Global Journal of Engineering Innovations & Interdisciplinary Research

Original Article

Sign Language Recognition And Translation Into Text And Speech

Razia S, Ajay Kumar C, Iswarya Y, Ganesh B, Indra C, Naveen C, Jeelan Basha P.

Computer Science & Engineering, Gates Institute Of Technology, Gooty, Andhra Pradesh, India

Correspondence

Razia S

Computer Science and Engineering, Gates Institute of Technology, Gooty, India

Abstract

This project aims to allow easy and low- cost communication between people with hearing/speech disabilities and people without these disabilities, by introducing an IOT device medium. We propose a simple system that converts images to text which in turn convert speech. Images are acquired by a low-cost camera, these images are processed and used for training a neural network model. This trained model then converts the image inputs into suitable textual outputs. Once the text outputs are displayed, we can then convert them into speech. The IoT medium which will process all these tasks and make our project feasible is a raspberry-pi 3. The raspberry- pi is a credit card-sized single-board micro- computer chip that is powerful enough to run neural networks and image processing. This micro-computer board also has general purpose I/O (GPIO) pins, allowing us to attach hardware components such as a camera, display, and speaker. The finished product is meant to be affordable and efficient.

- Received Date: 09 Feb 2025
- Accepted Date: 30 Mar 2025
- Publication Date: 02 Apr 2025

Keywords

Raspberry Pie, Open CV, Neural Network, Hand Gesture recognition.

Introduction

Verbal communication performed by humans is one of the most unique traits in the entire animal kingdom. Humans have used communication as a tool to share and expand our knowledge of the world. It is safe to say that humans have created settlements, societies, technologies, strategies and more, only through efficient communication. In today's world, communication between individuals is essential to the development and maintenance of society. But unfortunately, some individuals with hearing/speech disabilities are unable to perform this basic human interaction. This barrier in communication alienates them from society and hinders effective communication. Since it is not feasible to assume that every person who communicates with such disabled individuals knows sign language, we need a method that will eradicate this communication barrier. In this project, we are proposing one such method.

Inspired by a personal encounter with a young child who relied on sign language for communication, highlighting the need for accurate and accessible translation tools. Sign language, used by hearing-impaired individuals, involves hand shapes, body movements, and facial expressions. Traditional methods for sign language recognition (SLR) rely on hand-crafted features, which struggle to generalize across diverse gestures and users. This project aims to develop a CNN-based system to automatically translate sign language into text and speech, improving communication and social inclusion for the deaf community.

Sign language recognition faces challenges in tracking hand regions, segmenting hand shapes from complex backgrounds, and recognizing gestures accurately. Motion trajectory recognition is further complicated by variations and occlusions of hands and body joints. Traditional methods like Gaussian Mixture Models with Hidden Markov Models (GMM-HMM) rely on hand-crafted features, which are time-consuming to design and often fail to generalize. These limitations necessitate a robust, automated approach to SLR that can handle real-time processing and improve accuracy.

The objective is to develop a CNN-based SLR model using multi-source visual inputs (color, depth, and skeleton images) from Microsoft Kinect. The system will automatically extract spatial-temporal features, eliminating the need for hand- crafted features. Designed for real-time efficiency, it will translate sign language into text and speech with high accuracy, outperforming traditional methods like GMM-HMM. This project aims to bridge the communication gap between hearing- impaired and non-impaired individuals, enhancing accessibility and social inclusion.

Citation: Razia S, Ajay Kumar C, Iswarya Y, et al. Block Chain For Safer & Traceable Indian Dairy Products. GJEIIR. 2025;5(2):34.

Methodology

Our Methodology comprises of following steps:

- Data collection
- Pre-processing
- Training
- Setting up the raspberry pi
- Recognition and Text-to-Speech

Modeling and analysis

Data Collection: Our project will use hand gesture recognition to form text and speech. To do this we need to train a neural network. To train a neural network we need training data.

For training data, we decided to use the gestures for the alphabets in the American sign language (ASL). We chose the ASL because of its simple gestures and because it is widely used and understood. Initially, we searched for different datasets in online platforms like Kaggle. But after in-depth research, we concluded that we could not use any datasets we found online because most of the datasets used different image quality cameras with different backgrounds which were not compatible with the images produced by our cameras. So, we decided to make our custom dataset that would use our camera and would therefore yield better results.

Pre-processing: As we have already done the data acquisition part in the previous stage, now we must process the data to reduce the dimensionality of the image and suppress the unwanted characteristics of the image so that our CNN based Computer vision works on the data set efficiently. Firstly, we make a directory where we will keep our processed training and testing data sets. Then we fetch the data from the directory where we have kept our acquired images in the previous stages. After fetching the data, we start our pre-processing part. We have used the OpenCV library for image processing in this project. For every image, we firstly fetch it from the path, then, we convert it into a grey image. This reduces the dimensionality of the image from BGR to Grey. After this, we apply Gaussian Blur, which is used for removing noise from the image. Gaussian to Blur is widely used in the pre-processing stage in computer vision algorithms to enhance image structures at different levels. Next, we have used Adaptive Thresholding. This is useful when we have different lighting conditions in different areas. We have used this algorithm to increase the efficiency of the model. This algorithm determines the threshold for a pixel, based on a small region around it. Adaptive Thresholding is used for segmentation. In this, we set a threshold value for a pixel intensity, and according to that, the pixel is segmented as foreground and background. Adaptive thresholding sets the value of the threshold dynamically rather than using a global threshold value. Adaptive thresholding served as the best option for our project. Before this, we have tried the binary inversion technique, but data loss was significant, and it is not an adaptive method. After this, we split our data set into training and testing data set. This step is important to check the accuracy of the model. This can be used to minimize the effect of data discrepancy and better understanding the characteristics of the model. The pre-processing completes with this final step.

Training: Once the pre-processing stage was complete, we were left with training and testing datasets that contained the processed images of our gestures. Now we come to one of the main steps, the training of the neural network model. After all our research, we decided to go with the convolutional neural network or CNN for short. We chose this model because of its low computational needs and high performance. Compared to the usual artificial neural network the CNN significantly reduces the need for the number of neurons for image classification and we also get location invariant feature detection. This heavily impacts computational resources and at the same time does not compromise on accuracy. For feature extraction, we used two convolutions and two max-pooling layers. For the number of features, we chose a common number 32 that we observed in many similar research papers. Therefore, the model searches for 32 features in every image passed onto it. For our filter size, we chose 3 by 3 after performing many experiments. We set the activation functions for the convolution layers to the ReLU activation function. We chose the ReLU activation function as it introduces non-linearity and speeds up the computation. We chose a pooling size of 2 by 2 based on experimental results. We then added 2 dense layers with ReLU activation and the output layer with SoftMax activation to get a more normalized output. We trained the model to identify 11 classes and training the model with 1500 images of each gesture for 5 epochs. Out of the 1500 images 1125 images were used for making the training set and 375 images were used for making testing and validation sets. After many attempts at parameter tunings, we finally achieved an accuracy of 96.84 per cent while training and a 93.14 per cent accuracy when tested with new datasets. This model performed well under different lighting and background conditions. The model did not show any signs of overfitting either. We decided to use this model for our main code in the raspberry pi. This trained model was then saved as a JSON file and its weights were saved in h5 file format.

Recognition and Text to Speech: Once our model is trained then we load the trained model and weights onto our main code. The main code is where we will process the new images that the user inputs as gestures. First, we take images from the camera in real-time using modules like PiCamera and OpenCV. We use the PiCamera module so that the raspberry pi seamlessly acquires images from the camera attached to it. We found that using the regular OpenCV modules to acquire images from the camera performed poorly on the raspberry pi. The images are taken from a region of interest that we created to extract the hand gestures without many distractions from the background. These images are then pre-processed using the same techniques used to pre-process the training data in realtime. This ensures that there is no deviation between training data and our new testing data. After the images are converted in real-time into the black and white format that we wanted we pass the data into a test set, this test set is in a format that the CNN model was trained in. We then pass this test set into the trained model that we loaded earlier. The model produces a prediction in realtime which we then display on the camera screen for reference.

Existing System

TTraditional methods for sign language recognition rely on hand-crafted features and classifiers like HMM or GMM, which often result in limited accuracy and lack real-time processing capabilities. Vision- based systems use image processing techniques (e.g., OpenCV) or depth sensors (e.g., Kinect) but are expensive and sensitive to background clutter, making them less robust in dynamic environments. Deep learning-based systems leverage CNNs/RNNs for automatic feature extraction, offering improved accuracy but at the cost of high computational requirements and the need for large datasets. Sensor-based systems, such as those using wearable sensors (e.g., gloves),

provide precise motion tracking but are expensive, inconvenient, and not user- friendly. Commercial systems, while advanced, are often proprietary, expensive, and limited to specific sign languages, restricting their accessibility and adaptability.

Proposed System

The proposed system is a real time sign language recognition(SLR) system that utilizes Convolutional Neural Networks (CNNs) to recognize hand gestures from a webcam feed. Recognized gestures are converted into text and speech using google text-to-speech (Gtts), providing an intuitive and accessible interface for communication. The system captures live video feed from a webcam, isolates the hand region using background subtraction, and processes the frames through a pre-trained CNN model for gesture classification. By leveraging deep learning, the system automatically extracts features eliminating the need for hand- crafted features, and is trained on a custom dataset of hand gestures. It converts recognized gestures into text and synthesizes the text into speech using gTTS, enabling seamless communication. The system also features a user-friendly Tkinter-based GUI that displays recognized gestures, text, and speech output, making it easy to interact with. Additionally, the system is cost-effective and accessible, as it uses a standard webcam and open-source libraries, avoiding the need for specialized hardware like Kinect or wearable sensors.

Conclusion

The Sign Language Recognition to Text & Voice system recognizes hand gestures and converts them into text and voice in real-time. It leverages Convolutional Neural Networks (CNN) for high accuracy in gesture recognition. A user-friendly graphical interface ensures seamless conversion to text and voice. Empirical experiments on a hand gesture dataset show that the CNN-based approach

outperforms traditional methods. The system is designed to be scalable, maintainable, and easy to use. It facilitates communication between deaf-mute individuals and the general public. A Tkinter-based GUI enables users to interact with the system effortlessly. Users can upload datasets, train models, and perform real-time gesture recognition.

References

1. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In Advances in Neural Information Processing Systems, pages 1097–1105, 2012.
2. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-Based Learning Applied to Document Recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998.
3. T. Starner, J. Weaver, and A. Pentland. Real-Time American Sign Language Recognition Using Desk and Wearable Computer-Based Video. IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(12):1371–1375, 1998.
4. C. Vogler and D. Metaxas. Parallel Hidden Markov Models for American Sign Language Recognition. In Proceedings of the Seventh IEEE International Conference on Computer Vision, pages 116–122, 1999.
5. S. Ji, W. Xu, M. Yang, and K. Yu. 3D Convolutional Neural Networks for Human Action Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(1):221–231, 2013.
6. R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. arXiv preprint arXiv:1311.2524, 2013..
7. C. Farabet, C. Couprie, L.Najman, and Y. LeCun. Learning Hierarchical Features for Scene Labeling. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(8):1915–1929, 2013.
8. Mohammed Inayathulla and Karthikeyan C, "Image Caption Generation using Deep Learning For Video Summarization Applications" International Journal of Advanced Computer Science and Applications(IJACSA), 15(1), 2024. http://dx.doi.org/10.14569/IJACSA.2024.0150155.
9. Mohammed, I., Chalichalamala, S. (2015). TERA: A Test Effort Reduction Approach by Using Fault Prediction Models. In: Satapathy, S., Govardhan, A., Raju, K., Mandal, J. (eds) Emerging ICT for Bridging the Future - Proceedings of the 49th Annual Convention of the Computer Society of India (CSI) Volume 1. Advances in Intelligent Systems and Computing, vol 337. Springer, Cham. https://doi.org/10.1007/978-3-319-13728-5_25.
10. Inayathulla, M., Karthikeyan, C. (2022). Supervised Deep Learning Approach for Generating Dynamic Summary of the Video. In: Suma, V., Baig, Z., Kolandapalayam Shanmugam, S., Lorenz, P. (eds) Inventive Systems and Control. Lecture Notes in Networks and Systems, vol 436. Springer, Singapore. https://doi.org/10.1007/978-981-19-1012-8_18.