CONVOLUTIONAL NEURAL NETWORK-BASED REAL-TIME SIGN LANGUAGE RECOGNITION AND TRANSLATION

Authors

Prof. Shruthi S V
Assistant Professor
Department of CS & IT
Jain University
Bangalore, Karnataka, India
svsblg2023@gmail.com

Prof. Nagaratnamma S M
Assistant Professor
Department of Computer Science
Government First Grade College
Bangalore, Karnataka, India

Prof. Nalini R
Assistant Professor
Department of Computer Science
Government First Grade College
Bangalore, Karnataka, India

Abstract

Introduction: Context focuses on the development of a computer application and the training of a model specifically designed to interpret real-time video footage of hand gestures in American Sign Language (ASL). The primary objective is to create a system that can accurately recognize and translate these gestures into corresponding text formats on the screen. To achieve this, we have successfully implemented a set of 27 symbols representing the alphabet (A-Z) in ASL, along with a symbol for blank or no gesture. Our project aims to provide a reliable and efficient tool for facilitating communication between ASL users and non-ASL speakers.

Background: In the late 1990s, the first attempts were made to recognize sign language using electrochemical devices as the primary method. Parameters such as hand position, angle, and movement were investigated using these devices. Glove-based systems were developed based on this approach, requiring signers to wear cumbersome devices. However, these systems faced challenges in terms of accuracy and efficiency.

Another approach mentioned involved analyzing video clips of different sign language gestures and producing audio expressions based on the analysis. However, there were issues with the frame rate of the animation, which had to be manually adjusted to enhance understanding. Another system called the "Intelligent Assistant" was developed to facilitate communication with D&D personnel. It captured sound through a microphone and converted it into text using

Microsoft's Voice Command and Control Engine. However, this system suffered from inefficiency in noisy environments and generated incorrect outputs due to input noise.

Research Proposal: To provide a reliable and efficient tool for facilitating communication between ASL users and non-ASL speakers.

Conclusion: Our focus was on recognizing ASL alphabets. Our rigorous efforts have achieved an impressive final accuracy of 98.0% on our dataset. This project operates in real-time, allowing for immediate recognition and interpretation of ASL alphabets. By leveraging computer vision techniques and machine learning algorithms, we have made substantial progress in bridging the communication gap between Deaf and Mute individuals and the rest of the population. Compared to all the related works which were done before, we have tried to significantly improve the accuracy and have been successful.

Overall, our work signifies a significant step forward in ASL recognition technology, contributing to improved communication and inclusivity for Deaf and Mute individuals. We are confident that our project holds great potential for practical applications and can be further refined to achieve even higher accuracy in the future.

Keywords: Convolution Neural Networks (CNN), American Sign Language (ASL), Hand gesture, Gaussian Blur

I. INTRODUCTION

Approximately 15 percent of the global population is affected by various disabilities, with over 466 million individuals being deaf, accounting for more than five percent of the total population. By 2050, the world population is projected to be around 2.7 times larger than it was in 2000, with an estimated growth of 500 million people. Among these numbers, at least 70 million people experience speech and hearing impairments, leading to challenges in interacting with others, particularly in areas such as employment, education, healthcare, and transportation. Communication between individuals with hearing or speech impairments and those without disabilities often requires the assistance of interpreters. Nevertheless, assigning and training interpreters in underserved and remote areas can be challenging. Consequently, these populations lack a vital necessity that is essential for leading a normal life, just like their counterparts in developing, underdeveloped, and affluent nations. People with hearing or speech impairments predominantly rely on sign language as their primary mode of communication. However, the medium of sign language poses challenges in communication between individuals with hearing impairments and those without. Implementing a digital Sign Language Interpretation system can address this communication barrier. In our project, our main objective is to develop a model that can accurately recognize and interpret finger spelling-based hand gestures, which are used to spell out words by combining individual gestures. The specific set of gestures we aim to train and recognize are depicted in the image below. Our focus is on training the model to accurately identify each gesture and combine them to form complete words. By achieving this, we aim to provide a reliable and efficient tool for improving communication and understanding between users of sign language and non-sign language speakers.

II. BACKGROUND

In the late 1990s, the first attempts were made to recognize sign language using electrochemical devices as the primary method. Parameters such as hand position, angle, and movement were investigated using these devices. Glove-based systems were developed based on this approach, requiring signers to wear cumbersome devices. However, these systems faced challenges in terms of accuracy and efficiency. Another approach mentioned involved analyzing video clips of different sign language gestures and producing audio expressions based on the analysis. However, there were issues with the frame rate of the animation, which had to be manually adjusted to enhance understanding. Another system called the "Intelligent Assistant" was developed to facilitate communication with D&D personnel. It captured sound through a microphone and converted it into text using Microsoft's Voice Command and Control Engine. However, this system suffered from inefficiency in noisy environments and generated incorrect outputs due to input noise.

In comparison to the aforementioned systems, a more complex approach was presented. It involved the use of a glove with different dots on each finger to display signs. Real-time photos of the signs were captured using a digital input, and the program examined the dot patterns in the image file to understand the displayed sign. The system recognized pre- recorded wav files associated with regular language signs, and clustering techniques were used to group the dots. The results of this clustering were mapped to predefined tables. For a specific case involving Bengali numbers ranging from 1 to 10, a simple system was developed without the need for advanced intelligence. Neural networks (NN) were also employed for sign recognition. A hybrid vocabulary of static and dynamic hand gestures was classified using Radial Basis Functions and Bayesian Classifiers. Additionally, it was demonstrated that NN could be utilized to classify Japanese Sign Language (JSL). Another recent approach proposed utilized a slow learning algorithm for recognizing Bangladesh Sign Language (BdSL) in cases where the test case was not properly described.

III. EXPERIMENTAL SETUP

In this context, we have trained our own dataset through the webcam present in our laptop. We have trained each alphabet from A-Z and numbers from 0-9 through different hand sign gestures and also a space symbol which is required to provide a gap between each word. The IDE which we used to execute our program is Jupyter Notebook. In order to pre-process the dataset and train our model, we will utilize various Python libraries and frameworks that are commonly used in machine learning. These libraries and frameworks include OpenCV, TensorFlow, Keras, NumPy, and Pandas. These tools provide essential functionality for tasks such as data pre-processing, feature extraction, model training, and evaluation. To install these libraries and frameworks, we can leverage the pip package manager, which is commonly used in Python for installing third-party packages. Pip allows us to easily install and manage the dependencies required for our project. By executing the appropriate pip commands, we can ensure that all the necessary libraries and frameworks are installed and available for our project. By utilizing these powerful Python libraries and frameworks, we can streamline the pre-processing of our dataset and effectively train our machine-learning model to recognize and interpret fingerspelling-based hand gestures.

IV. MATERIALS AND METHODS

While numerous sign language datasets are available online, most of them focus on specific languages, numbers, or characters. However, our system aims to interpret the sign language of the alphabet and translate it into English words. In order to work effectively and efficiently, we have created our own dataset tailored to our requirements.

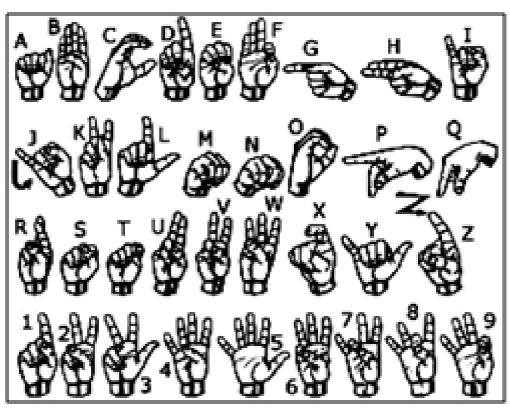

Figure 1: Data Set of Alphabet Generation

In our dataset, each alphabet contains a different sign and based on that sign we trained some 800 images of each alphabet. These images were used for training and testing our CNN (Convolutional Neural Network) model. Although our dataset may seem small in size, our model has demonstrated excellent results even with this limited dataset. The system we developed is based on computer vision, specifically using bare hands to represent the signs. This approach eliminates the need for artificial devices, making the interaction more natural and intuitive.

1. **Dataset Generation:** To create our dataset, we couldn't find pre-existing datasets in the form of raw images that met our requirements. The available datasets were mostly provided as RGB values, which did not suit our needs. Therefore, we decided to generate our own dataset using the OpenCV library. We captured approximately 800 images for each symbol in American Sign Language (ASL) for training purposes, and around 200 images per symbol for testing purposes. The images were captured using the webcam of our machine. In each frame, we defined a region of interest (ROI) denoted by a blue bounded square. This ROI represents the specific area containing the hand gesture.

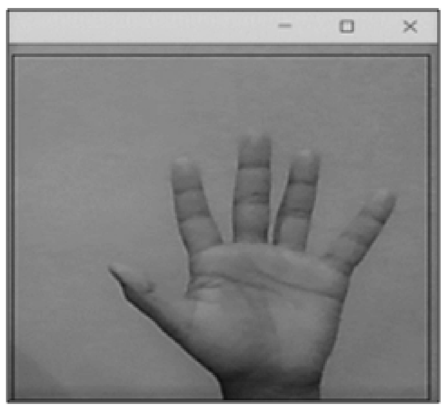

Figure 2: ROI represents the Specific Area Containing the Hand Gesture

From the captured frame, we extracted the ROI, which was initially in RGB format, and converted it into a grayscale image. This grayscale image simplifies the image processing and analysis.

Figure 3: Gaussian Blur Appears Smoother and Less Noisy

To enhance the image features, we applied a Gaussian blur filter to the grayscale image. The Gaussian blur helps in extracting various features of the hand gesture. The resulting image after applying the Gaussian blur appears smoother and less noisy.

By following these steps, we successfully created our own dataset of hand gesture images, which we used for training and testing our ASL sign language and detection system.

2. **Gesture Classification:** For gesture classification in our project, we adopted a two-layered algorithmic approach. Here are the details of each layer.

   - **Algorithm Layer 1:** The first step involves applying a Gaussian blur filter and thresholding to the frames captured with OpenCV. This helps in extracting the relevant features from the image and reducing noise. The processed image is then passed to a Convolutional Neural Network (CNN) model for prediction. The CNN model analyzes the image and predicts the corresponding symbol or letter. If a particular letter is consistently detected for more than 50 frames, it is considered valid and printed. These valid letters are then taken into consideration for forming words. We also incorporate the "blank" symbol to represent spaces between words.

   - **Algorithm Layer 2:** In this layer, we focus on detecting and classifying various sets of symbols that exhibit similar results during detection. We create separate classifiers for each set of symbols to differentiate and classify them accurately. By implementing these two algorithmic layers, we achieve effective gesture classification and prediction in our system, enabling users to form words and sentences using sign language.

Figure 4: Represents Blank Screen

Figure 5: Recognition of Letter 0

Figure 6: Recognition of Letter L

To improve the accuracy of symbol detection and prediction, we have implemented a two- layered algorithmic approach that specifically addresses cases where certain symbols may show similarities and result in incorrect classifications. Based on our testing, we have identified the following problematic symbol sets:

- **Symbol D:** It sometimes gets misclassified as R or U.
- **Symbol U:** It sometimes gets misclassified as D or R.
- **Symbol I:** It sometimes gets misclassified as T, D, K, or another I.
- **Symbol S:** It sometimes gets misclassified as M or N.

To handle these cases and ensure accurate classification, we have developed three separate classifiers, each targeting a specific set of symbols: Classifier for {D, R, U}: This classifier focuses on distinguishing between the symbols D, R, and U, ensuring that each symbol is correctly identified.

Classifier for {T, K, D, I}: This classifier is designed to classify the symbols T, K, D, and I accurately, minimizing any misclassifications within this set.

Classifier for {S, M, N}: This classifier specifically addresses the symbols S, M, and N, ensuring precise classification and reducing the occurrence of misclassifications. By implementing these specialized classifiers for the respective symbol sets, we aim to enhance the overall accuracy and reliability of our symbol detection and prediction system.

V. EXPERIMENTAL PROCEDURE

In our symbol verification and prediction process, we have implemented several steps to ensure accurate results and improve the user experience. Here are the implementation steps in detail

1. **Counting and Thresholding:** We maintain a count of detections for each symbol. When the count for a specific symbol exceeds a predefined value (set as 50 in our code), and no other symbol is detected within a certain threshold (set as 20), we consider the symbol valid and print it. The detected symbol is then added to the current string. If the conditions are not met, we clear the current dictionary to avoid incorrect predictions.

2. **Blank Detection:** We also detect blank or plain background regions. If the count of blank detections exceeds a specific value and the current buffer is empty, no spaces are detected. This helps in differentiating between words and ensures accurate word formation.

3. **End of Word Prediction:** When space is detected, we predict the end of a word by printing space and appending the current string to the sentence below.

4. **Autocorrect Feature:** To assist users in reducing spelling mistakes and predicting complex words accurately, we incorporate an autocorrect feature. We use the Hunspell_suggest Python library to suggest correct alternatives for each incorrect input word. A set of words matching the current word is displayed, allowing the user to select a word to append to the current sentence.

5. **Training and Testing:** For training and testing our model, we preprocess the input images by converting them to grayscale and applying a Gaussian blur to remove noise. We then apply adaptive thresholding to extract the hand from the background. The images are resized to a standard size of 128 x 128 pixels.

   The pre-processed images are fed into our model for training and testing, following the mentioned operations. The prediction layer of our model estimates the likelihood of the image belonging to each class. To achieve this, we normalize the output between 0 and 1 using the SoftMax function, ensuring that each class's values sum to 1. To optimize the performance of our neural network, we trained the model using labeled data. We utilized cross-entropy, a continuous function that measures the difference between the predicted and labeled values, as our performance measurement for classification. We used the Adam Optimizer, a gradient descent optimizer, to minimize the cross-entropy and adjust the weights of our neural networks accordingly. Tensor Flow provides an inbuilt function to calculate the cross entropy, simplifying the optimization process.

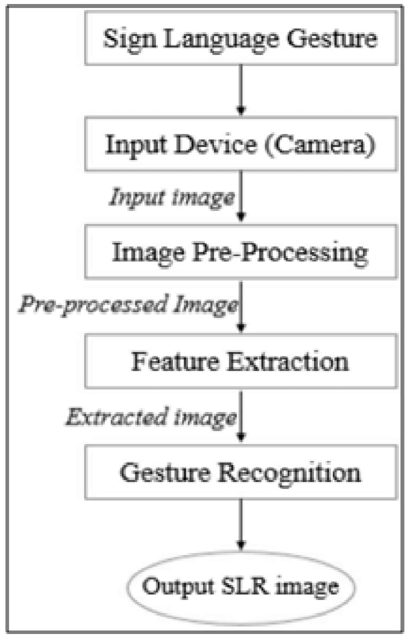

Figure 7: Processing Steps for the Vision Sensor-Based SLR System

VI. RESULTS AND DISCUSSION

To enhance the accuracy and precision of our system, we implemented two layers of algorithms. These algorithms allow us to verify and predict symbols that share similarities with each other. By doing so, we have significantly improved our ability to detect almost all the ASL symbols, provided that they are presented clearly, without any background noise, and under adequate lighting conditions, with an accuracy of 95.4 percent.

Figure 8: Depicts Word and Sentence Suggestions for the Next Set of Letters

Figure 9: Depicts Sentence Suggestion

It is important to note that while our project demonstrates high accuracy, its performance may be affected by certain factors such as unclear hand gestures, the presence of background noise, and insufficient lighting. We acknowledge that further improvements can be made to handle these challenges effectively.

VII. CONCLUSION

In this work, we have successfully developed a functional real-time vision-based American Sign Language (ASL) recognition system specifically designed for Deaf and Mute individuals. Our focus was on recognizing ASL alphabets. Our rigorous efforts have achieved an impressive final accuracy of 98.0% on our dataset. This project operates in real-time, allowing for immediate recognition and interpretation of ASL alphabets. By leveraging computer vision techniques and machine learning algorithms, we have made substantial progress in bridging the communication gap between Deaf and Mute individuals and the rest of the population. Compared to all the related works which were done before, we have tried to significantly improve the accuracy and have been successful.

Overall, our work signifies a significant step forward in ASL recognition technology, contributing to improved communication and inclusivity for Deaf and Mute individuals. We are confident that our project holds great potential for practical applications and can be further refined to achieve even higher accuracy in the future.

REFERENCES

[1] Janeera, D. A., Raja, K. M., Pravin, U. K. R., & Kumar, M. K. (2021, April). Neural network based real time sign language interpreter for virtual meet. In 2021 5th International Conference on Computing Methodologies and Communication (ICCMC) (pp. 1593-1597). IEEE.

[2] Gedkhaw, E. (2022). The performance of Thai sign language recognition with 2D convolutional neural network based on NVIDIA Jetson nano developer kit. TEM Journal, 11(1), 411-419.

[3] Shahriar, S., Siddiquee, A., Islam, T., Ghosh, A., Chakraborty, R., Khan, A. I., ... & Fattah, S. A. (2018, October). Real-time american sign language recognition using skin segmentation and image category classification with convolutional neural network and deep learning. In TENCON 2018-2018 IEEE Region 10 Conference (pp. 1168-1171). IEEE.

[4] He, S. (2019, October). Research of a sign language translation system based on deep learning. In 2019 International conference on artificial intelligence and advanced manufacturing (AIAM) (pp. 392-396). IEEE.

[5] Jarabese, M. B. D., Marzan, C. S., Boado, J. Q., Lopez, R. R. M. F., Ofiana, L. G. B., & Pilarca, K. J. P. (2021, November). Sign to speech convolutional neural network-based filipino sign language hand gesture recognition system. In 2021 International Symposium on Computer Science and Intelligent Controls (ISCSIC) (pp. 147-153). IEEE.

[6] Ko, S. K., Son, J. G., & Jung, H. (2018, October). Sign language recognition with recurrent neural network using human keypoint detection. In Proceedings of the 2018 conference on research in adaptive and convergent systems (pp. 326-328).

[7] Rwelli, R. E., Shahin, O. R., & Taloba, A. I. (2022). Gesture based Arabic Sign Language Recognition for Impaired People based on Convolution Neural Network. arXiv preprint arXiv:2203.05602.

[8] Qin, W., Mei, X., Chen, Y., Zhang, Q., Yao, Y., & Hu, S. (2021, December). Sign language recognition and translation method based on VTN. In 2021 International Conference on Digital Society and Intelligent Systems (DSInS) (pp. 111-115). IEEE.

[9] Dabwan, B. A. (2020). Convolutional neural network-based sign language translation system. International Journal of Engineering, Science and Mathematics, 9(6), 47-57.

[10] Subburaj, S., & Murugavalli, S. (2022). Survey on sign language recognition in context of vision-based and deep learning. Measurement: Sensors, 23, 100385.

[11] Hoque, O. B., Jubair, M. I., Islam, M. S., Akash, A. F., & Paulson, A. S. (2018, December). Real time bangladeshi sign language detection using faster r-cnn. In 2018 international conference on innovation in engineering and technology (ICIET) (pp. 1-6). IEEE.

[12] Yemenoglu, I. H., Shah, A. S., & Ilhan, H. (2021, October). Deep Convolutional Neural Networks-Based Sign Language Recognition System. In 2021 IEEE 12th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON) (pp. 0573-0576). IEEE.

[13] Cui, R., Liu, H., & Zhang, C. (2019). A deep neural framework for continuous sign language recognition by iterative training. IEEE Transactions on Multimedia, 21(7), 1880-1891.

[14] Al-Obodi, A. H., Al-Hanine, A. M., Al-Harbi, K. N., Al-Dawas, M. S., & Al-Shargabi, A. A. (2020). A Saudi Sign Language recognition system based on convolutional neural networks. Department of Information Technology, College of Computer, Qassim University, Buraydah, Saudi Arabia.

[15] Limaye, H., Shinde, S., Bapat, A., & Samant, N. (2022). Sign Language Recognition using Convolutional Neural Network with Customization. Available at SSRN 4169172.

[16] Amrutha, K., & Prabu, P. (2023). Evaluating the pertinence of pose estimation model for sign language translation. International Journal of Computational Intelligence and Applications, 22(01), 2341009.

[17] Panneer, S. E., & Sornam, M. (2022, April). Recent Advances in Sign Language Recognition using Deep Learning Techniques. In 2022 6th International Conference on Trends in Electronics and Informatics (ICOEI) (pp. 1261-1265). IEEE.

[18] Elakkiya, R., Vijayakumar, P., & Kumar, N. (2021). An optimized Generative Adversarial Network based continuous sign language classification. Expert Systems with Applications, 182, 115276.

[19] Bhadra, R., & Kar, S. (2021, January). Sign language detection from hand gesture images using deep multi-layered convolution neural network. In 2021 IEEE Second International Conference on Control, Measurement and Instrumentation (CMI) (pp. 196-200). IEEE.

[20] Katoch, S., Singh, V., & Tiwary, U. S. (2022). Indian Sign Language recognition system using SURF with SVM and CNN. Array, 14, 100141.

[21] Khedkar, V. N., Prasad, A., Mishra, A., Saha, V., & Kumar, V. (2021). Analysis of recent trends in continuous sign language recognition using NLP. Library Philosophy and Practice, 0_1-25.

[22] Wang, F., Zhao, S., Zhou, X., Li, C., Li, M., & Zeng, Z. (2019). An recognition–verification mechanism for real-time Chinese sign language recognition based on multi-information fusion. Sensors, 19(11), 2495.

[23] Alawwad, R. A., Bchir, O., & Ismail, M. M. B. (2021). Arabic sign language recognition using faster R-CNN. International Journal of Advanced Computer Science and Applications, 12(3).

