



Arabic Alphabet Sign Language Translator

Prepared By

Name: Yasir Aljarroof ID No.: 243001813

Submitted in fulfillment of the requirements for the degree Master of Science in
Artificial Intelligence

At

Collage of Informatics

Midocean University

UAE

Supervisor Name: Ashraf Elsayed

Collage of Informatics, Midocean University, Comoros

Email: 243001813@st.midocean.edu.km

Acknowledgements

Special thanks to Dr. Ashraf Elsayed for the support and guidance.

Abstract:

This research project aims to develop an AI-powered application for translating Arabic Alphabet Sign Language into text, addressing the communication challenges faced by the deaf community. The project focuses on creating a tool that can recognize and translate static sign language gestures accurately into Arabic text which will help bridge the communication barriers. The methodology involves collecting a dataset of Arabic Alphabet sign language images and implementing hand detection and gesture recognition to finally translate these into text. The developed model can recognize 32 of the Arabic sign Language including the 28 alphabet letters plus 4 more gestures. A hypered model was developed with a CNN pre-trained model (EfficientNetB0) that used the actual image to classify it on one end, and a fully connected model that uses landmarks points extracted from the image (extracted with mediapipe), with the hyper model achieving an accuracy of %98.

Background and motivation

The deaf community faces significant communication barriers in their daily interactions with hearing individuals. This tool will help the deaf community so that they don't have to rely on sign language interpreters to assist with communication which presents substantial challenges like the availability of sign language interpreters and the lack of automated translation tools. This project addresses this critical gap by using artificial intelligence and computer vision technologies to create an accessible and efficient translation system. The goal is to help deaf people communicate better and feel included while also improving AI for Arabic Alphabet sign language recognition.

Problem statement

The current lack of Arabic Alphabet Sign Language knowledge creates significant barriers for deaf individuals trying to communicate with hearing people. The challenge lies in developing an automated translation system that can accurately recognize and translate Arabic Alphabet Signs while being accessible, user-friendly, and practical for everyday use.

Research objectives

The research aims to develop an AI-powered system for accurate Arabic Alphabet Sign Language recognition and translation and design a user-friendly interface, achieving high accuracy in sign recognition while maintaining real-time processing capabilities.

Related Work

The advancements in deep learning have made huge progress, which helped in a lot of areas. One of these areas is the sign language recognition system, especially for Arabic and Saudi Sign Language , which is aiming to bridge communication gaps between deaf and hearing communities. Researchers have mostly used convolutional neural networks (CNNs) for static gesture classification, though hybrid models and real-world applications remain areas of active exploration.

Al-Obodi et al. (2020) [1] made a CNN-based system for Saudi Sign Language recognition, achieving high accuracy by training on 40 classes of static gestures, including 28 alphabetical signs, numbers 1-10 signs, and two more gestures for me “لنا” and “لا”. Their work demonstrated the viability of CNNs for isolated gestures. However, it had limitations including a restricted vocabulary and reliance on still images rather than continuous sentences. Similarly, El Rwelli et al. (2021) [4] developed a CNN-driven system to translate Arabic Sign Language (ArSL) gestures into vocalized speech, attaining 90% accuracy. However, their model’s training on only 30 sign letters highlighted a recurring challenge in sign language recognition: limited vocabulary scalability.

To address dynamic gestures, hybrid architectures combining CNNs with sequential models have emerged. Noor et al. (2024) [3] proposed a CNN-LSTM framework for real-time ArSL recognition. They have achieved 94.40% accuracy for static gestures, and 82.70% for dynamic ones. They have trained the models on 10 static words and 10 dynamic gestures. The results were great, but the dataset was limited, suggesting the need for larger datasets for both static and dynamic gestures. Al Khuzayem et al. (2024) [2] have developed a CNN-based SSL recognition model and made it into a mobile application, Efhhamni, which supports both static and dynamic gestures. Despite its innovation, the system required further testing in varied lighting and background conditions, as the data set is not available to see, and the sample provided had different people doing the signs, but all had the same lighting and background.

The efficacy of CNNs for static gestures is further supported by Samonte et al. (2022) [5], whose research and literature review of 20 sign language recognition studies have highlighted CNNs and Connectionist Temporal Classification (CTC) as dominant for static and dynamic recognition.

All together, these studies highlight CNN models robustness in static gesture recognition but reveal some challenges. like limited vocabulary, dynamic gesture recognition less than static performance.

Future work must prioritize larger, diverse datasets, and different environments.

Datasets Description

The dataset [6] we chose for this project is Arabic Sign Language Letters Dataset (ArSL21L). It is available on Kaggle by Munkhjargal Gochoo from United Arab Emirates University and it was Published on 11 February 2022. Its is made of 14,202 Images of 32 distinct Arabic sign language letter signs, including 28 arabic letters, and the following: ” لا , لا , لا , لا ”. This data set has various backgrounds, and was collected from 50 different subjects. The data set has been evaluated using the YOLOv5l model achieving high results.

Proposed Approach

The first step was **Data Preprocessing and Data Augmentation**. In this step, we first used mediapipe by google to crop the images around the hands, and also saved a copy of the hands landmarks to use with our model. Then we have resized the images to a consistent input size (224x224). Next we did Data Augmentation, applying rotations, horizontal flips, slight shifts, and slight zoom to images, and normalized the landmarks using StandardScaler in order to balance the dataset for better generalization.

Since CNN seems to be the most popular choice and has a high accuracy from previous studies, we chose to explore using CNN to learn distinctive features present in the dataset. We have tested a few pretrained models including resnet50 and efficientNetB0. I decided to go with efficientNetB0 as it was able to get to 97% accuracy. Then we tried training a dense neural network with the landmarks that were extracted with mediaPipe, which yielded great results (96%) in a very short time compared to training and tuning efficientNetB0. The next step was **Hyperparameter Tuning for both:** Experiment with the number of dense layers, learning rates, and dropout rates, and unfreezing layer from base model..

Experimental Results and Discussion

First, we used the dataset as it is since it had almost the same number of images across classes. The first model we tried was ResNet50, with base_model, GlobalAveragePooling2D, Dense layer (128), dropout of 0.5, then an output layer with softmax activation function with the 32 classes. Then we started fitting the model with a 0.0001 learning rate. After 50 epochs, the validation accuracy reached 57% and the model stopped improving.

After that I tried ResNet50, with base_model, GlobalAveragePooling2D, two dense layers (512 and 256), each followed by a dropout of 0.3, then an output layer with softmax activation function with

the 32 classes. After 115 epochs, it reached 64% validation accuracy then stopped improving. Then I tried unfreezing the layers of resnet50 to make the model learn and adopt more from our arabic sign language dataset with lower learning rate (0.00001). This has improved the model drastically, achieving 89% validation accuracy after 40 more epochs.

The next model we tried was EfficientNet B0 with the same set up as our last model. This was able to get the same results in less epochs. And after unfreezing 30% of the layers, it reached 91% validation accuracy.

After that I started to do more preprocessing to try to achieve better results. For this, I used mediapipe by google to detect the hand, and crop the image around it to make the image more focused on the hand and ignore the rest. With the new dataset (cropped images around hands) with EfficientNet B0, achieving 83% validation accuracy before unfreezing any layers. Then we proceeded with unfreezing layers little by little until we reached 97% validation accuracy.

The next step was to try another approach to this. When I used mediapipe, i also saved the landmarks of the detected hand (21 landmark * 3 coordinate each). So I used these as my dataset and fed them to a Dense Neural Network with 3 dense layers (256-128-64), each followed by a dropout layer (0.3-0.2-0.1), then an output layer with softmax activation function with the 32 classes. This was the fastest model to train since it's not actually an image, and the fastest to get great results reaching around 90% at the 7th epoch. At the end of the training, the model reached 96% validation accuracy which is very close to our results with EfficientNet B0.

While analyzing the result for both of these two models, I noticed that each model excelled in different parts. For example, EfficientNet B0 model had people distinguishing between “taa” and “dha”, but is better distinguishing between “kaaf” and “seen”. So I made a decision tree method that

will take the prediction of both of these models, and decide which one to pick based on the two predictions and the confidence of the predictions. By doing so, the final Hyperd model have reached an accuracy 98%.

References

1. Al-Obodi, A. H., Al-Hanine, A. M., Al-Harbi, K. N., Al-Dawas, M. S., & Al-Shargabi, A. A. (2020). A Saudi Sign Language Recognition System based on Convolutional Neural Networks. *Building Services Engineering Research and Technology*, 13(11), 3328–3334. <https://doi.org/10.37624/IJERT/13.11.2020.3328-3334>
2. Al Khuzayem, L., Shafi, S., Aljahdali, S., Alkhamesie, R., & Alzamzami, O. (2024). Efhamni: A Deep Learning-Based Saudi Sign Language Recognition Application. *Sensors*, 24(3112). <https://doi.org/10.3390/s24103112>
3. Noor, T. H., Noor, A., Alharbi, A. F., Faisal, A., Alrashidi, R., Alsaedi, A. S., Alharbi, G., Alsanoosy, T., & Alsaeedi, A. (2024). Real-Time Arabic Sign Language Recognition Using a Hybrid Deep Learning Model. *Sensors*, 24(3683). <https://doi.org/10.3390/s24113683>
4. El Rwelli, R., Shahin, O. R., & Taloba, A. I. (2021). Gesture-Based Arabic Sign Language Recognition for Impaired People Using CNN. *International Journal of Advanced Computer Science and Applications*, 12(12), 574–582. <https://doi.org/10.14569/IJACSA.2021.0121273>
5. Samonte, M. J. C., Guingab, C. J. M., Relayo, R. A., Sheng, M. J. C., & Tamayo, J. R. D. (2022). Using Deep Learning in Sign Language Translation to Text. *Proceedings of the International Conference on Industrial Engineering and Operations Management*, Istanbul, Turkey, 4036–4044.

6. Munkhjargal Gochoo (2022). Arabic Sign Language Dataset 2022. Kaggle. Retrieved from:
<https://www.kaggle.com/datasets/ammarsayedtaha/arabic-sign-language-dataset-2022>