient propagation but, in this case, the dataset is quite simple from the visual point of view – it consists of low-resolution single-channel images of hand shapes. In the five-epochs regime which is the same for all tested networks, the increased depth and number of parameters do not bring any advantages, and the architecture needs some more epochs to achieve the maximum possible performance. With the fixed computational budget, the custom compact architecture is better.

4.3 Comparison with Prior Studies

Table 7 puts the measured accuracy into context with respect to previous studies. It should be noted that different datasets, number of classes and evaluation procedures were used in previous studies.

Table (7): Comparison with Previous Sign Language Recognition Works

Study	Dataset	Approach	Accuracy
Proposed Custom CNN (this study)	ArASL, 54,049 images, 32 classes	Custom three-block CNN	94.88%
Noor (2024) - CNN	Custom set, 4,000 images	Hybrid CNN component	94.40%
Noor (2024) - LSTM	Custom set, 4,000 images	Hybrid LSTM component	82.70%
Podder et al. (2023)	Custom RGB video set	MobileNetV2 with LSTM	87.69%
Das (2023)	ISL video word dataset	VGG-19 with BiLSTM	87.67%

Proposed method obtains accuracy that rivals the best achieved by Noor (2024) in 94.40% even when tested on a set that is more than thirteen times bigger and includes the entire set of 32 classes constituting the alphabet. It also surpasses the independent signer benchmark accuracy of 87.69% obtained by Podder et al. (2023). However, it should be noted that an independent signer test is a more restrictive testing paradigm compared to the one used here.

4.4 Practical Implications

The results have several important practical implications for building assistive sign language systems. One of them has to do with the architecture selection. People working in the domain of sign language recognition should not expect that the transfer learning from a large and generic model will be the best choice if there is enough in-domain data. In that case, a smaller architecture learned from scratch can be more accurate and much cheaper.

Another implication has to do with the deployment. In this work, the training was done in less than two minutes using just a CPU. In this way, the system is usable for those who do not have GPU hardware.

Finally, it has to do with the benchmarking practices in sign language recognition. Accuracy difference between the proposed model and MobileNetV2 of 1.24 percentage points would look insignificant if reported alone. However, the fact that it is paired with the 63% reduction in computation costs significantly changes the recommendation. It means that cost reporting should become a regular benchmarking practice for sign language recognition.

The fourth implication has to do with accessibility and sustainability. If the assistive technology meant for Deaf communities in those regions that lack specific computing facilities needs to be sustained locally, then it needs to be trainable and run on regular computing hardware. The model that requires less than two minutes of central processing unit training time is one that can easily be retrained by a local institute whenever new information is made available. The second model that requires extended graphics processing unit training time makes such retraining impossible, and therefore, the efficiency advantage described above also implies sustainability implications beyond experimental considerations.

Finally, a general methodological consideration for any sign language recognition system can be drawn from the presented experiments. The conventional assumption that larger pretrained models will perform better in transfer learning does not seem to hold true in this particular case, and will not be discovered unless the transfer learning is complemented by from-scratch training of a model. Therefore, a purposefully designed baseline model should be included in comparative evaluations rather than relying only on the pretrained architectures.

4.5 Evaluation of the Deployed System

The trained model was implemented into the described real-time translation system (Section 3.8), which was tested in operational setting. It operated with interactive speeds, with latency that includes capturing, transferring, preprocessing, inferring, and displaying remaining quite short and sufficient to allow natural rhythm of signing. The word-building module successfully built words out of recognized letter combinations, and the stabilization mechanism performed well in preventing temporary misclassification of hands during switching from one sign to another.

Two practical considerations were discovered during the operational testing. First, recognition accuracy depends on the lighting and background condition due to the model being trained on imagery acquired in controlled environment. Second, the confidence score displayed next to predictions gives important feedback to the user as it indicates that hand position and frame need adjustment in case of persistently low score.

4.7 Answers to the Research Questions

The obtained results allow for providing direct answers to the three formulated research questions in Section 1.2. As for the first research question regarding the comparison of deep learning and traditional machine learning methods in the context of accuracy obtained when using the same dataset and identical conditions, the answer is clear: both of the two best performing convolutional neural networks scored higher than 93% in accuracy, whereas no traditional classifier reached an accuracy score higher than 81%. The difference of 14.63 percentage points between the proposed network and the best-performing traditional method confirms the superiority of the former and provides theoretical justification in Section 2.2.

As far as the second research question, related to the identification of the best-performing architecture in terms of its trade-offs between accuracy, speed of training, and feasibility of deployment, is concerned, the answer is straightforward: the proposed custom convolutional network. It proved to be the most accurate model among all seven architectures considered and required the shortest training time compared to other deep learning models. Additionally, the relatively small number of parameters of the proposed architecture makes possible inference at interactive speed on standard hardware. Of special note is the Random Forest architecture, which required the shortest training time – 1.90 seconds. However, the figure was achieved on the downsized sample and feature space and its accuracy of 80.25% was too low for practical implementation.

Concerning the third research question related to the comparison of the performance of the proposed network with transfer learning in the task of classification of grayscale sign imagery, the answer is positive under the present conditions. The proposed custom network performed better in terms of accuracy (outperforming MobileNetV2 by 1.24 percentage points) and required approximately 37% of its training time. This result can be explained by the mismatch between the domain of the ImageNet dataset (natural photography) and the domain of the used sign imagery (grayscale images of hands). The condition of limited data volume, under which transfer learning is expected to perform best, is no longer relevant since a sufficiently big dataset for in-domain training exists.

4.8 Limitations

There are several limitations to the results obtained. First, the research is limited to the consideration of static signs in Arabic manual alphabet and does not cover dynamic word-level signs or continuous signing, which requires a temporal modelling of multiple sequential frames. Second, all of the imagery was obtained from a single dataset, collected in a certain geographic region under controlled conditions.

Stratified random partition is used in the evaluation process, allowing images of the same signer to be included in both training and validation datasets, which means that it is not a very strict evaluation protocol for testing generalization ability of the models since it is not signer-independent. All deep learning models were trained using five epochs budget, which might be too

little for the deeper residual architecture to demonstrate its strength. Finally, evaluation is based only on validation accuracy and training time metrics. Calculation of per-class precision, recall, F1-score and confusion matrix would reveal signs that cause the most confusion for the algorithm.

5. Conclusion

In this research, a deep learning-based recognition system for Arabic Sign Language was developed and tested, along with the development of a controlled benchmark experiment for seven models using the publicly available Arabic Alphabets Sign Language dataset of 54,049 images. The custom convolutional network presented in the paper showed the best results of validation accuracy of 94.88% with minimal training time among other deep learning models, and thus was successfully incorporated in a real-time translation tool.

There are three main conclusions drawn from this experiment. First, deep learning performs much better in image classification task than traditional machine learning models for Arabic Sign Language, since the proposed model has exceeded the best of the latter by 14.63 percentage points; this superiority is caused by convolutional layers' ability to maintain spatial structure. Second, a compact purpose-designed architecture can outperform a substantially larger pretrained model when the target domain differs markedly from the source domain of pretraining, as demonstrated by the proposed network exceeding MobileNetV2 in accuracy while requiring approximately 37% of its training time. Third, having additional architectural depth does not always contribute to better performance in case of limited training budget, which can be seen from the 80.97% accuracy of Light ResNet.

Accordingly, there are three contributions of this research. First, it is a comparative benchmark experiment for seven models for the large public Arabic Sign Language dataset with unified evaluation protocol eliminating the confounding effect caused by aggregation of results obtained in separate experiments. Second, it shows computational cost of each model in addition to accuracy, giving practitioners enough information to make deployment decision. Third, this study provides not only accurate models but also a functional real-time translation prototype instead of just stopping after benchmarking.

Several recommendations for future work based on the results of this research are as follows. Integrating an Arabic text-to-speech module will close the communication cycle between deaf and hearing people allowing to deliver words recognized through sign language through sound; this is the most direct way to extend the current system. Using temporal architectures, recurrent or transformer-based, will enable moving beyond isolated letters recognition to full translation of conversations.

Use of a signer-independent protocol, where each person does not appear in the training and validation sets, will make the experiment substantially more rigorous and allow the comparison with other signer-independent experiments, such as that of Podder et al. (2023). Diversity of the dataset in terms of lighting, background, camera angle, and dialect variants of Arabic Sign Language will increase robustness in deployment on real-world devices, taking into account the issue with capture heterogeneity discussed in Section 3.2.

Redoing the benchmark with the use of hardware acceleration and longer training time will verify the stability of the architecture hierarchy in terms of performance found in this experiment and will test whether the Light ResNet performs better when allowed to converge. Finally, calculation of

class precision, recall, F1-score, and analysis of the confusion matrix will reveal which particular signs of Arabic Sign Language are confused, allowing data augmentation or architectural optimization for these classes instead of a homogeneous approach throughout the alphabet.

These directions define the research agenda for progressing beyond the fundamental ability demonstrated in this paper and developing a complete Arabic Sign Language translation system. The current work proves that the basic task is solvable with reasonable amount of resources available, which is the essential prerequisite for this agenda, and for the practitioners, working in resource-limited environments, an encouraging one.

References list

- Bani Baker, R. (2023). Image-based Arabic sign language recognition system using transfer deep learning models. *Applied Computational Intelligence and Soft Computing*, 2023, 5195007.
- Das, S. (2023). A deep sign language recognition system for Indian sign language. *Neural Computing and Applications*, 35, 1469-1481.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770-778.
- Hussain, A. (2023). Advances in Indian sign language recognition systems: Towards improved communication and accessibility for the deaf and hearing impaired. *Asian Journal of Engineering Sciences*, 12(2).
- Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. *Proceedings of the International Conference on Learning Representations*.
- Kumar, R. (2023). Real-time sign language recognition using Mediapipe and CNNs. *arXiv preprint arXiv:2305.05296*.
- Latif, G., Mohammad, N., Alghazo, J., AlKhalaf, R., & AlKhalaf, R. (2019). ArASL: Arabic alphabets sign language dataset. *Data in Brief*, 23, 103777.
- Noor, T. H. (2024). Real-time Arabic sign language recognition based on a hybrid deep learning model. *Sensors*, 24(11), 3683.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., & Grisel, O. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830.
- Podder, K. K., Ezeddin, M., Chowdhury, M. E. H., Sumon, M. S. I., Tahir, A. M., & Ayari, M. A. (2023). Deep learning-based signer independent Arabic sign language recognition system. *Sensors*, 23(16), 7156.
- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4510-4520). Institute of Electrical and Electronics Engineers.
- Srivastava, N., Hinton, G., Krizhevsky,