

# Greek sign language recognition using machine learning

Mohamed Aqib Abid\*, Neetu Pillai and Nahsam Ahammed

University of West London, Ras Al-Khaimah - United Arab Emirates

Received: 10-September-2023; Revised: 08-February-2024; Accepted: 10-February-2024

©2024 Mohamed Aqib Abid et al. This is an open access article distributed under the Creative Commons Attribution (CC BY) License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

## Abstract

*This research develops a real-time Greek sign language (GSL) recognition system using machine learning and computer vision techniques to assist communication for the deaf community. The hand tracking module from the CVZone library was utilized for hand detection and tracking in video feeds. A dataset of 37,589 images capturing 24 Greek alphabet letter gestures was collected from different users under varying conditions. This dataset was used to train a deep neural network model based on the MobileNetV2 architecture. The model achieved 96.86% accuracy on the validation set during training. On the final test set, it obtained 95.66% accuracy in classifying and recognizing GSL hand gestures. Compared to prior rule-based and fuzzy clustering approaches, the model demonstrated significantly improved recognition performance. The high accuracy indicates that combining robust computer vision techniques for hand tracking with deep convolutional neural networks can be an effective approach for real-time GSL recognition. This system has the potential to facilitate communication and accessibility for the deaf and hard-of-hearing community. Further work should focus on expanding the vocabulary beyond individual letters to incorporate common words and phrases, as well as optimizing the system's real-time performance by reducing processing lags. Overall, this research demonstrates a promising machine learning and computer vision-based system for automatic GSL recognition that can aid the deaf community.*

## Keywords

*Greek sign language, Machine learning, Computer vision, Hand tracking, Real-time recognition.*

## 1.Introduction

Sign language involves gestures. Grammar and syntax of sign language is very complex to understand. Hence, sign language recognition (SLR) systems address this issue. Artificial intelligence (AI) and computer vision help develop SLR systems that automatically recognize and interpret sign language gestures and convert them into text or audio. Rule-based, machine learning-based, and deep learning-based methods can develop it. Due to massive datasets and improvements in deep learning-based methods, the systems have made significant progress. Sign language variation is a major issue due to regional dialects, signing styles, and context. Without standardized datasets or evaluation metrics, it is difficult to compare and benchmark SLR systems. "PeoplesGroup" reported that there are 62,500 deaf individuals in Greece [1]. Sign language is a nonverbal language characterized by specific hand gestures. The real challenge lies in the difficulty non-disabled individuals face with the grammar and rules of sign language.

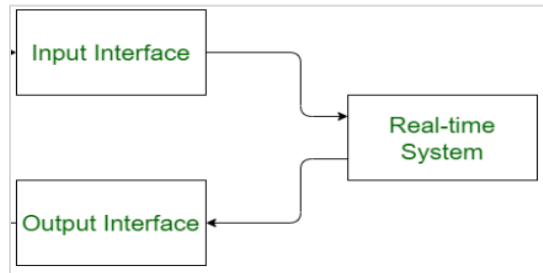
This situation inspired the development of automated sign language recognition (ASLR), which uses machine learning to detect and interpret sign language gestures into written language [2–5]. This technology makes life easier for disabled and able-bodied people.

Greek alphabet recognition employs ASLR to identify motions corresponding to Greek letters. Deep learning, convolutional, and recurrent neural networks are types of machine learning algorithms. Our model is trained using a convolutional neural network (CNN), which consists of layers made up of interconnected nodes known as neurons. These neurons process information and pass it on to the next layer.

Our labelled data is Greek alphabet hand gestures. The machine learning algorithm classifies hand gestures based on labels [2, 3]. Real-time models instantly process input data and output results. The proposed SLR system's real-time framework detects and translates hand gestures instantly. Hearing people and normal people can communicate without delay.

\*Author for correspondence

Input inference involves identifying and interpreting input data, such as a camera-captured hand gesture. The trained model processes the picture and extracts relevant characteristics to generate the output in the input inference phase. Output inference generates the final output from input data (*Figure 1*).



**Figure 1** A real-time model that takes input data and produces output

Real-time sign language identification can facilitate emergency communication and is utilized in education, healthcare, and social communication. SLR systems have enabled non-hearing-impaired individuals to learn sign language in educational settings [6, 7]. In healthcare, these systems bridge communication between doctors and hearing-impaired patients. For social communication, the same systems simplify daily interactions.

The development of SLR spans over 20 years, beginning with rule-based sign recognition. Modern technologies such as neural networks and support vector machines have introduced machine learning approaches. Numerous academic institutions and research initiatives are currently enhancing SLR systems for various applications.

Convolutional and recurrent neural networks are employed in SLR systems to enhance accuracy and resilience [8–10]. The Greek sign language (GSL) recognition system instantly improves accessibility for Greece's hearing-impaired population and aligns with current trends through its deep learning and computer vision models. Technological advancements may further reduce communication barriers for the deaf in the future. Effective communication is essential in all relationships; however, individuals who are mute cannot speak. Here, SLR becomes a powerful tool for communication.

The objective of this paper is to develop and assess a machine-learning-based GSL recognition system that employs image-based methodologies for hand gesture

recognition. This research approach emphasizes quantitative data collection and analysis to evaluate the model's efficacy and efficiency. The paper details the use of CNNs for training the recognition model and tests its performance by comparing its predictions against new datasets. The goal is to enhance communication accessibility for Greece's hearing-impaired population by leveraging advanced machine learning techniques and computer vision models.

This paper is arranged as follows. Review and analysis have been elaborated in section 2. Section 3 covers the methodology discussion. Results and discussion have been explored in section 4. It is concluded in section 5.

## 2. Literature review

In this section, several relevant studies are discussed. In the study conducted by Riad et al. [2], the SLR system enhances quality of life and accessibility. It translates gestures into signs. CNN classifies ASL in the study. After filtering, the hand image is passed through a classifier to predict hand gesture class. The Kaggle "ASL hand sign dataset" was the basis for this research article. 30526 images in 27 classes and 8958 images are used for training. The main aim of the research was to develop a SLR system that interpret hand gestures and convert them into signs, for American sign language (ASL). Performance and accuracy were assessed through a comprehensive 20-epoch analysis of both training and test data. The proposed model exhibited an impressive accuracy of 96.3% across the 26 alphabets.

Both our model and the study article's model shared a common objective of bridging communication gaps, albeit our model focused specifically on the Greek language, addressing the linguistic nuances unique to this context.

Advanced natural language processing algorithms would be essential for translating and generating complete ASL sentences [11–13]. The system's usability extends to sign language learners and members of the mute community. The SLR system's success emphasized the importance of a large dataset and CNN classification in achieving high SLR accuracy. Utilizing TensorFlow, OpenCV, and Keras facilitated the development of the SLR system. Techniques such as converting hand images to grayscale, Gaussian blur, and adaptive threshold processing were employed to enhance system performance. The implementation of a sign language

to text converter tool enabled real-time translation of sign language into letters and sentences. These findings contribute valuable insights for the creation and improvement of SLR systems.

Rule-based systems cannot recognize new gestures without updating their rules. It cannot handle multi-movement gestures either [9, 10]. This algorithm recognizes gestures without gloves or sensors [14, 10], making it easier and cheaper to use.

The research conducted by Wu et al. [4], focused on video processing-based SLR systems. System uses preprocessing, morphological transformations, background noise reduction, feature extraction, and fuzzy clustering. The proposed system blurs video frames to remove high-intensity noises. Morphological transformations remove edge noise from binary images. Contour detection isolates the face, left hand, and right hand. These areas provide vector characteristics for video frames. The fuzzy c-means (FCM) clustering algorithm clusters data based on the chosen criteria. The algorithm compares the test file with the existing clusters and identifies the cluster center that has the highest membership. Algorithms group input data into clusters. The gestural labeling accuracy was 75%.

However, a notable limitation surfaced when utilizing fuzzy clustering, particularly in instances where clusters lacked proper initialization. The imperative arises for a system capable of efficiently and accurately managing large datasets, although the fuzzy clustering approach proved adept only with simpler datasets. In contrast, our CNN model demonstrated proficiency in recognizing and distinguishing complex gestures, a capability not as effectively mirrored by the fuzzy clustering method, especially in the context of multi-sign gestures.

The accuracy of SLR systems hinges significantly on the approach to feature extraction. Notably, optimal results were obtained through edge detection and motion tracking. The impact of lighting and background variations on SLR accuracy is substantial. Techniques such as adaptive thresholding and background removal prove instrumental in enhancing accuracy, particularly in challenging situations.

These findings indicate the critical role of feature extraction strategies and SLR system techniques that account for variations in lighting and background

conditions. FCM clustering emerges as a method adept at handling data ambiguity, clustering data items based on similarity. The application of FCM to clustering gestures significantly enhances classification accuracy. Its resilience to data noise and outliers further fortifies the identification system. The amalgamation of these methodologies holds the potential to elevate the accuracy and reliability of GSL recognition.

In the study conducted by Rekha et al. [5], preprocessing, segmentation, feature extraction, selection, and classification are involved. Preprocessing involves synchronizing IMU and sEMG data for fusion and timestamping it with the personal computer (PC) clock. sEMG signals and automated segmentation for real-time applications are highlighted. Paper suggests many sEMG and IMU sensor features. Only the most suitable are chosen. This is done to reduce overfitting and information redundancy. Decision tree, support vector machine, nearest neighbor, and naïve Bayes are used. LibSVM's radial basis function (RBF) kernel was selected using Weka's grid search to find the best kernel parameters [15–17]. The experiment setup, with four primary muscle group sensors on the right forearm and the IMU sensor on the wrist. The system only analyzes right hand movements for one-hand and two-hand signals, and it can be used with two hands to improve identification precision [18, 19].

The limitation was that only a few participants tested this proposed system, and the testing was confined to laboratory settings. Our model differs significantly. It utilizes visual input, unlike the proposed system that relies on sEMG and IMU sensors. Our model only requires a camera, a component ubiquitous in all modern devices, making it more practical for real-time situations. The proposed system achieved an accuracy of over 90%. Among the four classification algorithms evaluated, the RBF kernel-based support vector machine (SVM) emerged as the most effective.

A drawback of the proposed system is that sEMG can detect changes in hand position and identify indications, even though the wearable inertial sensor may yield the same results for multiple signs with the same arm movement. The system exclusively analyzes right hand movements for one-hand and two-hand signals, rendering it unsuitable for certain applications. Incorporating real-time data synchronization, automatic segmentation, feature selection to avoid overfitting, and the exploration of

various classification methods are potential advantages that could be integrated into our model.

In a study conducted by Huu and Phung in 2021 [6], the ASL research proposes two methods. The first method identifies the hand in each frame using skin color segmentation and k-means clustering. Another retrieves hand gesture features using speeded up robust features and hu moment invariant features. These two traits allow letter recognition, and database images feature vectors are stored. k-nearest neighbors (KNN) and SVM classifiers classify these feature vectors to recognize single letters. The suggested method combines KNN and SVM classifier output, a hybrid classifier architecture, with an optimized feature set to improve recognition rates in less time. Finger-spelled word recognition using a lexicon-based HMM. HMMs identify words made from identified letters. The suggested method uses multiple strategies to identify hand motions in real time. But combination of SURF and Hu Moment Invariant features may not be the best option for all hand gestures, the skin color segmentation method may not work well in all lighting scenarios, and the lexicon-based hidden Markov model (HMM) approach proposed for recognizing finger-spelled words may not be enough for more complex signs. These significantly reduce hand detection accuracy. They could record videos of the same gesture and words in different lighting conditions. Recorded videos can be processed using paper methods. Recognition accuracy can be compared for each lighting condition to determine which require more preprocessing or algorithm modifications for precise recognition [20–25]. In the first study, an examination was conducted on a SLR system, which converted hand gestures into signs utilizing image processing and CNN-based classification, achieving a recognition accuracy of 96.3% for 26 alphabets. The second paper outlined a rule-based hand recognition system that recognized gestures through feature extraction and IF-THEN rules [26–28]. An experiment was conducted to determine the system's threshold for dataset gesture identification. The third study employed unsupervised learning, morphological adjustments, feature extraction, fuzzy grouping, and preprocessing to classify hand gestures. FCM clustering was utilized to sort data by face, left hand, and right-hand vectors, achieving a 75% correct labeling of gestures [29–34].

The studies indicate the importance of selecting optimal methods and algorithms for specific

application domains and ensuring effective data processing through feature selection and reduction.

### 3.Methods

Two methodologies were employed in developing the model used in this study: image-based and sensor-based. The image-based approach was favored as the cornerstone of our strategy, surpassing sensor-based alternatives. This preference stemmed from the image-based approach's inherent advantages, characterized by its cost-effectiveness, elimination of the need for additional equipment, and practicality for daily usage. Conversely, the sensor-based approach was deemed less favorable due to its inherent drawbacks, including high hardware costs, fragility, and maintenance requirements. Moreover, the image-based approach capitalized on the ubiquity of cameras integrated into modern smartphones and gadgets, further emphasizing its practicality and accessibility.

Rule-based and machine-learning methods classify hand gestures. We chose machine learning because the rule-based technique requires manually encoding feature input rules, which takes time and requires an in-depth knowledge of all relevant hand features to interpret the output. Machine learning uses intelligence to teach computers through experience [35–40]. Second, rule-based strategies do not allow learning. New data may lead to inaccurate results. However, machine learning can improve its accuracy with new input. Thirdly, machine learning can implement complex and nonlinear correlations between features, unlike rule-based systems.

#### 3.1Research approach

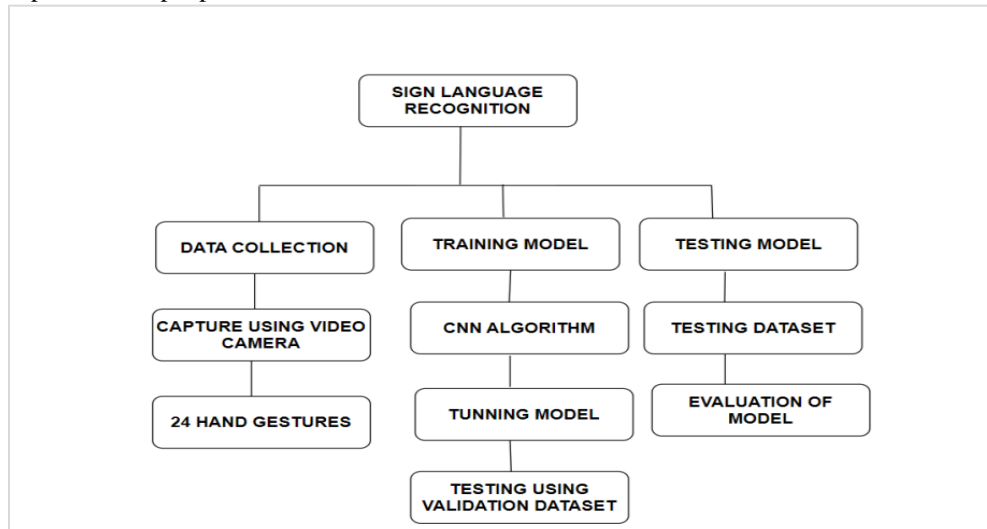
Our SLR system research is quantitative. It involves collecting numerical data and finding patterns and relationships. Three stages comprise the project. The process begins with video camera data collection of 24 Greek alphabets (Figure 2). After collection, data will be tagged to ensure model reliability. Second, CNN algorithm trains the model. Finally, the system's performance is tested by collecting new datasets and comparing them to the model's predictions.

#### 3.2Justification

This project established a machine-learning GSL recognition system. Quantitative data collection and analysis were employed to evaluate the efficacy and efficiency of the model, ensuring unbiased qualitative research. Accurate data was imperative for constructing a robust system, and the analysis revealed patterns, trends, and relationships crucial for

understanding sign language movements. This approach also facilitated replication and comparison to previous findings. In contrast, qualitative research explores into people's core beliefs and motivations.

Quantitative analysis was not chosen due to its lack of numerical data, making the evaluation of results challenging [41–45].



**Figure 2** Block diagram of system

### 3.3 System requirements

Conducting tests on the GSL recognition system within the PyCharm integrated development environment (IDE) and Google Teachable Machine demanded strict adherence to specific system requirements. The experimental setup utilized a Microsoft Windows 11 Home operating system on an Acer laptop featuring a 1.80GHz Intel Core i7-8550U processor with 4 cores and 8 logical processors. To optimize the testing procedure, the laptop was exclusively dedicated to executing testing scripts, promptly terminating resource-intensive processes. This precautionary measure aimed to mitigate potential interferences, ensuring the reliability of test outcomes [46, 47]. Furthermore, the notebook computer was intricately connected to a dependable power supply to fortify the system against abrupt power outages.

Both the PyCharm IDE and Google Teachable machine, being advanced tools, prescribe precise system prerequisites for optimal functionality. PyCharm demands a 1 GHz processor, 4 GB of RAM, and 2.5 GB of hard disk space, while Google teachable machine has its own set of requirements. It is noteworthy that the laptop utilized in this experiment surpassed the stipulated requirements for both tools, contributing to a seamless and effective testing process.

### 3.4 Selected hand gesture recognition libraries

The "cvzone.HandTrackingModule" library detected hands, incorporating a HandDetector class equipped with a pre-trained model for real-time hand recognition in video streams. This class tracked hands, identified gestures, and performed other tasks through computer vision [48, 49]. The library's parameters, such as MaxHands, detectionCon, and trackCon, were configured based on the application's requirements. These parameters managed the minimum confidence threshold for hand recognition and tracking. HandDetector utilized the open-source, cross-platform MediaPipe framework, known for creating multimodel machine learning models. The framework was also compatible with mobile and embedded systems.

### 3.5 The corpus





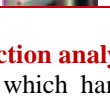
PyCharm incorporated various hand-tracking libraries based on OpenCV/Mediapipe, with histogram of oriented gradients (HOG+SVM) and HandTrackingModule forming the corpus. In our project, a pilot study was conducted to compare these libraries, emphasizing the need for accurate hand movement tracking across diverse lighting conditions, hand positions, and gestures. Notably, the HandTrackingModule emerged as the most effective choice [50]. The study involved participants of varying ages and genders, engaging in the Alpha gesture against a neutral background to ensure consistency. Each gesture was meticulously recorded as a series of 10 JPEG images, each sized at 400×400

pixels. *Table 1* in the report enumerates the Alpha gestures analyzed during the study.

Results demonstrated the superior performance of the HandTrackingModule, particularly when hands were well-lit, and fingers were spread out. Its tracking accuracy surpassed that of HOG+SVM. Furthermore, the HandTrackingModule showcased versatility by

seamlessly integrating with other machine learning libraries, deep learning frameworks, and low-powered devices such as mobile phones and Raspberry Pi. This highlighted its strength as a versatile and adaptable solution for hand tracking and gesture recognition. Conversely, HOG+SVM, as a traditional method, was limited to detecting only one hand.

**Table 1** Corpus-selected images

Image name	Gesture	Image size	Resolution	Gender	Age group
Alpha		30- 35KB	400×400px	Male	Above>18
Alpha		30- 35KB	400×400px	Male	Below>14
Alpha		30- 35KB	400×400px	Male	Above>50
Alpha		30- 35KB	400×400px	Female	Above>18
Alpha		30- 35KB	400×400px	Female	Above>40

### 3.6 Data collection analysis

To determine which hand tracking method worked best, they were compared. The study used *Table 2* images to evaluate accuracy, detection, and

identification times for both methods. Performance measures determined the best hand tracking strategy. The table shows how both methods work.

**Table 2** Method comparison

Details	HOG+SVM	cvzone.HandTrackingModule
Landmarks	Hand landmarks, hand position	Hand landmarks, hand position
Hand details	Hand details (e.g. orientation)	Hand details (e.g. orientation)
Feature extraction	Hand features (HOG descriptors)	Hand features (e.g. fingertip position)
Gesture recognition	SVM classifier	Neural network
Confidence	Prediction confidence score	Prediction confidence score
Extra information	Nil	Number of hands detected

To begin the analysis, the necessary libraries were imported, including OpenCV, NumPy, cvzone2, and time. Following this, the HOG feature descriptor and SVM classifier were initialized. The SVM classifies data based on features, while the HOG descriptor characterizes image edges. The images from the 'alpha' dataset were loaded for comparison, with the program looping through the directory and adding each file to the dataset list.

Subsequently, a list was established to document the performance metrics of the hand gesture recognition system. This included factors such as the time taken

to detect and recognize hand gestures, the number of correct predictions, and the total count of hand gestures in the dataset. The cvzone2.handtracking module was then initiated to facilitate the subsequent steps. The system's process involved identifying hands using the HOG+SVM method. Initially, a pre-trained model capable of identifying hands was loaded, and its detector was configured to scan images using sliding windows. For each window, a feature vector was calculated using HOG, and if the prediction strength was sufficient, that portion of the image was recognized as a hand.



Moving forward, the system evaluated hand gestures by analyzing landmarks through the MediaPipe Hands API. Considerations such as gender and age were considered to categorize gestures, resulting in labels like 'Alpha\_below18\_male' or 'Alpha\_above18\_female' based on the user's profile. A function was employed to determine gestures from image filenames by extracting the first part before an underscore.

After detecting hands and understanding gestures, the process involved further analysis of these detected hand regions. Feature vectors were extracted from these regions using HOG, representing the unique qualities of each hand. Ultimately, the system predicted hand gestures based on these extracted features using a pre-trained SVM model. If all detected hand regions matched a certain gesture category (e.g., 'Alpha'), the system concluded that the user was making that gesture. If not, all hand regions

matched, the system remained unsure about the gesture. This approach was then compared with another method utilizing deep learning (cv2zone.handtracking) for hand detection. Both methods underwent evaluation based on their accuracy and the time taken to process images. HOG+SVM extracted features from detected hands, while cvzone2.handtracking predicted gestures using landmarks. The time and percentage of accurate predictions for each approach in identifying movements and detecting hands were recorded across a series of images. Upon comparing the results, it was determined that the cvzone2.handtracking module was more suitable for our project.

### 3.7 Data collection method

Our analysis found the best hand gesture recognition module. Thus, cvzone.handtracking module collects datasets (*Table 3*).

**Table 3** Data collection methods

Steps	Description
Identify gestures	Research the standard ASL signs for each letter and create a list of the corresponding hand gestures for the system.
Capture images	Use a web camera to capture 1000-1400 images of each gesture.
Extract features	Utilize the cvzone.HandTrackingModule to detect and extract the relevant features of each hand gesture.
Label data	Manually label each gesture with its corresponding Greek letter.
Split data	Split the data into training and testing sets. Aim for a ratio of 70:30 for training and testing respectively.
Address class imbalance	Collect additional data to address class imbalance
Store data	The information is kept in a directory that has 24 further sub-directories. JPEG is the file type used to store the image.

The GSLR necessitated reliable data collection, which commenced by defining 24 distinct Greek letter gestures. To ensure dataset diversity, a webcam was utilized to capture 1000–1400 photos of each gesture, encompassing different age groups, angles, and lighting conditions. The HandTrackingModule was employed to extract the requisite features, including the 21 hand points associated with each gesture. Manual tagging with the corresponding Greek letter facilitated labeling for each gesture.

The collected data was then partitioned into training, testing, and validation sets with a ratio of 70:20:10. To address class imbalance, additional data was gathered. The dataset directory contained 24 subdirectories, and images were stored in JPEG format. Employing supervised learning, the machine learning system was trained to recognize hand

motions. The validity of the model was assessed using the validation set, and subsequently, the testing set was utilized for model evaluation. Continual efforts were made to enhance accuracy and performance by refining the data gathering method and machine learning model. This iterative process aimed to ensure the accuracy and reliability of the machine learning-based recognition system.

### 3.8 Software testing

During our project, the optimal approach to testing was found in the combination of unit and integration testing. Unit testing, focusing on individual functions and methods, served to guarantee the overall functionality of the system by systematically examining test inputs and outputs. This meticulous process proved instrumental in identifying and addressing development issues.

Simultaneously, integration testing was employed to elucidate the interactions among different system components, streamlining the coordination of the entire system. Specifically, our GSL decoding system underwent rigorous testing, emphasizing hand gesture detection and identification.

The imperative of meeting user needs was indicated through acceptance testing, where actual users participated to assess system usability. Testing methodologies encompassed both black-box and white-box methods. White-box testing, involving an understanding of the system's internal structure and code, was juxtaposed with black-box testing, which focused on system behavior without knowledge of its internal workings. The amalgamation of these testing methodologies enhanced the thoroughness of the testing procedure.

This approach, rigorously implemented during the testing phase, ensured the reliability, accuracy, and user satisfaction of our system. The incorporation of well-established testing frameworks, namely unittest, pytest, and doctest, facilitated the comprehensive execution of the testing process.

### 3.9 Design and implementation

The gesture training module was utilized to train the recognition model using labeled gesture data from the database. The language translation module transformed gestures into words or text for meaningful conversation. The system controller managed module interactions, maintained data flow, and ensured the smooth operation of the gesture recognition system.

#### 3.9.1 Dataset

There were 24 different Greek alphabets in total. The Greek alphabets were presented in both uppercase and lowercase. Uppercase: A, B, Γ, Δ, E, Z, H, Θ, I, K, Λ, M, N, Ξ, O, Π, P, Σ, T, Y, Φ, X, Ψ, Ω. Lowercase: α, β, γ, δ, ε, ζ, η, θ, ι, κ, λ, μ, ν, ξ, ο, π, ρ, σ, τ, υ, φ, χ, ψ, ω. GSL represented the alphabets, where fingerspelling mirrored the Greek alphabet using hand gestures. Each sign was formed by creating a hand shape resembling a letter.

The system's popularity stemmed from its standardization, guided by established guidelines from the Deaf Organization. The rules of this system were formulated to be readily embraced and found comfortable by the hard of hearing individuals.

Our model utilized GSL hand gestures (Figure 3). Each Greek alphabet was represented by one hand,

and our training encompassed both left and right hands. Users had the flexibility to portray hand motions using either their left or right hand, based on their preference. The dataset consisted of images captured by the webcam, showcasing hand gestures corresponding to the Greek alphabet.

A dataset comprises labels and features that describe its classes or results, and a well-constructed dataset should contain sufficient data to yield the desired outcome. In the case of supervised learning, it is recommended to have 100–1000 images per hand pose. For our model, 1000–1400 photos were gathered for each Greek letter. The cvzone.HandTrackingModule was employed to identify 21 hand points, facilitating the recognition of hand positions from the input images. This module played a crucial role in enabling the model to distinguish various hand motions and predict outputs accordingly.



**Figure 3** OMKE-approved GSL adapted from “GSL” Speak Greek available at <https://www.speak-greek.com/p/greek-sign-language.html>

#### 3.9.2 Data collection

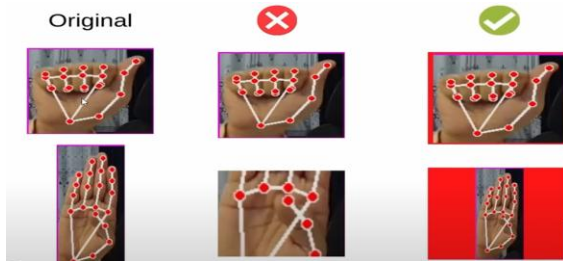
The initial step is data collection, where the hand is recognized as an object and cropped into an image that is used for training.

The primary concern was the influence of image height and width on hand postures. Classification necessitated a square image, prompting the need for image cropping. However, cropping the image posed



a challenge, as it might not reveal all 21 landmarks or points, resulting in the loss of significant information.

The solution is to use an image with a set pixel size of 300×300 (square) and whatever size the image was captured. It gets added to the center of another fixed image. If the image is not square, it is adjusted and placed in the center to ensure data consistency. *Figure 4* illustrates the situation. *Figure 4* depicts alpha and beta. The cropped photos for both scenarios and the margin adjustments are done to center them in the fixed image. Thus, helping classifier in classification.



**Figure 4** Problem and solution

A total of 37,589 photos were taken to train the hand recognition system to detect hands in various situations and improve the model's accuracy.

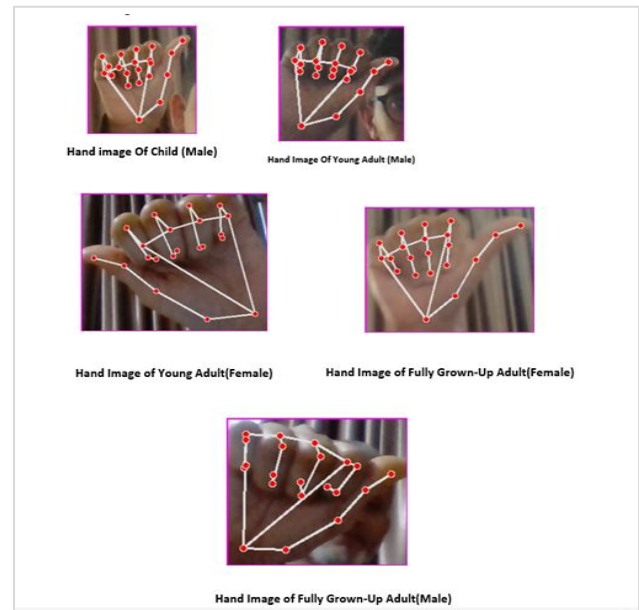
This ensured that the model detected all hand sizes. Without a size range, the system might have failed to recognize hands outside the specified limits. The images collected encompassed a diverse range, including young children to adults. Consequently, this broad size spectrum guaranteed that the hand identification system could discern hand characteristics irrespective of size.

Male and female hands are being collected. This is significant because male and female hands differ. This is done to accurately depict male and female hand gestures and collect diverse data. *Figure 5* shows age-group and gender-specific images.

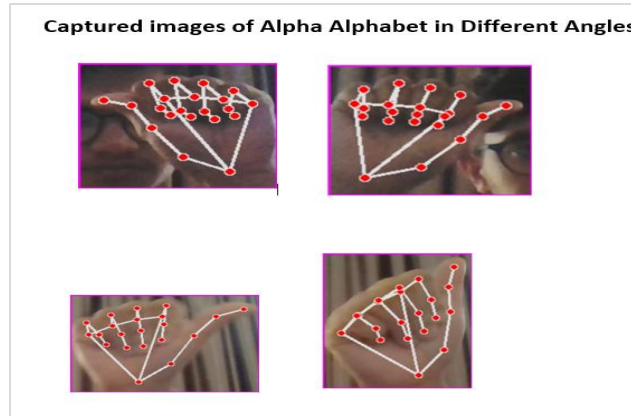
Images are taken from different lighting and angles as it allowed for the capture of the hand's features from multiple angles, aiding in the understanding of finger curvature and skin texture. It facilitated the model in recognizing hand gestures from any direction. To ensure the accurate detection of Greek alphabet hand gestures under diverse lighting conditions, images were captured in various lighting scenarios. Recognizing the impact of lighting, shadows, and reflections on the hand's appearance,

the system underwent training with images to detect hand traits consistently, irrespective of lighting conditions. This is shown in *Figure 6* and *7*.

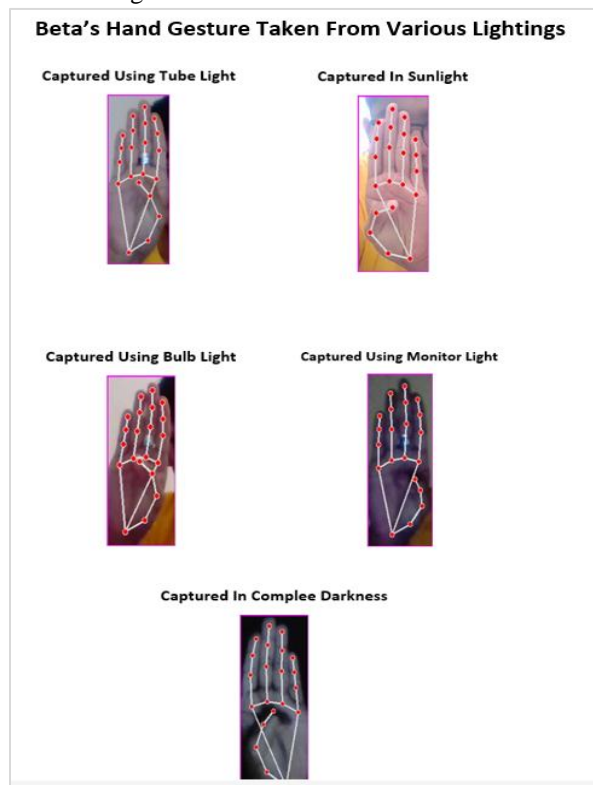
To attain better results, high-quality images were imperative. Blurred images containing noise posed challenges in identifying hand characteristics, subsequently diminishing model accuracy. Consequently, efforts were directed towards identifying and eliminating hazy images, as they could detrimentally impact the hand recognition system. During the collection of the dataset, mislabeling of many finger points occurred, posing a potential threat to the hand recognition system. Following each alphabet dataset collection, a meticulous quality control check was implemented to reduce errors and address the problem. To prevent the loss of crucial hand information, it was imperative to remove cropped photos. Retaining such images would compromise size, form, and position characteristics, thereby making hand identification difficult or even impossible. Hence, a systematic elimination process was executed, as depicted in *Figure 8*.



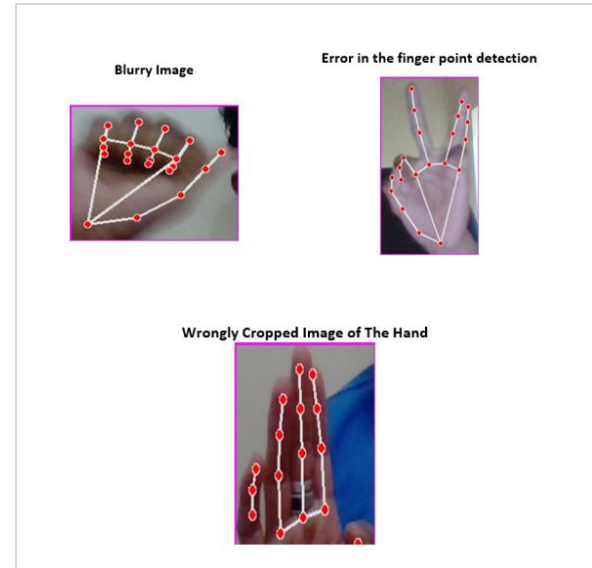
**Figure 5** Collected images of various types



**Figure 6** Captured images of alpha alphabet in different angles

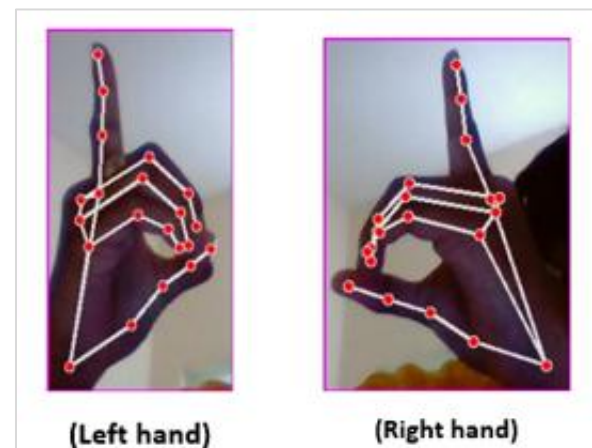


**Figure 7** Different lightings



**Figure 8** Blurry images

Due to the use of both hands by real people, the dataset included both hands to ensure the system could identify and convert gestures into a Greek alphabet, regardless of the user's chosen hand. Additionally, as they had mirror-image hands, incorporating data from both hands aided the system in leveraging symmetry to enhance reliability, as illustrated in *Figure 9*.



**Figure 9** Both hands

### 3.9.3Scripts

The project employed two scripts, Python scripts that could be executed from the PyCharm IDE to perform specific tasks. When a script was created in PyCharm, it resulted in the generation of a .py file where Python code could be written, edited, and executed.

The script `Datacollection.py` was responsible for organizing Greek letter images for the training of the machine learning model and computer vision tasks. The images were systematically stored in a "data" directory, with 24 subdirectories dedicated to each Greek letter. The collection and organization of Greek alphabet images within these subdirectories were facilitated using OpenCV and other relevant libraries. This script played a crucial role in image organization and contributed to the model-building process.

On the other hand, the `test.py` file was designed to evaluate the algorithm of a hand detection system. It involved loading a pre-trained hand detection model, importing necessary libraries and modules, and tracking hands in input images. This file executed various tasks:

**Loading the hand detection model:** This involved importing libraries and modules, loading a pre-trained model, and configuring model parameters.

**Processing input videos:** The script detected hands in input videos, followed by pre-processing to prepare the input data for analysis.

**Identifying hands in input data:** The `test.py` file could identify hands and track their movements using a pre-trained hand identification model. It also incorporated image processing methods such as filtering, segmentation, and feature extraction.

**Outputting results:** The `test.py` file generated outputs detailing hand detection algorithm results, including hand position, motion, and detection confidence scores. These scripts, particularly `Datacollection.py` and `test.py`, were integral components of the project, each serving specific roles in data organization, model training, and algorithm evaluation.

### 3.10 Working

For the model completion it is split into two phases, namely: Detection phase and classification phase.

#### 3.10.1 Detection phase

A hand detection technique identified image regions, located the hand, and determined the classes to which each hand pose belonged. This process supplied essential data for the later stages of the pipeline, underscoring its critical role. Accurate hand detection and feature extraction were imperative for SLR.

#### Setting up webcam

The `cv2` package processed images and videos

utilizing computer vision techniques. This package facilitated reading, writing, filtering, and thresholding of images and videos. It played a crucial role in constructing a video capture object for our project, enabling the reading of frames from the default camera.

#### Finding hands

The `HandDetector` class from the `cvzone` package was used to recognize and track hand landmarks such as fingers, knuckles, and wrists in videos. This data was then utilized for performing hand gesture recognition.

#### Cropping images

Cropping entailed selecting a rectangle from the original image, a crucial step during the model's construction. This process was essential for isolating the hand from the background and eliminating any noise or artifacts, including shadows, reflections, or motion blur, from the original image. By doing so, the model became more robust against lighting and environmental factors. During the model's development, the image was cropped to streamline data analysis. This efficiency improvement resulted from the need to process only a small portion of the image, enhancing the overall efficiency of image processing.

#### Custom image

Hand gestures vary while cropping, resulting in images of different sizes. For classification purposes, all images must be square. This is achieved by creating a matrix. A 300×300 white image is created for this purpose.

#### Overlay

The hand sign detecting system employed a process of cropping the hand and overlaying it onto a 300 × 300 white image. Overlaying involved combining two images into one, with cropped images strategically positioned over white backgrounds. This approach aimed to provide the classifier with a uniform input image, ensuring consistency and a fixed size for the hand image on the white background. The standardization of input images played a crucial role in facilitating the classifier's ability to identify and categorize hand signs.

To achieve this, the aspect ratio of the cropped image was calculated, and subsequent adjustments were made either horizontally or vertically based on the aspect ratio to fit within a 300×300 white background image. This process, as described, was implemented

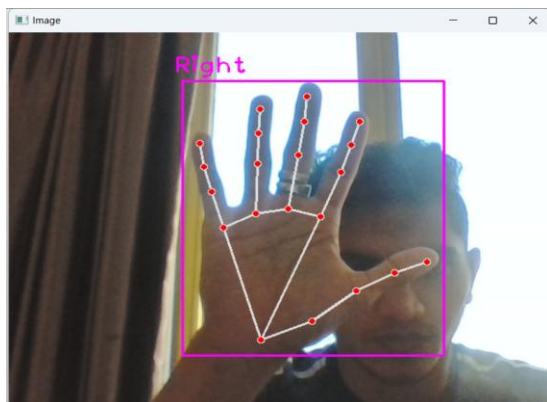
to optimize the input images for the classifier's recognition and categorization of hand signs.

### **Saving image**

There are 24 alphabetical folders. Folder variable is updated whenever we save an alphabet. The counter variable is incremented when an image is saved. Each filename will be unique even if multiple images are taken at once.

### **3.10.2 Classification phase**

The hand sign detection system employed a process of cropping the hand and overlaying it onto a  $300 \times 300$  white image. Overlaying involved the combination of two images into one, strategically positioning cropped images over white backgrounds. This approach aimed to furnish the classifier with a uniform input image, ensuring both consistency and a fixed size for the hand image against the white background. The standardization of input images played a crucial role in facilitating the classifier's ability to identify and categorize hand signs. To achieve this, the aspect ratio of the cropped image was calculated, and subsequent adjustments were made either horizontally or vertically based on the aspect ratio to fit within a  $300 \times 300$  white background image. This process, as described, was implemented to optimize the input images for the classifier's recognition and categorization of hand signs (Figure 10).



**Figure 10** Output hand

## **4. Results and Discussion**

### **4.1 Model's accuracy testing**

To assess the model's accuracy, the true labels of the test images were compared with the predicted labels. A Boolean array was generated, with a value of 1 assigned for every correct prediction and 0 for every incorrect prediction.

The current model's accuracy is 80.24%. To improve this accuracy, hyperparameters were modified and a new dataset was included. The following sections discuss the steps taken to increase the model's accuracy.

### **4.2 Enhancing accuracy**

#### **4.2.1 Created training and validation dataset**

Increased the number of training and validation dataset (Figure 11).

#### **Output(train\_generator,valid\_genarator)**

```
Found 37589 images belonging to 24 classes.
Found 9385 images belonging to 24 classes.
```

**Figure 12** Training and validation data

#### **4.2.2 Loaded predefined model**

The MobileNetV2 architecture was imported, representing a convolutional neural network architecture specifically designed for mobile and edge devices. An enhancement over the original MobileNet, this architecture offered improved performance while maintaining low latency and a compact model size. MobileNetV2 found particular suitability for tasks such as image classification, object detection, and image segmentation on resource-constrained devices. It effectively balanced model size, speed, and accuracy, making it well-suited for deployment in mobile applications and edge devices. Over time, it gained popularity in the field of computer vision, particularly for on-device inference.

#### **4.2.3 compiling model**

Adam, an abbreviation for Adaptive Moment Estimation, served as an optimization algorithm employed in the training of artificial neural networks. It represented an extension of the stochastic gradient descent (SGD) optimization algorithm, incorporating the computation of adaptive learning rates for each parameter. Adam amalgamated concepts from two other widely-used optimization algorithms: RMSprop and Momentum.

#### **4.2.4 Training model**

In each epoch, the model received a batch of training data and updated its parameters to minimize the loss function, assessing the alignment between the model's predictions and the training data. Accuracy metrics, represented as percentages, illustrated the frequency with which the model's predictions aligned with the actual values of the training data. Higher accuracy indicated superior model performance.



#### 4.2.5 Determining validation accuracy

Validation accuracy was employed to assess the effectiveness of the model during training and to identify overfitting by comparing the model to a validation set. A higher validation accuracy signified improved predictions on unknown data. The loss function, indicating the proximity of the model's predictions to the training data, needed to be

minimized as much as possible to enhance the model's performance.

Table 4 shows the result analysis considering precision, recall and F1-Score using different hyperparameters.

**Table 4** Results of precision, recall and F1-Score using different hyperparameters

	Precision	Recall	F1-Score	Support
Alpha	0.98	1.00	0.99	250
Beta	0.97	1.00	0.99	250
Chi	0.99	1.00	1.00	250
Delta	1.00	0.96	0.98	250
Epsilon	0.99	1.00	1.00	250
Eta	1.00	1.00	1.00	250
Gamma	1.00	1.00	1.00	250
Iota	0.88	0.90	0.89	250
Kappa	0.81	1.00	0.89	250
Lambda	0.85	1.00	0.92	250
Mu	0.96	0.66	0.78	250
Nu	1.00	1.00	1.00	250
Omega	0.97	0.96	0.96	250
Omicron	0.98	1.00	0.99	250
Phi	1.00	1.00	1.00	250
Pi	1.00	0.76	0.86	250
Psi	0.99	0.97	0.98	250
Rho	1.00	1.00	1.00	250
Sigma	1.00	1.00	1.00	250
Tau	0.74	1.00	0.85	250
Theta	1.00	1.00	1.00	250
Upsilon	1.00	0.98	0.99	250
Xi	0.98	0.89	0.93	244
Zeta	1.00	0.88	0.93	250
accuracy	-	-	0.96	5994
macro avg	0.96	0.96	0.96	5994
weighted avg	0.96	0.96	0.96	5994

#### 4.3 Discussion and analysis

The decision to opt for cv2.handtracking over HOG+SVM was based on several factors. The hand tracking module exhibited faster and more precise detection compared to HOG+SVM. In HOG+SVM, the average time for recognition and detection was 0.0138 seconds and 0.0315 seconds, whereas in the cv2.handtracking module, it was 0.0058 and 0.0217 seconds. Additionally, the accuracy of "handtracking" was 0.92, surpassing HOG+SVM, which had an accuracy of 0.85. The cv2.handtracking module utilized advanced algorithms and pre-trained models, ensuring accurate and reliable performance across varying lighting levels. This feature facilitated real-world gesture detection, enhancing the system's usability and overall performance. The classifier's proficiency in detecting and recognizing gestures in

diverse illumination settings significantly contributed to improving the user experience and achieving project goals.

Our GSL recognition system demonstrated superiority over numerous models from literature reviews. Notably, our model achieved a validation accuracy of 96.86% and a final accuracy of 95.66%, outperforming fuzzy clustering, which managed only 75% accuracy on small datasets. In contrast to rule-based models that struggle with new or complex hand gestures without rule updates, our model overcame this limitation by adapting to new gestures and varying weights. This adaptability allowed it to recognize complex patterns and characteristics in images, making it more ready for real-world scenarios.



Furthermore, our model exhibited faster recognition times compared to rule-based systems, with a processing time of 0.0058 seconds as opposed to 1.38 seconds. While a sensor-based model demonstrated higher accuracy, its impracticality for real-world use due to additional sensor hardware requirements made our camera-based model more viable. The universality of cameras in smartphones and electronics rendered our model accessible and less cumbersome than wearing additional gloves with sensors.

Lastly, our model addressed the limitations of Hu Moment Invariants and speeded-up robust features (SURF) in various lighting conditions. The dataset, captured from different angles and lighting conditions, contributed to our model's ability to outperform these limitations across diverse lighting scenarios.

The initial evaluation of the model on a dataset of 12,000 samples yielded an accuracy of 80.24%. The resulting classification report presented a comprehensive breakdown of precision, recall, and F1-Score for each class, indicating varying performance across distinct categories. Upon expanding the dataset to 37,068 samples and fine-tuning hyperparameters, the subsequent classification report demonstrated improved precision, recall, and F1-Score for most classes, indicating the enhanced performance achieved on the larger dataset. The utilization of the pre-trained MobileNetv2 architecture proved instrumental in effectively capturing data patterns and relationships, contributing to the overall improvement in model accuracy and performance.

Several noteworthy aspects of our model emerged. Firstly, it surpassed rule-based and fuzzy models with an accuracy of 95.6623% on the evaluation dataset. This elevated accuracy underscored our model's superior predictive and classifying capabilities for target categories. Moreover, our model excelled in precision and recall across the majority of classes, with improved classification performance attributed to balanced precision and recall. With 37,589 samples, our model demonstrated enhanced performance, highlighting its scalability and suitability for real-world scenarios. The model showcased good generalization, improving accuracy on the larger dataset and demonstrating proficiency in making accurate predictions on new, unseen data—an essential feature for practical applications. The model's superior performance was attributed to its

adept utilization of machine learning techniques, particularly its flexible approach to capturing intricate patterns and relationships within the data. This adaptability enabled our model to learn from diverse datasets and accommodate different sign language gestures, potentially enhancing performance compared to rule-based or less flexible models. Owing to these advantages, our SLR model emerged as more reliable and effective than alternative models.

#### 4.3 Limitation

During the evaluation, the Greek Sign Language Recognition (GSLR) system encountered challenges in distinguishing similar hand gestures. Specifically, the model faced difficulty in discerning the Greek letters Alpha, Upsilon, Omega, and Beta. The similarities between the hand configurations of Alpha and Upsilon posed a particularly challenging distinction. Consequently, the model tended to misidentify the gesture for Alpha as Upsilon, introducing potential misinterpretations in sign language communication.

To mitigate this issue, enhancements in model training and dataset augmentation were explored as potential solutions. By refining the discrimination capabilities between these Greek letters, misclassifications and ambiguities could be effectively eliminated. Additionally, the introduction of new features or richer contextual data was considered to aid the system in distinguishing similar gestures with greater accuracy.

However, a notable drawback surfaced as the GSLR system necessitated extensive computations and intricate mathematical calculations. The system's operational efficiency was dependent on the hardware specifications, specifically requiring a 2.0 GHz multi-core processor and 4GB of RAM. Failure to meet these specifications not only resulted in sluggish system performance but also led to rapid battery drainage. These challenges highlighted the imperative of continual improvement in model training and dataset enrichment for enhanced gesture discrimination, along with the consideration of hardware specifications to optimize the GSLR system's overall performance. A complete list of abbreviations is listed in *Appendix I*.

#### 5. Conclusion and future work

This study demonstrated that the image-based approach is more effective and practical for hand gesture recognition compared to the sensor-based

approach. The image-based method's advantages include cost-effectiveness, elimination of additional equipment, and its seamless integration with ubiquitous smartphone cameras, making it highly accessible and practical for daily use. Conversely, the sensor-based method faced challenges like high hardware costs, fragility, and maintenance requirements. By leveraging machine learning over rule-based methods, the study capitalized on the ability of machine learning to handle complex and nonlinear relationships, adapt to new data, and improve accuracy through experience. The research employed a quantitative approach, collecting numerical data to analyze the efficacy and efficiency of the machine learning model for GSL recognition. The CNN algorithm trained on video data of 24 Greek alphabets demonstrated promising results. Through rigorous testing and validation, the `cvzone.HandTrackingModule` was identified as the optimal tool for accurate hand gesture detection, reinforcing the study's conclusions on the superiority of image-based and machine learning techniques for GSL recognition.

### 5.1 Future work

While our algorithm demonstrated proficiency in identifying individual letters, there existed untapped potential for further development and increased capacity. Subsequent efforts were directed toward enhancing the system's precision and expanding its vocabulary beyond Greek letters to encompass words and phrases. One avenue for improving accuracy involved the collection of a more extensive and diverse dataset or refining the machine learning models algorithm. To accommodate phrases and words within the system, a focused endeavor was made to train it to recognize and interpret complete, commonly used phrases and words. This approach aimed to render communication more meaningful and fluent, elevating the overall utility of the GSLR system. The next focus of our work involved optimizing real-time performance to make the GSLR system more practical and user-friendly. This entailed implementing measures to reduce processing lags, ensuring nearly instantaneous gesture recognition and translation. Inclusivity was prioritized by involving individuals with hearing impairments and sign language experts in the future improvement of our model, aligning it more closely with the needs of the target user community.

Iterative development strategies were pursued, emphasizing the incorporation of user feedback and insights to tailor the system to users' specific needs

and preferences. This user-centric approach aimed to enhance the GSLR system's efficiency and relevance, contributing to its continuous improvement and alignment with user requirements.

Future efforts should aim to expand the vocabulary to include common words and phrases and optimize the system's real-time performance to minimize processing delays

### Acknowledgment

None.

### Conflicts of interest

The authors have no conflicts of interest to declare.

### References

- [1] Obi Y, Claudio KS, Budiman VM, Achmad S, Kurniawan A. Sign language recognition system for communicating to people with disabilities. *Procedia Computer Science*. 2023; 216:13-20.
- [2] Riad AM, Elminir HK, Shohieb SM. Hand gesture recognition system based on a geometric model and rule based classifier. *British Journal of Applied Science & Technology*. 2014; 4(9):1432-44.
- [3] Mariappan HM, Gomathi V. Real-time recognition of Indian sign language. In international conference on computational intelligence in data science (ICCIDS) 2019 (pp. 1-6). IEEE.
- [4] Wu J, Sun L, Jafari R. A wearable system for recognizing American sign language in real-time using IMU and surface EMG sensors. *IEEE Journal of Biomedical and Health Informatics*. 2016; 20(5):1281-90.
- [5] Rekha J, Bhattacharya J, Majumder S. Hand gesture recognition for sign language: a new hybrid approach. In proceedings of the international conference on image processing, computer vision, and pattern recognition (IPCV) 2011 (pp. 1-7). WorldComp.
- [6] Huu PN, Phung NT. Hand gesture recognition algorithm using SVM and HOG model for control of robotic system. *Journal of Robotics*. 2021; 2021:1-3.
- [7] Shinde P, Shinde P, Shinde S, Shinde S, Shinde S. Augmented reptile feeder. In Pune section international conference (PuneCon) 2022 (pp. 1-4). IEEE.
- [8] Ismail MH, Dawwd SA, Ali FH. A review on Arabic sign language recognition. *Journal of Advances in Computer and Electronics Engineering*. 2021; 6(12):1-12.
- [9] Madhubala B, Sarathambekai S, Vairam T, Seelan KS, Sathya RS, Swathy AR. Pre-trained convolutional neural network model based pneumonia classification from chest X-Ray images. In proceedings of the international conference on smart data intelligence (ICSMDI 2021) 2021(pp.1-8).
- [10] Michele A, Colin V, Santika DD. Mobilenet convolutional neural networks and support vector

- machines for palmprint recognition. *Procedia Computer Science*. 2019; 157:110-7.
- [11] Carney M, Webster B, Alvarado I, Phillips K, Howell N, Griffith J, et al. Teachable machine: approachable web-based tool for exploring machine learning classification. In *extended abstracts of the 2020 CHI conference on human factors in computing systems 2020* (pp. 1-8). ACM.
- [12] Dogo EM, Afolabi OJ, Nwulu NI, Twala B, Aigbavboa CO. A comparative analysis of gradient descent-based optimization algorithms on convolutional neural networks. In *international conference on computational techniques, electronics and mechanical systems (CTEMS) 2018* (pp. 92-9). IEEE.
- [13] Patro VM, Patra MR. Augmenting weighted average with confusion matrix to enhance classification accuracy. *Transactions on Machine Learning and Artificial Intelligence*. 2014; 2(4):77-91.
- [14] Agarwal A, Sharma P, Alshehri M, Mohamed AA, Alfarraj O. Classification model for accuracy and intrusion detection using machine learning approach. *Peer J Computer Science*. 2021; 7:e437.
- [15] Wadhawan A, Kumar P. Deep learning-based sign language recognition system for static signs. *Neural Computing and Applications*. 2020; 32:7957-68.
- [16] Wen F, Zhang Z, He T, Lee C. AI enabled sign language recognition and VR space bidirectional communication using triboelectric smart glove. *Nature Communications*. 2021; 12(1):1-13.
- [17] Rastgoo R, Kiani K, Escalera S. Sign language recognition: a deep survey. *Expert Systems with Applications*. 2021; 164:113794.
- [18] Rastgoo R, Kiani K, Escalera S. Hand sign language recognition using multi-view hand skeleton. *Expert Systems with Applications*. 2020; 150:113336.
- [19] Wadhawan A, Kumar P. Sign language recognition systems: a decade systematic literature review. *Archives of Computational Methods in Engineering*. 2021; 28:785-813.
- [20] Al-hammadi M, Muhammad G, Abdul W, Alsulaiman M, Bencherif MA, Alrayes TS, et al. Deep learning-based approach for sign language gesture recognition with efficient hand gesture representation. *IEEE Access*. 2020; 8:192527-42.
- [21] Corballis MC. From mouth to hand: gesture, speech, and the evolution of right-handedness. *Behavioral and Brain Sciences*. 2003; 26(2):199-208.
- [22] Oyedotun OK, Khashman A. Deep learning in vision-based static hand gesture recognition. *Neural Computing and Applications*. 2017; 28(12):3941-51.
- [23] Nagi J, Ducatelle F, Di CGA, Cireşan D, Meier U, Giusti A, et al. Max-pooling convolutional neural networks for vision-based hand gesture recognition. In *international conference on signal and image processing applications (ICSIPA) 2011* (pp. 342-7). IEEE.
- [24] Rioux-maldague L, Giguere P. Sign language fingerspelling classification from depth and color images using a deep belief network. In *Canadian conference on computer and robot vision 2014* (pp. 92-7). IEEE.
- [25] Huang J, Zhou W, Li H, Li W. Sign language recognition using 3d convolutional neural networks. In *international conference on multimedia and expo (ICME) 2015* (pp. 1-6). IEEE.
- [26] Huang J, Zhou W, Li H, Li W. Sign language recognition using real-sense. In *China summit and international conference on signal and information processing (ChinaSIP) 2015* (pp. 166-70). IEEE.
- [27] Pigou L, Dieleman S, Kindermans PJ, Schrauwen B. Sign language recognition using convolutional neural networks. In *computer vision-ECCV workshops: Zurich, Switzerland, Proceedings, Part I 13 2015* (pp. 572-8). Springer International Publishing.
- [28] Molchanov P, Gupta S, Kim K, Pulli K. Multi-sensor system for driver's hand-gesture recognition. In *11th IEEE international conference and workshops on automatic face and gesture recognition (FG) 2015* (pp. 1-8). IEEE.
- [29] Tang A, Lu K, Wang Y, Huang J, Li H. A real-time hand posture recognition system using deep neural networks. *ACM Transactions on Intelligent Systems and Technology (TIST)*. 2015; 6(2):1-23.
- [30] Yang S, Zhu Q. Video-based Chinese sign language recognition using convolutional neural network. In *9th international conference on communication software and networks (ICCSN) 2017* (pp. 929-34). IEEE.
- [31] Tushar AK, Ashiquzzaman A, Islam MR. Faster convergence and reduction of overfitting in numerical hand sign recognition using DCNN. In *region 10 humanitarian technology conference (R10-HTC) 2017* (pp. 638-41). IEEE.
- [32] Bheda V, Radpour D. Using deep convolutional networks for gesture recognition in american sign language. *arXiv preprint arXiv:1710.06836*. 2017.
- [33] Rao GA, Syamala K, Kishore PV, Sastry AS. Deep convolutional neural networks for sign language recognition. In *conference on signal processing and communication engineering systems (SPACES) 2018* (pp. 194-7). IEEE.
- [34] Koller O, Zargaran S, Ney H, Bowden R. Deep sign: enabling robust statistical continuous sign language recognition via hybrid CNN-HMMs. *International Journal of Computer Vision*. 2018; 126:1311-25.
- [35] Kumar EK, Kishore PV, Kumar MT, Kumar DA. 3D sign language recognition with joint distance and angular coded color topographical descriptor on a 2-stream CNN. *Neurocomputing*. 2020; 372:40-54.
- [36] Prabhu R. Understanding of convolutional neural network (CNN)-deep learning. *Medium Com*. 2018:1-11.
- [37] Rahaman MA, Jasim M, Ali MH, Hasanuzzaman M. Real-time computer vision-based Bengali sign language recognition. In *17th international conference on computer and information technology (ICCIT) 2014* (pp. 192-7). IEEE.
- [38] Uddin MA, Chowdhury SA. Hand sign language recognition for Bangla alphabet using support vector machine. In *international conference on innovations in*

science, engineering and technology (ICISSET) 2016 (pp. 1-4). IEEE.

- [39] Rao GA, Kishore PV. Selfie video based continuous Indian sign language recognition system. *Ain Shams Engineering Journal*. 2018; 9(4):1929-39.
- [40] Zhou H, Zhou W, Zhou Y, Li H. Spatial-temporal multi-cue network for continuous sign language recognition. In *proceedings of the AAAI conference on artificial intelligence 2020* (pp. 13009-16).
- [41] Sarhan N, Wilms C, Closius V, Brefeld U, Frintrop S. Hands in focus: sign language recognition via top-down attention. In *international conference on image processing (ICIP) 2023* (pp. 2555-9). IEEE.
- [42] Marais M, Brown D, Connan J, Boby A. Spatiotemporal convolutions and video vision transformers for signer-independent sign language recognition. In *international conference on artificial intelligence, big data, computing and data communication systems (icABCD) 2023* (pp. 1-6). IEEE.
- [43] Tarrés L, Gállego GI, Duarte A, Torres J, Giró-i-nieto X. Sign language translation from instructional videos. In *proceedings of the conference on computer vision and pattern recognition 2023* (pp. 5624-34). IEEE.
- [44] Laines D, Gonzalez-mendoza M, Ochoa-ruiz G, Bejarano G. Isolated sign language recognition based on tree structure skeleton images. In *proceedings of the conference on computer vision and pattern recognition 2023* (pp. 276-84). IEEE.
- [45] Arkushin RS, Moryossef A, Fried O. Ham2pose: Animating sign language notation into pose sequences. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition 2023* (pp. 21046-56).
- [46] Papadimitriou K, Sapountzaki G, Vasilaki K, Efthimiou E, Fotinea SE, Potamianos G. SL-REDU GSL: a large greek sign language recognition corpus. In *international conference on acoustics, speech, and signal processing workshops (ICASSPW) 2023* (pp. 1-5). IEEE.
- [47] Papadimitriou K, Potamianos G. Sign language recognition via deformable 3d convolutions and modulated graph convolutional networks. In *international conference on acoustics, speech and signal processing (ICASSP) 2023* (pp. 1-5). IEEE.
- [48] Naz N, Sajid H, Ali S, Hasan O, Ehsan MK. Signgraph: an efficient and accurate pose-based graph convolution approach toward sign language recognition. *IEEE Access*. 2023; 11:19135-47.
- [49] Al-qaisy NE, Al-kaseem BR, Al-dunainawi Y. AI-based portable gesture recognition system for hearing impaired people using wearable sensors. In *15th international conference on developments in eSystems engineering (DeSE) 2023* (pp. 33-8). IEEE.
- [50] Mahyoub M, Natalia F, Sudirman S, Mustafina J. Sign language recognition using deep learning. In *15th international conference on developments in eSystems engineering (DeSE) 2023* (pp. 184-9). IEEE.



**Neetu Pillai** is a lecturer at the University of West London, Ras Al Khaimah, UAE. She received her M.E. degree in Computer Engineering from Mumbai University in 2019 and her B.E. degree in Computer Engineering from Mumbai University in 2017. Her key areas of interest include Cyber Security, Digital Forensics, Programming, and Edge Computing.

Email: neetu.pillai012@gmail.com



**Mohamed Aqib Abid**, is a Bachelor of Computer Science student of University of West London, Ras AL Khaimah, UAE.

Email: mohamedaqibabid@gmail.com



**Nahsam Ahammed** is University of West London Graduate who is majored in Computer Science. He is working towards his master in cyber security and done relevant certifications and projects in the field of cyber security.

Email: Nahsamahammed106@gmail.com

#### Appendix I

S. No.	Abbreviation	Description
1	AI	Artificial Intelligence
2	ASL	American Sign Language
3	ASLR	Automated Sign Language Recognition
4	CNN	Convolutional Neural Network
5	FCM	Fuzzy c-Means
6	GSL	Greek Sign Language
7	GSLR	Greek Sign Language Recognition
8	HOG	Histogram of Oriented Gradients
9	IDE	Integrated Development Environment
10	kNN	k-Nearest Neighbors
11	PC	Personal Computer
12	RBF	Radial Basis Function
13	SLR	Sign Language Recognition
14	SVM	Support Vector Machine