Deep Learning Based Web Application for Hand Sign Recognition

Mrs.Sonali Salunkhe[1], Dr. Dinesh Hanchate[2], Dr. Sachin Bere[3]

[1] Student, Dept. of Computer Engineering, Dattakala Group of Institution, Maharashtra, India
[2] Professor, Dept. of Computer Engineering, Dattakala Group of Institution, Maharashtra, India
[3] Head of Department,Dept. of Computer Engineering, Dattakala Group of Institution, Maharashtra, India

ABSTRACT

Recognizing sign language remains one of the more complex tasks within the field of computer vision, especially as it serves as a vital means of communication for individuals who are deaf or mute. However, with the rapid growth of deep learning technologies, significant progress has been made in enabling machines to understand sign language more accurately. This study explores the application of Convolutional Neural Networks (CNNs) for identifying hand gestures corresponding to the American Sign Language (ASL) alphabet. Using a dataset of 1,815 static images representing all 26 letters of the English alphabet, the model was trained and tested, achieving an impressive validation accuracy of 94.34%. This performance surpasses many previously reported methods in the same domain.

Keyword - *Deep Learning, Neural Network, Convolutional Neural Network, American Sign Language, Hand Gesture Recognition* **.**

1. INTRODUCTION

The goal of this project is to create a web application powered by deep learning that can accurately identify and interpret alphabets from American Sign Language (ASL). For individuals who are unable to hear or speak, sign language serves as their primary means of communication. It is one of the oldest and most intuitive languages used to convey thoughts and emotions. However, since the majority of people are unfamiliar with sign language and trained interpreters are often not readily available, we have developed a real-time recognition system powered by neural networks. This project introduces an innovative approach to recognizing sign language alphabets. By leveraging computer vision and deep learning techniques, our system can interpret hand gestures and translate them into corresponding alphabet letters. The process begins by applying a filter to isolate the hand from the background. Once filtered, the image is processed by a classifier that determines which letter the hand gesture represents.

1.1 Objectives:

The goal of this project is to develop a real-time software application capable of identifying American Sign Language (ASL) hand gestures using advanced deep learning methods. The system specifically focuses on recognizing gestures that represent alphabets in the ASL system. By leveraging deep learning, the application accurately detects regions of interest (ROI) in video feeds to interpret the gestures effectively. This solution is designed to assist individuals who are deaf or speech-impaired, enabling smoother communication and greater accessibility. The system offers round-the-clock availability through an online platform, breaking down physical and communication barriers. To ensure robust security, the application is built using Python with the Django framework. Ultimately, this project presents a vision-based, AI-powered web application that combines the strengths of deep learning and Python to deliver a reliable, secure, and socially impactful communication tool.

2. Literature Survey

Sign language serves as a vital communication bridge for individuals with hearing and speech impairments. Among the various sign systems, American Sign Language (ASL) holds a prominent place. However, accurately interpreting

ASL gestures using computer vision technologies remains a complex challenge due to high variability in gesture forms and subtle intraclass distinctions (Dong et al., 2015; Bheda & Radpour, 2017).This study presents a method for recognizing ASL alphabets by employing Convolutional Neural Networks (CNNs), a type of deep learning architecture that has demonstrated remarkable performance in image classification tasks (Krizhevsky et al., 2012). Initially, a pre-processing phase is carried out on a dataset similar to MNIST, where the input images of hand gestures are cleaned and normalized to enhance recognition accuracy (Kalbhor & Deshpande, 2018; Ramzan et al., 2018).Once pre-processed, the CNN model extracts and analyzes significant visual features from each image. These features help the network to learn and distinguish between the various alphabetic gestures. In the final stage, the model's performance is assessed using two key metrics: accuracy and Area Under the Curve (AUC) score. The proposed model demonstrated excellent performance, achieving an AUC score of 0.9981 and an overall classification accuracy of 99.63%, indicating its effectiveness for real-world applications in sign language translation (Rao et al., 2018; Garcia & Viesca, 2016).This research highlights the capability of CNNs in addressing the complexity of ASL recognition and supports the growing integration of AI-based systems in assistive technologies for the hearing and speech impaired.

Globally, individuals who are deaf or mute face significant challenges in everyday communication, primarily due to the scarcity of skilled sign language interpreters. To bridge this communication, divide between sign language users and non-users, this study introduces an innovative hand gesture recognition system specifically designed for Bangla Sign Language. The approach involves detecting hand gestures using a combination of HSV and YCbCr color spaces for robust hand segmentation. Our system is capable of identifying thirty-seven unique Bangla sign language characters, encompassing eight vowels and twenty-nine consonants, through the application of deep convolutional neural networks (CNNs). Each of the 37 classes corresponds directly to the respective Bangla alphabets.To support this gesture recognition, a novel dataset comprising 3,219 images was developed, collected from six different individuals to ensure diversity and accuracy. Utilizing this dataset, the proposed model achieved an impressive recognition accuracy of 99.22%. This high level of precision demonstrates the potential of deep learning frameworks in enhancing communication accessibility for the Bangla-speaking deaf community.This work aligns with prior research efforts that have applied CNNs for sign language recognition in different languages and contexts (Bheda & Radpour, 2017; Rao et al., 2018; Dong, Leu, & Yin, 2015). Moreover, the use of tailored datasets has proven critical for achieving high performance in gesture recognition tasks (Barczak et al., 2011; Ahmed et al., 2019). Our study thus contributes a valuable resource and method to the ongoing development of assistive technologies for sign language users.

3. Proposed System

The proposed system utilizes a specially curated and pre-processed image dataset tailored to the specific application domain. This customization contributes to improved model accuracy. Moreover, the system architecture is designed to minimize the need for manual intervention, thus enhancing automation. It also operates efficiently on low computational resources, making it suitable for deployment in constrained environments. Additionally, the training process is optimized to require comparatively less time, thereby accelerating the development cycle (Sharma et al., 2021).

3.1 Requirement Analysis

Anaconda is a widely used Python distribution tailored for scientific computing. It streamlines the process of managing packages and deploying applications, making it especially beneficial for data science tasks. This distribution comes equipped with a variety of essential data science libraries and tools, and it is compatible with Windows operating systems (Anaconda, 2024).

Python is a versatile and high-level programming language renowned for its simplicity and readability. Its syntax relies heavily on indentation, which enhances the clarity of the code. Python is dynamically typed and uses automatic memory management. It supports several programming paradigms, such as object-oriented, procedural, and functional programming, making it adaptable for many types of software development (Van Rossum & Drake, 2009).

Bootstrap is an open-source front-end framework designed to facilitate the development of responsive and mobile-first websites. It provides a collection of reusable code components in HTML, CSS, and JavaScript that help developers create consistent and visually appealing web interfaces, including elements like forms, buttons, navigation bars, and typography (Bootstrap, 2024).

Django is a high-level, open-source web framework written in Python. It adopts the model–template–views (MTV) architectural pattern, enabling rapid development of secure and maintainable web applications. Django is actively

maintained by the Django Software Foundation, a non-profit organization established to support its development and growth (Django Software Foundation, 2024).

3.2 Algorithm and Mathematical Model

Sign language recognition is the method of translating hand signs and gestures made by individuals into readable text. This technology plays a vital role in closing the communication gap between those who are speech impaired and the wider community. By leveraging image processing techniques and deep learning models, the system identifies each gesture and accurately matches it to a corresponding letter from the trained dataset. This way, raw hand gesture images are transformed into meaningful alphabetic characters. People with speech disabilities rely on hand movements to express themselves, but these are often difficult for the general public to understand. Therefore, a system capable of interpreting these gestures and converting them into text is essential. It acts as a communication bridge, enabling better interaction and understanding between differently-abled individuals and those without impairments.

Fig -1: Proposed Model

As per fig.1 proposed model consist of following steps:

Input Image

The process begins with an input image, which may contain one or more objects of interest. These could be shapes, items, or any visual content captured by a camera or loaded from a dataset.

Segmentation

In this step, the image is processed to segment or isolate individual objects from the background. This is done to simplify the image and extract only the parts relevant for analysis. Common segmentation techniques include thresholding, edge detection, and region-based methods. To reduce noise and irrelevant background data, improving focus on the object(s) of interest.

Feature Extraction

Once the objects are isolated, the next step is to extract features, measurable properties or patterns that describe the object. These features might include:

Shape, Texture, Colour, Edges, Moments (e.g., Hu moments), Histogram of Oriented Gradients (HOG)

These features are then represented as a feature vector (shown as x1, x2,...,x_n) a numerical representation of the object.

Classification

The extracted feature vector is passed to a classifier, which uses machine learning or deep learning algorithms to determine the type of object. The classifier labels the object based on training data, resulting in the final object type output.

Convolutional Neural Networks (CNNs) are made up of several layers, including convolutional and subsampling (or pooling) layers, which may be followed by fully connected layers. In this context, we are focusing on two-dimensional (2D) CNNs, which are commonly used for processing 2D images. While CNNs can also be extended to 1D or 3D for other data types, the fundamental ideas remain consistent across these variations.

A typical input to a convolutional layer is represented as a 3D image tensor of size (r×c×n), where:

r: number of rows (height),

c: number of columns (width),

n: number of channels (e.g., RGB channels).

The convolutional layer contains a set of K learnable filters (also called kernels), each with dimensions (kr×kc×kn). These filters are responsible for detecting specific patterns or features in the input image.

To manage edge cases where the filters extend beyond the boundaries of the image, zero-padding is commonly applied. Padding the image with (pr,pc)pixels allows filters to slide over the edge pixels without reducing the spatial dimensions of the output. The padding size ppp is typically chosen as half of the filter dimensions: p=(kr/2,kc/2).

Another important parameter is the stride sss, which determines how many pixels the filter moves at each step across the image.After applying the convolution operation with K filters, the output consists of Kdifferent feature maps. The spatial dimensions (rows and columns) of the k-th feature map, denoted by M^k_r and M^k_c are calculated as follows:

$$M^k_r = (r - k_r + 2p_r)/s_r) + 1$$

$$M^k_c = (c - k_c + 2p_r)/s_c) + 1$$

Here, s_r and s_c are the strides in the row and column directions, respectively.
Once convolution is performed, a non-linear activation function is typically applied. A commonly used function is the Rectified Linear Unit (ReLU), which helps the model learn complex patterns by introducing non-linearity and speeding up the training process.Pooling layers are usually added after activation layers to reduce the spatial size of the feature maps. This helps in decreasing computational load and makes the features more robust by extracting dominant features. Pooling also allows deeper layers to process larger receptive fields while maintaining critical spatial hierarchies.As the image data passes through successive convolution, activation, and pooling layers, increasingly abstract and complex features are extracted. These intermediate feature maps are then forwarded through additional layers.In some architectures, fully connected layers are placed after the convolutional layers to classify the extracted features into desired categories. These layers output a final vector of size N, where N corresponds to the number of classes in the classification task.To train the CNN model effectively, a loss function is used to measure how well the predicted output matches the true labels. One common loss function is the Mean Squared Error (MSE), especially in regression-like tasks. The MSE loss is defined as: $err = 1/2 \sum(y - y\hat{})^2$
Where:

y: actual label,
y^: predicted output.

This error is minimized during training to improve the performance of the model.

4. CONCLUSIONS

The proposed system aims to completely replace conventional model training methods. It requires just a single interface for image upload, making the process remarkably simple. This innovation paves the way for a new era of human-computer interaction, where physical contact with devices is no longer necessary. With the power of deep learning, users can effortlessly control and operate a computer, making the technology accessible to everyone.

6. REFERENCES

[1] Ahmed, S., Islam, M., Hassan, J., Ahmed, M. U., Ferdosi, B. J., Saha, S., & Shopon, M. (2019). Hand sign to Bangla speech: A deep learning in vision-based system for recognizing hand sign digits and generating Bangla speech. *arXiv preprint arXiv:1901.05613*.

[2] Barczak, A., Reyes, N., Abastillas, M., Piccio, A., & Susnjak, T. (2011). A new 2D static hand gesture colour image dataset for ASL gestures.

[3] Bheda, V., & Radpour, D. (2017). Using deep convolutional networks for gesture recognition in American Sign Language. *arXiv preprint arXiv:1710.06836*.

[4] Chollet, F., et al. (2015). *Keras*. https://keras.io

[5] Dong, C., Leu, M. C., & Yin, Z. (2015). American sign language alphabet recognition using Microsoft Kinect. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (pp. 44–52).

[6] Garcia, B., & Viesca, S. A. (2016). Real-time American sign language recognition with convolutional neural networks. *Convolutional Neural Networks for Visual Recognition*, 2.

[7] Kalbhor, S. R., & Deshpande, A. M. (2018). Digit recognition using machine learning and convolutional neural network. In *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)* (pp. 604–609). IEEE.

[8] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems* (pp. 1097–1105).

[9] Kuznetsova, A., Leal-Taixé, L., & Rosenhahn, B. (2013). Real-time sign language recognition using a consumer depth camera. In *Proceedings of the IEEE International Conference on Computer Vision Workshops* (pp. 83–90).

[10] Ramzan, M., Khan, H. U., Akhtar, W., Zamir, A., Awan, S. M., Ilyas, M., & Mahmood, A. (2018). A survey on using neural network-based algorithms for handwritten digit recognition. *Environment*, 9(9).

[11] Rao, G. A., Syamala, K., Kishore, P., & Sastry, A. (2018). Deep convolutional neural networks for sign language recognition. In *2018 Conference on Signal Processing and Communication Engineering Systems (SPACES)* (pp. 194–197). IEEE.

[12] Anaconda. (2024). *Anaconda distribution*.https://www.anaconda.com/products/distribution

[13] Bootstrap. (2024). *Introduction to Bootstrap*. https://getbootstrap.com

[14] Django Software Foundation. (2024). *The web framework for perfectionists with deadlines*. https://www.djangoproject.com

[15] Van Rossum, G., & Drake, F. L. (2009). *Python 3 Reference Manual*. CreateSpace.