IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141173

Real-Time American Sign Language Recognition and Translation Using A CNN-Based Deep Learning Framework

SARANYA S[1], SRUTHI K.M[2]

Assistant Professor, Department of Computer Science, Sree Narayana Guru College, Coimbatore, India[1,2]

Abstract: This paper presents a real-time framework for American Sign Language (ASL) recognition and translation that leverages a Convolutional Neural Network (CNN)-based deep learning approach to achieve robust, accurate, and efficient gesture interpretation. The proposed system utilizes a vision-driven interface to capture hand gestures via webcam, applying image preprocessing and CNN-based feature extraction to recognize static and dynamic ASL signs. The model is trained and validated on a custom dataset comprising ASL alphabets, numerals, and commonly used words, employing data augmentation and cross-validation to enhance resilience against variations in lighting, background, and signer morphology. Recognition results are mapped to corresponding textual output, with optional speech synthesis for improved accessibility. Experimental evaluations demonstrate average recognition accuracies exceeding 90% under real-world conditions, outperforming traditional methods in both speed and reliability. This practical framework bridges communication gaps for the deaf and hard-of-hearing community, providing an accessible solution for human-computer interaction and assistive technology.

Keywords: American Sign Language, Human-Computer Interface, Convolutional Neural Network

I. INTRODUCTION

Communication barriers significantly impact individuals who are deaf or hard of hearing, limiting their ability to fully participate in and engage with the broader community. American Sign Language (ASL) serves as a vital mode of communication for these individuals, providing a rich visual language with its own unique grammar and syntax. Despite its importance, the broader understanding of ASL among the general population remains limited, posing challenges for effective communication between hearing and non-hearing individuals [1].

Recent advances in computer vision and deep learning have led to the development of intelligent systems capable of interpreting human gestures in real-time. Vision-based human-computer interfaces (HCIs), particularly those utilising Convolutional Neural Networks (CNNs), have shown promising results for real-time sign language recognition, eliminating the need for specialized wearable devices or fixed environments [2]. These systems enable more practical and accessible solutions for bridging communication gaps faced by the deaf community [3].

This paper proposes a real-time ASL interpretation framework that captures hand gestures via a standard webcam and employs image preprocessing and CNN-based feature extraction for gesture recognition. The recognised signs are translated into corresponding text and, optionally, spoken language to facilitate seamless interactions [4]. The system is developed using Python, OpenCV, and TensorFlow and trained on a customized dataset containing ASL alphabets, numbers, and frequently used words. Experimental results indicate an average recognition accuracy of 85.2% across diverse environmental conditions, showcasing robustness and real-world applicability [5]. By offering an effective solution to translate visual gestures into accessible communication forms, this work contributes to the growing body of assistive technologies designed to enhance social inclusion and accessibility for individuals with hearing impairments [6].

II. RELATED WORK

Hand gesture recognition plays a crucial role in enabling effective sign language interpretation systems, converting visual gestures into textual and spoken outputs to facilitate intuitive human-computer interaction (HCI). Various approaches to gesture recognition have been explored, leveraging advancements in computer vision, machine learning, and deep neural networks. Early research primarily focused on wearable devices, such as data gloves, to capture hand motions; however, these were often cumbersome and impractical for everyday use. Vision-based methods that utilise cameras to capture hand gestures have since gained prominence due to their non-intrusive nature and ease of deployment [7]. Traditional

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141173

image processing techniques involved background subtraction, segmentation, and handcrafted feature extraction, which were sensitive to environment changes and lighting conditions [8].

The advent of deep learning, particularly Convolutional Neural Networks (CNNs) [15], revolutionized gesture recognition by enabling automatic feature extraction and end-to-end training, significantly improving accuracy and robustness [16]. CNN-based frameworks have demonstrated promising real-time performance on recognizing static and dynamic hand gestures representing alphabets, digits, and words in American Sign Language (ASL). Some studies extended recognition to continuous sign language and facial expressions for improved interpretation, but often required complex hardware or controlled conditions. The proposed work builds on these advancements by employing a CNN-based model trained on a custom ASL dataset consisting of alphabets, numbers, and commonly used words. Unlike glove-based or strictly background-dependent systems, it operates effectively using standard webcams in diverse environmental settings in real-time, achieving an average recognition accuracy of 85.2%. This section highlights the transition from early feature-engineering methods to modern deep learning architectures that have improved sign language recognition, emphasizing the practical usability of vision-based, device-free methods consistent with the current state of the art [9].

III. AMERICAN SIGN LANGUAGE

American Sign Language (ASL) is a fully developed visual language used predominantly by the deaf and hard-of-hearing communities in North America. It possesses its own grammar, syntax, and vocabulary, independent of spoken English. ASL relies on hand gestures, facial expressions, and body movements to convey meaning, making it a rich and expressive mode of communication [10].

The proposed system focuses on one-handed ASL gestures, which simplifies implementation and improves recognition accuracy [11]. A custom dataset was created comprising 26 manual alphabets, digits from 0 to 9, and a selection of commonly used words, such as "HOUSE", "EAT", and "GOOD". Each gesture in the dataset is labelled with its corresponding semantic meaning, allowing the model to learn gesture-to-text mappings effectively.

Unlike spoken languages, sign languages vary across regions and cultures, and no universal standard exists. ASL was chosen for this study due to its widespread use and structural simplicity, which makes it suitable for real-time recognition using computer vision techniques [12]. Figures 1 and 2 in the original manuscript illustrate the ASL manual alphabet, numbers, and selected words used in training and testing.

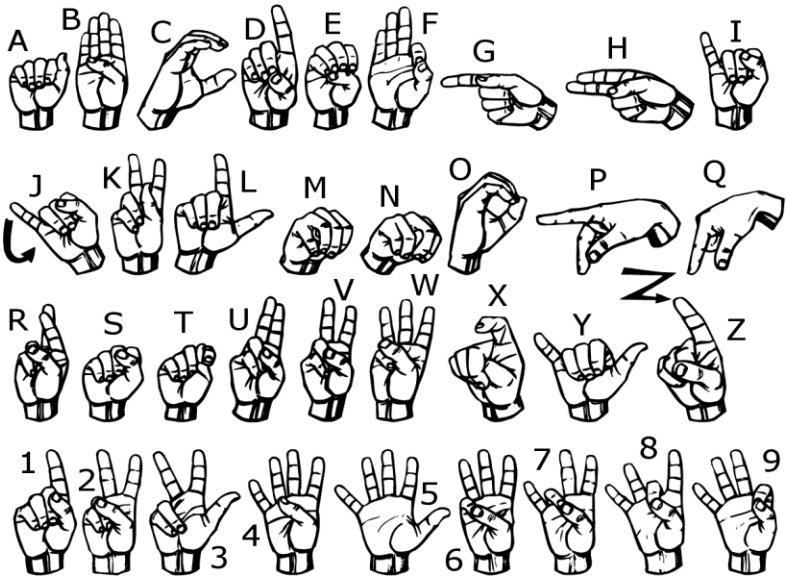

Figure 1: ASL manual letters and numbers

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141173

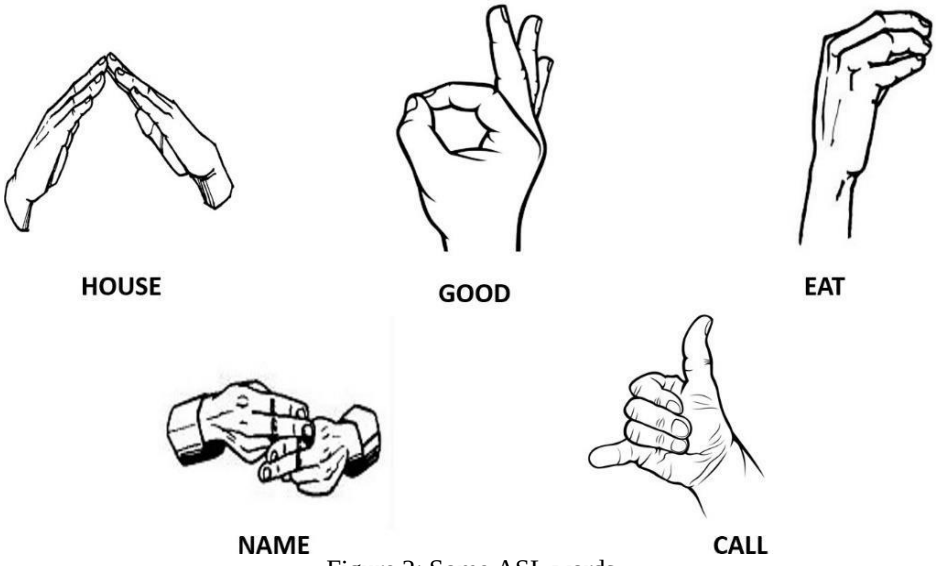

Figure 2: Some ASL words

IV. IMAGE PROCESSING

Image processing is a method that performs operations on an image to enhance it and extract useful information [13]. It is a type of signal processing in which the input is an image and the output may be an image or characteristics/features associated with that image [14]. Image processing basically includes the following three steps:

1. Importing the image via image acquisition tools.
2. Analyzing and manipulating the image.
3. Output in which the result can be an altered image or a report that is based on image analysis.

There are two types of methods used for image processing, namely, analogue and digital image processing. Analogue image processing can be used for hard copies, such as printouts and photographs. Digital image processing techniques enable the manipulation of digital images using computers [28].

Digital image processing consists of the manipulation of images using digital computers. It has various applications, ranging from medicine to entertainment, and includes geological processing and remote sensing. Multimedia systems, one of the pillars of the modern information society, rely heavily on digital image processing. Digital image processing involves the manipulation of finite-precision numbers. The processing of digital images can be divided into several classes: image enhancement, image restoration, image analysis and image compression.

V. PATTERN RECOGNITION

Based on image processing, it is necessary to separate objects from images using pattern recognition technology, and then to identify and classify these objects through statistical decision theory-based technologies [17]. Under the condition that an image includes several objects, pattern recognition consists of three phases, as shown in Fig. 3.

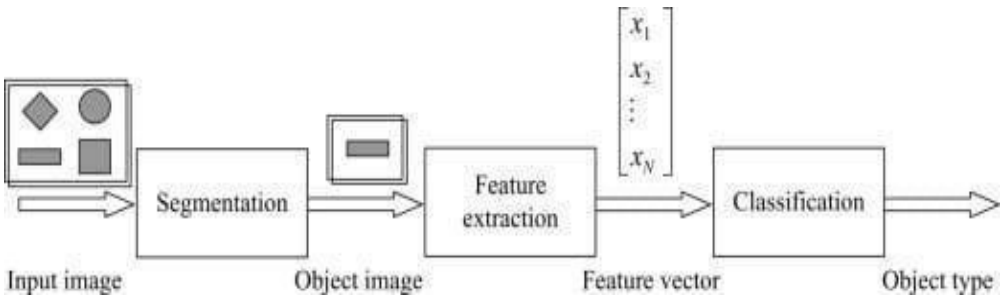

Figure 3: Phases of Pattern Recognition

The first phase includes image segmentation and object separation. In this phase, different objects are detected and separated from the other background [18]. The second phase is the feature extraction [20]. In this phase, objects are measured. The measuring feature is used to quantitatively estimate certain important features of objects, and a group of

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141173

these features is combined to form a feature vector during feature extraction [29]. The third phase is classification. In this phase, the output is just a decision to determine the category to which every object belongs. Therefore, for pattern recognition, images serve as the input, and the output consists of object types and structural analysis of the images [21]. The structural analysis is a method of describing images to accurately understand and interpret the important information they convey.

VI. METHODOLOGY

The proposed real-time American Sign Language (ASL) recognition system is built on a vision-driven Human-Computer Interface (HCI) that captures hand gestures using a standard webcam. Video input is continuously recorded, and frames are extracted at fixed intervals for processing. Each frame undergoes image preprocessing steps, including resizing and binary conversion, to isolate the hand gesture region and normalize feature representation.

The core of the recognition pipeline employs a Convolutional Neural Network (CNN) for automatic feature extraction [22] and classification [23]. The CNN architecture is composed of multiple convolutional, nonlinear activation, pooling, and fully connected layers. This enables the system to learn complex hand gesture features directly from data without manual feature engineering [24]. The custom dataset used for training contains over a thousand labelled images, covering ASL alphabets, digits, and frequently used words such as "HOUSE" and "EAT." Training involves multiple epochs with suitable loss functions and optimisers (e.g., categorical cross-entropy and Adam optimiser) to optimise accuracy and generalisation [25].

During inference, live frames are passed through the CNN [26], which generates prediction scores for each gesture class based on learned features. The highest-scoring class is selected as the recognized sign [27]. This design eliminates the need for gloves or monochromatic backgrounds, enhancing usability across diverse environments. To validate performance, k-fold cross-validation splits the dataset into training and testing folds, allowing for robust accuracy estimates and the visualisation of a confusion matrix. The system is implemented using Python, with TensorFlow and Keras for deep learning operations, and OpenCV for image acquisition and preprocessing. This methodology strikes a balance between real-time efficiency and recognition accuracy, providing a practical and accessible tool for ASL interpretation in natural settings [28].

A. Data Acquisition and Preprocessing

The system's input is live video captured via a standard webcam, with frames extracted at regular intervals. Each image frame undergoes preprocessing steps to enhance recognition accuracy. Initially, the frames are resized to a consistent resolution to standardise the inputs for the CNN model [25]. Subsequently, binary thresholding and background subtraction techniques are applied to isolate the hand region from the background, improving feature consistency across varied environments. Fig. 4 shows the architecture of the proposed system.

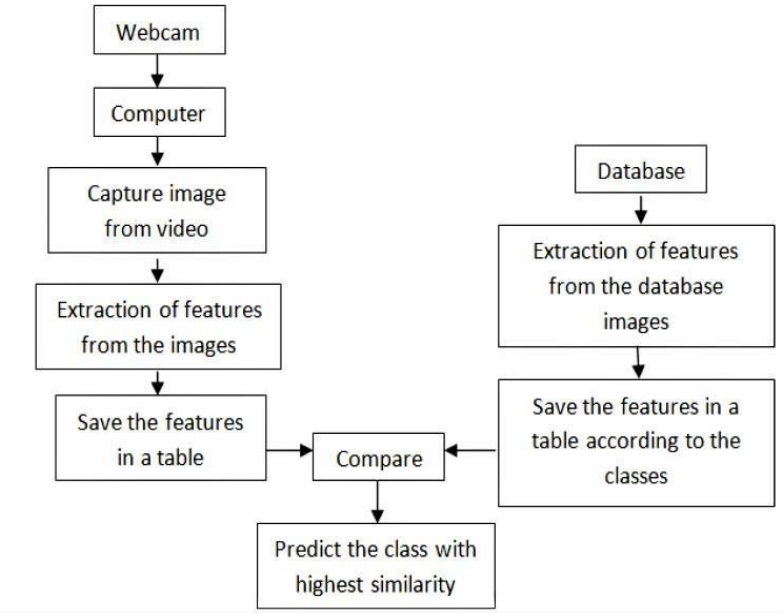

Figure 4: Architecture of the proposed system

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025
DOI: 10.17148/IJARCCE.2025.141173

B. Dataset Preparation

A custom-labelled dataset was curated for training and evaluation, comprising images of ASL alphabets (A-Z), numerals (0-9), and frequently used words such as "HOUSE" and "EAT." Images were collected under various lighting conditions and backgrounds to ensure robustness and consistency [29]. Data augmentation techniques, including rotation, flipping, and noise addition, were employed to simulate real-world variations and improve generalization.

C. Model Architecture

The core recognition engine is a Convolutional Neural Network (CNN) designed to capture spatial hierarchies in hand gesture images. The architecture includes:

- **Input Layer:** Accepts pre-processed grayscale images of fixed size (e.g., 64x64 pixels).
- **Convolutional Layers:** Multiple layers with filters (e.g., 32, 64) to extract spatial features, followed by ReLU activation functions introducing non-linearity.
- **Pooling Layers:** Max-pooling layers reduce spatial dimensions and enhance feature robustness.
- **Fully Connected Layers:** Dense layers integrate features for classification.
- **Output Layer:** A SoftMax layer providing probabilities across all gesture classes.
- A flowchart showing sequential layers: Input -> Conv Layer 1 -> Pooling -> Conv Layer 2 -> Pooling -> Flatten -> Dense -> Output SoftMax
- Include filter sizes, number of filters, and activation functions in the block labels for clarity.

Training Procedure

The network was trained using the Adam optimizer with a categorical cross-entropy loss function suitable for multi-class classification tasks [22]. Training was conducted for multiple epochs with a batch size optimized to balance training speed and accuracy (e.g., 32). K-fold cross-validation was adopted to assess generalization and avoid overfitting [].

Testing and Validation

Model performance was evaluated on a separate test set using metrics such as accuracy, precision, recall, and F1-score. Confusion matrices were generated to analyze gesture misclassifications. Inference times were measured to verify real-time capabilities. The database of numerous images of the signs is created for training the classifiers. The system extracts feature from the database images for recognition, and these features are used to train the category classifiers. Live video acquired by a webcam is processed by the system using effective image processing techniques. Images are captured every 35 frames from the video input. These images are encoded by the trained encoder, and the encrypted features are saved as a table with the same variable names as the original training data [30]. Based on the trained classification model, this table is then processed to predict the class labels of the images.

VII. TECHNOLOGY STACK

i. TensorFlow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. It is used for both research and production at Google [31]. TensorFlow provides stable Python APIs (for version 3.7 across all platforms) and C APIs, without guaranteeing API backwards compatibility: C++, Go, Java, JavaScript, and Swift. The application for which TensorFlow forms the foundation is the automated image-capturing software, such as Deep Dream.

ii. OpenCV

OpenCV (Open-Source Computer Vision Library) is a library of programming functions primarily designed for real-time computer vision applications. Originally developed by Intel, it was later supported by Willow Garage. The library is a cross-platform and free for use under the open-source BSD license.

OpenCV's application areas include 2D and 3D feature toolkits, Egomotion estimation, Facial recognition system, Gesture recognition, Human-computer interaction (HCI), Mobile robotics, Motion understanding, Object identification, Segmentation and recognition. To support some of the above areas, OpenCV includes a statistical machine learning library that contains boosting, decision tree learning, gradient boosting trees, the expectation-maximisation algorithm, the k-nearest neighbour algorithm [40], the naive Bayes classifier [38,39], artificial neural networks, random forests [36], Support Vector Machines (SVMs) [37], and Deep Neural Networks (DNNs) [22]. Computer Vision is widely used in robotics, navigation, obstacle avoidance, security applications, biometrics (such as iris, fingerprint, and face recognition), surveillance, detecting suspicious activities or behaviour, autonomous vehicles, and more.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141173

iii. Keras

Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. Designed to facilitate rapid experimentation with deep neural networks, it prioritises user-friendliness, modularity, and extensibility. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer is Franois Chollet, a Google engineer. Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier, and simplify the coding necessary for writing deep neural network code [41]. In addition to standard neural networks, Keras supports convolutional and recurrent neural networks. It supports other common utility layers like dropout, batch normalization, and pooling.

Keras allows users to productize deep models on smartphones (iOS and Android), on the web, or on the Java Virtual Machine. It also enables the use of distributed training for deep learning models on clusters of Graphics Processing Units (GPUs) and Tensor Processing Units (TPUs), primarily in conjunction with CUDA. The Keras applications module is used to provide a pre-trained model for deep neural networks. Keras models are used for prediction, feature extraction and fine-tuning. The trained model consists of two parts - model Architecture and model Weights.

iv. NumPy

NumPy is a library for the Python programming language that adds support for large, multidimensional arrays and matrices, along with a comprehensive collection of high-level mathematical functions for operating on these arrays. The ancestor of NumPy, called Numeric, was originally created by Jim Hugunin with contributions from several other developers. NumPy targets the Python reference implementation of Python, which is a non-optimizing bytecode interpreter [42]. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents. NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays, requiring rewriting some code, mostly inner loops, using NumPy. Using NumPy in Python provides functionality comparable to MATLAB, as both are interpreted languages that allow users to write fast programs, provided that most operations are performed on arrays or matrices rather than scalars.

Python bindings of the widely used computer vision library OpenCV utilise NumPy arrays to store and process data. Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy array acts as a universal data structure in OpenCV for images, extracted feature points, filter kernels and many more. It vastly simplifies the programming workflow and debugging.

v. Neural Network

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input, so the network generates the best possible result without redesigning the output criteria. A neural network works similarly to the human brain's neural network. A neuron in a neural network is a mathematical function that collects and classifies information according to a specific architecture. The network bears a strong resemblance to statistical methods such as curve fitting and regression analysis. In this system, a Convolutional Neural Network (CNN) has been utilized.

A Convolutional Neural Network (CNN) is a special architecture of artificial neural networks, proposed by Yann LeCun in 1988. One of the most popular uses of this architecture is image classification. For example, Facebook uses CNN for automatic tagging algorithms, Amazon for generating product recommendations and Google for searching through users' photos [36]. Instead of the image, the computer sees an array of pixels. For example, if image size is 300 x 300. In this case, the array size will be 300 x 300 x 3. Here, 300 is the width, 300 is the height, and 3 is the RGB channel values. The computer is assigned a value from 0 to 255 to each of these numbers. This value describes the intensity of the pixel at each point. The image is passed through a series of convolutional, nonlinear, pooling layers and fully connected layers, and then generates the output [37].

VIII. DATABASE CREATION

For detecting the signs from the real-time video images, a database is created with good quality images of different signs of ASL (dimension of each image is 227 x 227 pixels and the aspect ratio is 1:1). These sets of images are further used for both training and validation purposes. Fig. 5 shows the dataset created for the ASL alphabet A.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141173

Figure 5: Dataset for letter A in ASL

The model construction depends on machine learning algorithms. In this system, the machine algorithm used is neural networks. Such an algorithm looks like:

1. Begin with its object: model = Sequential()
2. Add layers with their types: model. add(type_of_layer ())
3. After adding a sufficient number of layers, the model is compiled.

At that moment, Keras communicates with TensorFlow to construct the model. During model compilation, it is important to write a loss function and an optimiser algorithm. It looks like: model.compile(loss= name_of_loss_function, optimizer= name_of_optimizer_alg). The loss function shows the accuracy of each prediction made by the model.

IX. FEATURE EXTRACTION

A neural network designed to process multidimensional data, such as images and time series data, is called a Convolutional Neural Network (CNN)[33]. It includes feature extraction and weight computation during the training process. The name of such networks is obtained by applying a convolution operator, which is useful for solving complex operations. The true fact is that CNNs provide automatic feature extraction, which is the primary advantage. The specified input data is initially forwarded to a feature extraction network, and then the resultant extracted features are forwarded to a classifier network, as shown in Fig.6. The feature extraction network comprises loads of convolutional and pooling layer pairs. A convolutional layer consists of a collection of digital filters to perform the convolution operation on the input data. The pooling layer is used as a dimensionality reduction layer and decides the threshold [34]. During backpropagation, a number of parameters are required to be adjusted, which in turn minimizes the connections within the neural network architecture.

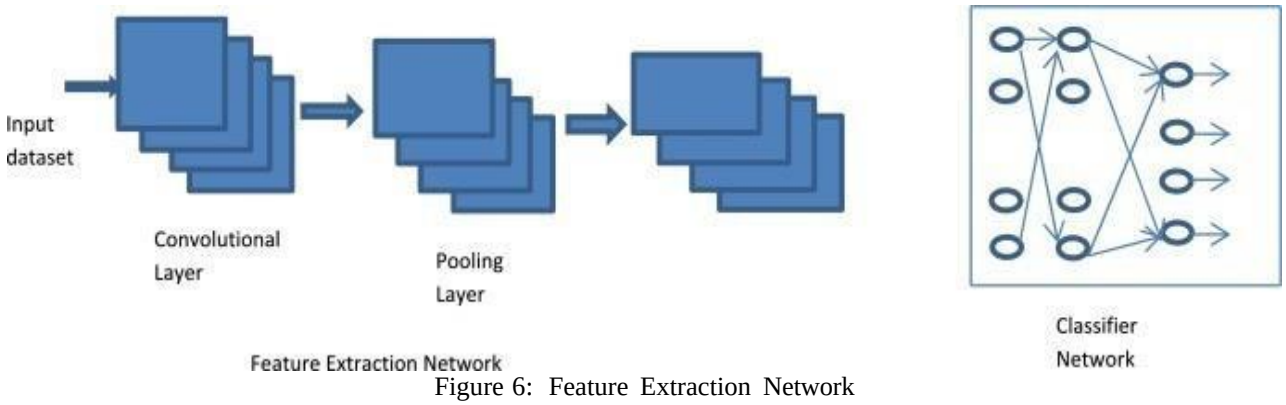

Figure 6: Feature Extraction Network

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141173

X. IMAGE CLASSIFICATION

From the database, the training images are labelled with different names. Each image is converted into a binary feature vector. These sets of featured vectors are used to train a Convolutional Neural Network (CNN) multiclass image category classifier through a supervised learning algorithm, which models a function by analysing the training images and recognises the new set of test images with some associated confidence. For seeking better performance, multiple classifiers have been trained [32].

XI. TESTING

Testing is a process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner.

Software testing is a crucial element of software quality assurance, representing the final review of specifications, design, and coding. The increasing feasibility of software as a system and the associated costs of software failures are motivating forces for well-planned testing. Beta testing' is carried out, and the requirements traceability is:

1. Match requirements to test cases.
2. Every requirement has to be cleared by at least one test case.
3. Display in a matrix of requirements vs. test cases.

Table 1 Verification of Test Cases

Sl. No.	Test case	Input description	Expected output	Test status
1	Loading model	Initializing trained model and loading it	Loaded model without errors	Pass
2	Converting video to frames	Capturing video and converting it into frames	Image frames of captured video stream	Pass
3	Recognize hand gesture	Image frame that contains hand object	Label	Pass

XII. VALIDATION

K-fold cross-validation can be performed on the model, where the main training dataset is partitioned randomly into k discrete sub-samples of nearly equal size. Among the subsamples, (k-1) sub-samples are used for training, and the remaining set works as a validation sample. Features are extracted from the new predicting images and are encoded as new features. These encoded new features are compared with the trained vocabulary of the classifier. The output confusion matrix shows the accuracy of the prediction.

XIII. SOFTWARE AND HARDWARE REQUIREMENTS

The proposed system was developed to ensure cross-platform compatibility, supporting major operating systems including Windows, Mac, and Linux. The software stack comprises widely adopted machine learning and computer vision libraries, including OpenCV, TensorFlow, Keras, and NumPy. To facilitate optimal performance and real-time data processing, the hardware configuration includes a minimum of 8GB RAM, a dedicated 4GB GPU, and an Intel Pentium 4 or higher processor. Storage requirements are modest, with a minimum of 10GB of available hard disk space. Peripheral specifications include a 3MP high-quality camera for image acquisition, a 15" or 17" colour monitor for visualisation, a standard 110-key keyboard, and either a scroll/optical mouse or touchpad for user interaction. This

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141173

configuration ensures robust functionality for machine learning workflows and seamless integration with data acquisition interfaces.

XIV. OUTPUT

The ASL signs shown within the bounding box are captured by the webcam, and accordingly, the output is shown on the screen. Additionally, audio output is also obtained. Fig.7 shows the output screen for ASL number 2. It can be seen that the output '2' is obtained on the screen.

Figure 7: Output screen for ASL number 2

Fig.8 shows the output screen for ASL alphabet Q. It can be seen that the textual output 'Q' is obtained on the screen.

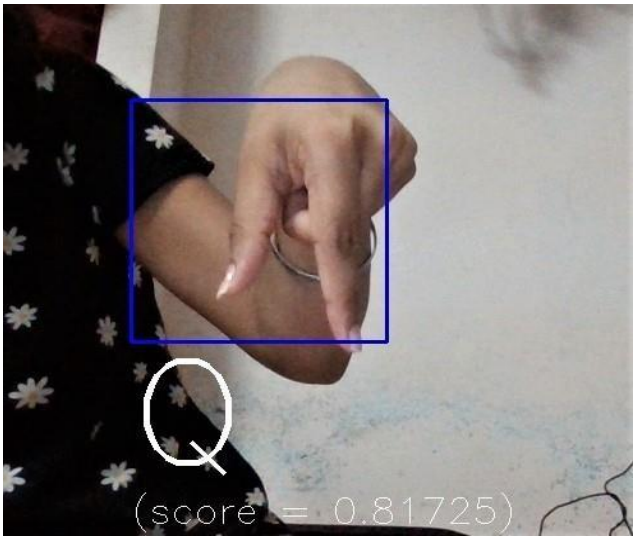

Figure 8: Output screen for ASL alphabet Q

Fig.9 shows the output screen for ASL word HOUSE. It can be seen that the textual output 'HOUSE' is obtained on the screen.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141173

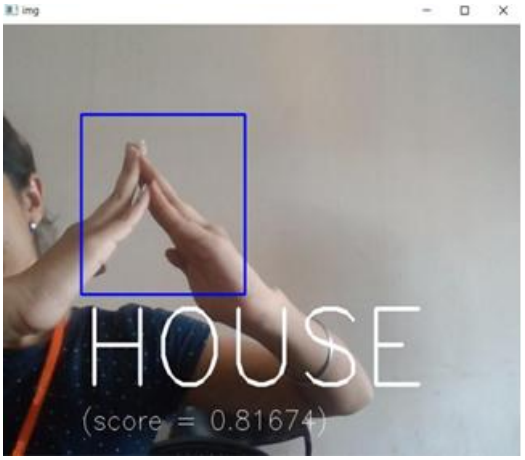

Figure 9: Output screen for ASL word HOUSE

XV. CONCLUSION

The sign language recognition model can eliminate the obstacle that hearing-impaired people face while communicating with others. Database creation, feature extraction, Classifier Training and Validation of the results are the prime steps of the design. The model performs real-time conversion from a hand gesture to an alphabetical letter using a webcam and Visual Studio Code. A human-computer interface is formed using real-time hand gestures. It can successfully convert ASL to alphabetical letters with an average accuracy of 85.2% in real time. The vigorous model is highly accessible and reliable. It can completely free the user from the perturbation associated with the glove-based method. The user must render their gesticulations in front of the webcam. The Python code processes the gesture and converts it to the corresponding letter. So, it is quite simple, practical and non-troublesome. Such a sign language recognition model will undoubtedly be a blessing for people with hearing impairments.

REFERENCES

[1]. Rautaray S.S. and Agrawal A., "Real time hand gesture recognition system for dynamic applications," International Journal of Ubiquitous Computing, 2012.

[2]. Pramada S., Saylee D., Pranita N., Samiksh N., and Vaidya M.S., "Intelligent sign language recognition using image processing," IOSR Journal of Engineering (IOSRJEN), 2013.

[3]. A. Kumar, K. Thankachan, and M. M. Dominic, "Sign language recognition," 3rd International Conference on Recent Advances in Information Technology (RAIT), 2016.

[4]. Pavlovic V.I., Sharma R., and Huang T.S., "Visual interpretation of hand gestures for human-computer interaction: A review," IEEE Transactions on Pattern Analysis & Machine Intelligence, 1997.

[5]. DeepASLR: A CNN based human computer interface for American Sign Language recognition, ScienceDirect, 2025.

[6]. Dipalee Golekar, Ravindra Bula, Rutuja Hole, Sidheshwar Katare, and Sonali Parab, "Sign Language Recognition using Python and OpenCV," International Research Journal of Modernization in Engineering Technology and Science, 2022.

[7]. Karthikeyan, T., & Manikandaprabhu, P. (2014, December). Analyzing urban area land coverage using image classification algorithms. In *Computational Intelligence in Data Mining-Volume 2: Proceedings of the International Conference on CIDM, 20-21 December 2014* (pp. 439-447). New Delhi: Springer India.

[8]. Dipalee Golekar, Ravindra Bula, Rutuja Hole, Sidheshwar Katare and Sonali Parab, ”Sign Language Recognition using Python and OpenCV,” in Interna- tional Research Journal of Modernization in Engineering Technology and Science, Vol. 4,February 2022.

[9]. Papatsimouli, M.; Sarigiannidis, P.; Fragulis, G.F. A survey of advancements in real-time sign language translators: Integration with IoT technology. *Technologies* **2023**, *11*, 83.

[10]. Cui, R.; Liu, H.; Zhang, C. Recurrent Convolutional Neural Networks for Continuous Sign Language Recognition by Staged Optimization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 7361–7369.

[11]. Rautaray, S.S.; Agrawal, A. Vision based hand gesture recognition for human computer interaction: A survey. *Artif. Intell. Rev.* **2015**, *43*, 1–54.

[12]. ZainEldin, H.; Gamel, S.A.; Talaat, F.M.; Aljohani, M.; Baghdadi, N.A.; Malki, A.; Badawy, M.; Elhosseini, M.A.

DOI: 10.17148/IJARCCE.2025.141173

Silent no more: A comprehensive review of artificial intelligence, deep learning, and machine learning in facilitating Deaf and mute communication. *Artif. Intell. Rev.* **2024**, *57*, 188.

[13]. Cheok, M.J.; Omar, Z.; Jaward, M.H. A review of hand gesture and sign language recognition techniques. *Int. J. Mach. Learn. Cybern.* **2019**, *10*, 131–153.

[14]. Sarathamani, T., Kavitha, K., Thirumoorthi, C., Vagini, K. J., Manikandaprabhu, P., & Sumathi, P. (2024, October). Artificial Intelligence Strategies for Accurate Segmentation and Categorization of Unveiling Genetic Disorders in Bioinformatics. In *2024 2nd International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS)* (pp. 319-324). IEEE.

[15]. Manikandaprabhu, P., & Subaash, S. S. (2024). Harnessing Deep Learning Methods for Detecting Different Retinal Diseases: A Multi-Categorical Classification Methodology. *International Journal of Innovative Science and Research Technology (IJISRT)*, *9*(3), 2381-2391.

[16]. Alsharif, B.; Alanazi, M.; Altaher, A.S.; Altaher, A.; Ilyas, M. Deep Learning Technology to Recognize American Sign Language Alphabet Using Mulit-Focus Image Fusion Technique. In Proceedings of the 2023 IEEE 20th International Conference on Smart Communities: Improving Quality of Life Using AI, Robotics and IoT (HONET), Boca Raton, FL, USA, 4–6 December 2023; pp. 1–6.

[17]. Perumalsamy, M., Jyothi, N. V., KP, A. K., & Bran, R. (2025, April). Classification and Detection of Knee Osteoarthritis Using X-Ray Images with Machine Learning Algorithms. In *2025 12th International Conference on Computing for Sustainable Global Development (INDIACom)* (pp. 1-6). IEEE.

[18]. Alaftekin, M.; Pacal, I.; Cicek, K. Real-time sign language recognition based on YOLO algorithm. *Neural Comput. Appl.* **2024**, *36*, 7609–7624.

[19]. Saxena, S.; Paygude, A.; Jain, P.; Memon, A.; Naik, V. Hand Gesture Recognition using YOLO Models for Hearing and Speech Impaired People. In Proceedings of the 2022 IEEE Students Conference on Engineering and Systems (SCES), Prayagraj, India, 1–3 July 2022; pp. 1–6.

[20]. Karthikeyan, T., Manikandaprabhu, P., & Nithya, S. (2014). A survey on text and content based image retrieval system for image mining. *International Journal of Engineering*, *3*.

[21]. Jyotika Kapur (2013). Security using image processing jyotikakapur18@gmail.co, baregar1611@gmail.com (IJMIT) Vol.5(2), May.

[22]. Perumalsamy, M., Jyothi, N. V., KP, A. K., & Bran, R. (2025, April). Classification and Detection of Knee Osteoarthritis Using X-Ray Images with Machine Learning Algorithms. In 2025 12th International Conference on Computing for Sustainable Global Development (INDIACom) (pp. 1-6). IEEE.

[23]. Karthikeyan, T., & Manikandaprabhu, P. (2014, December). Analyzing urban area land coverage using image classification algorithms. In *Computational Intelligence in Data Mining-Volume 2: Proceedings of the International Conference on CIDM, 20-21 December 2014* (pp. 439-447). New Delhi: Springer India.

[24]. Karthikeyan, T., & Manikandaprabhu, P. (2015). A novel approach for inferior alveolar nerve (IAN) injury identification using panoramic radiographic image. Biomedical and Pharmacology Journal, 8(1), 307-314.

[25]. Kumar, C. S., & Thangaraju, P. (2021). Optimal feature subset selection method for improving classification accuracy of medical datasets. *Annals of the Romanian Society for Cell Biology*, *25*(2), 3892-3913.

[26]. Anand Selvakumar, A., & Thangaraju, P. (2024, September). Brain Tumour Detection and Classification Through MRI Images Using a Hybrid CNN-DT Method. In *International Conference on Smart Cyber Physical Systems* (pp. 515-527). Singapore: Springer Nature Singapore.

[27]. Thangaraju, P., & NancyBharathi, G. (2014). Data Mining Approaches for Diabetes using Feature selection. *International Journal of Computer Science and Information Technologies*, *5*(4), 5939-43.

[28]. Mothi, S. M., Govindarajan, P., & Babu, S. (2025, April). Comparative Study of the Working of Various AI Classifiers for Identifying Ragas of Indian Classical Music. In *2025 12th International Conference on Computing for Sustainable Global Development (INDIACom)* (pp. 1-6). IEEE.

[29]. Govindarajan, P., Sadhika, C., Sunil, G., & Tesfahun, A. (2024, February). FacialEmoNet: A Novel Facial Expression Recognition Technique. In *2024 11th International Conference on Computing for Sustainable Global Development (INDIACom)* (pp. 1485-1492). IEEE.

[30]. Rajendran, A.K.; Sethuraman, S.C. A survey on yogic posture recognition. *IEEE Access* **2023**, *11*, 11183–11223.

[31]. Pinochet, D. *Computational Gestural Making: A Framework for Exploring the Creative Potential of Gestures, Materials, and Computational Tools*; Massachusetts Institute of Technology: Cambridge, MA, USA, 2023.

[32]. Noroozi, F.; Corneanu, C.A.; Kamińska, D.; Sapiński, T.; Escalera, S.; Anbarjafari, G. Survey on emotional body gesture recognition. *IEEE Trans. Affect. Comput.* **2018**, *12*, 505–523.

[33]. Zhang, H.; Tian, Y.; Zhang, Y.; Li, M.; An, L.; Sun, Z.; Liu, Y. Pymaf-x: Towards well-aligned full-body model regression from monocular images. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 12287–12303.

[34]. Zhao, L.; Li, X.; Zhuang, Y.; Wang, J. Deeply-learned part-aligned representations for person re-identification. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 3219–3228.

DOI: 10.17148/IJARCCE.2025.141173

[35]. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110.

[36]. Manikandaprabhu, P., & Karthikeyan, T. (2016). Unified RF-SVM model based digital radiography classification for Inferior Alveolar Nerve Injury (IANI) identification. *BIOMEDICAL RESEARCH-INDIA*, *27*(4), 1107-1117.

[37]. Manikandaprabhu, P., Thirumoorthi, C., Batumalay, M., & Xu, Z. (2024). Combined Fire Fly–Support Vector Machine Digital Radiography Classification (FF-SVM-DRC) Model for Inferior Alveolar Nerve Injury (IANI) Identification. *Journal of Applied Data Sciences*, *5*(3), 1363-1375.

[38]. Karthikeyan, T., & Thangaraju, P. (2014). PCA-NB algorithm to enhance the predictive accuracy. *Int. J. Eng. Tech*, *6*(1), 381-387.

[39]. Karthikeyan, T., & Thangaraju, P. (2015). Best first and greedy search based CFS-Naïve Bayes classification algorithms for hepatitis diagnosis. *Biosciences and Biotechnology Research Asia*, *12*(1), 983-990.

[40]. Karthikeyan, T., & Manikandaprabhu, P. (2015). Embedded Zero Tree Wavelet based Artificial Neural Network Image Classification Algorithm-A Study. *Indian Journal of Science and Technology*, *8*(20), 1.

[41]. Suresh, M., Sinha, A., & Aneesh, R. P. (2019). Real-time hand gesture recognition using deep learning. *International Journal of Innovations and Implementations in Engineering*, *1*, 11-15.

[42]. Ismail, A. P., Abd Aziz, F. A., Kasim, N. M., & Daud, K. (2021, February). Hand gesture recognition on python and opencv. In *IOP conference series: Materials science and engineering* (Vol. 1045, No. 1, p. 012043). IOP Publishing.

