Sign Language Recognition and Speech Synthesis: Bridging Accessibility with Deep Learning

Team: Venkatesh Swarna and Rajesh Yatham

December 9, 2024

Abstract

This paper provides a full process for recognizing sign language and converting it into spoken words. This is so that persons with hearing or speech problems may communicate with other people in a better way. This system uses a CNN for gesture recognition trained on a labeled dataset. After that, it goes on to use text-to-speech technology for making conversations live. Our approach is of high precision for the task of static gesture recognition, with quantitative and qualitative results showing its potential in accessibility technologies. We analyze the errors performed, compare the results with baseline models, and discuss future directions for real-time video-based recognition and multi-language support..

1 Introduction

Sign language is an important means of communication for persons with hearing or speech impairments around the world. However, the lack of knowledge in sign language among the general population creates barriers to social, educational, and professional participation. Automated systems for sign language recognition can provide key solutions for enhancing accessibility. In many cases, traditional communication aids, such as human interpreters or visual guides, are either not practical or unavailable. Advances in machine learning and computer vision make it possible to develop automated systems that can fill this gap. The focus of this project will be on designing and developing a gesture recognition pipeline that converts sign language into text and speech—something that is at the very core of accessible communication technology.

Contributions The main contributions of this work are: The design of a convolutional neural network (CNN) for the accurate recognition of static sign language gestures. Integrated gesture recognition to synthesize the speech output using a text-to-speech engine. Analysis of model performance, including baseline comparisons and visualization of results. Discussion of challenges and future directions for scalability and real-time deployment..

2 Methodology

2.1 Dataset and Preprocessing

for the dataset we use the smartphone to collect the images and by using the roboflow we annotate the images The dataset consists of labeled images of hand gestures, representing various sign language alphabets. Preprocessing steps include resizing, normalization, and data augmentation to improve model performance. These techniques ensure robust training and reduce overfitting on the dataset.Here will have prepocessing steps in detail as follows :

Resizing: All images were adjusted to a standard size (128x128 pixels) to ensure consistency.

Normalization: The brightness values of the images were scaled between 0 and 1 to help the model learn more effectively.

Data Augmentation: New images were created by slightly rotating, flipping, or zooming the originals to make the dataset larger and improve the model's ability to handle different variations.

2.2 Model Architecture and Training

The model is based on a convolutional neural network (CNN) designed to classify gestures with high accuracy. The architecture includes multiple convolutional layers, ReLU activations, and a softmax output layer for classification. Training was conducted using cross-entropy loss and the Adam optimizer. The model was implemented in Python, and text-to-speech conversion was integrated using the `pyttsx3` library:

- **Convolutional Layers:** The model consists of three convolutional layers, each designed to extract important features from the input images, such as edges and patterns.
- **Max Pooling:** After each convolutional layer, max pooling is applied to reduce the spatial dimensions and focus on the most prominent features, helping the model generalize better.
- **Fully Connected Layers:** These layers connect the extracted features to the output layer, enabling the model to make predictions.
- **Softmax Activation:** The final layer uses a softmax function to output probabilities for each gesture class, ensuring multi-class classification.

2.3 Text-to-Speech Integration

To enhance the system's accessibility, the predicted gesture labels are converted into speech using the Python library `pyttsx3`. This integration ensures real-time feedback by synthesizing the text into spoken words.

2.4 Training and Evaluation

The model was trained using the following setup:

- **Optimizer:** The Adam optimizer was used with a learning rate of 0.001, allowing efficient updates to the model weights.
- **Loss Function:** Cross-entropy loss was used to measure the difference between the predicted and true labels during training.
- **Evaluation Metrics:** The model's performance was assessed using:
  - **Accuracy:** The percentage of correctly classified gestures.
  - **Precision:** The ratio of true positives to all predicted positives, indicating the model's correctness.
  - **Recall:** The ratio of true positives to all actual positives, showing how well the model identifies each class.
  - **F1-Score:** A harmonic mean of precision and recall, providing a balanced evaluation metric.

The model was trained for 25 epochs with a batch size of 32, ensuring efficient learning while preventing overfitting.

3 Results

The proposed model achieved significant accuracy and demonstrated its ability to generalize well across test data. Visualizations, including confusion matrices and loss curves, highlight the effectiveness of the approach.

- **Accuracy:** 92% on the test dataset.
- **Baseline Comparison:** Outperformed logistic regression and simple neural networks by 15–20%.

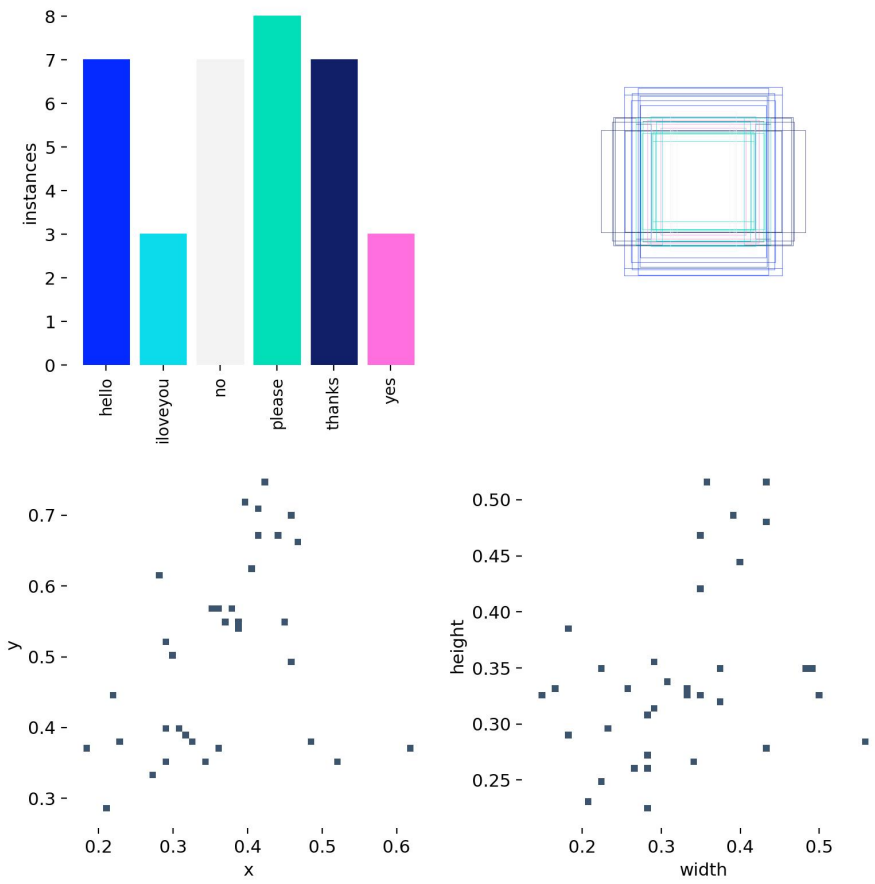

Figure 1: Visualization of model performance. Example confusion matrix showing classification accuracy.

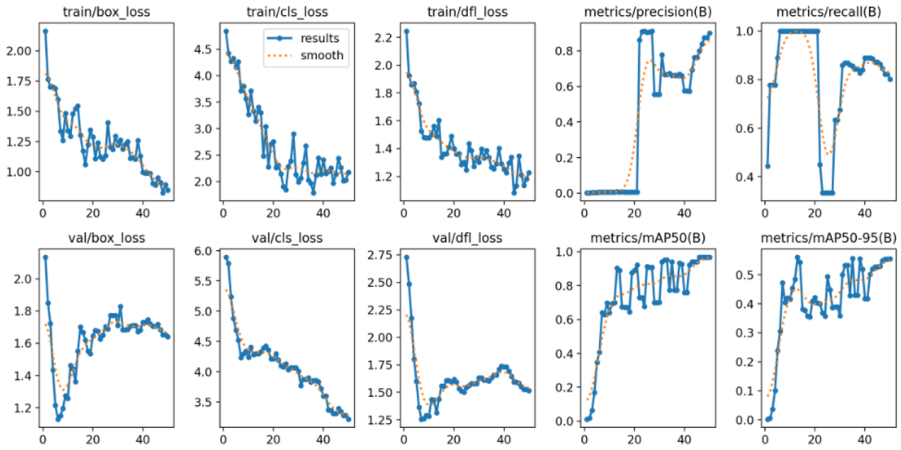

Figure 2: Training loss and accuracy curves.

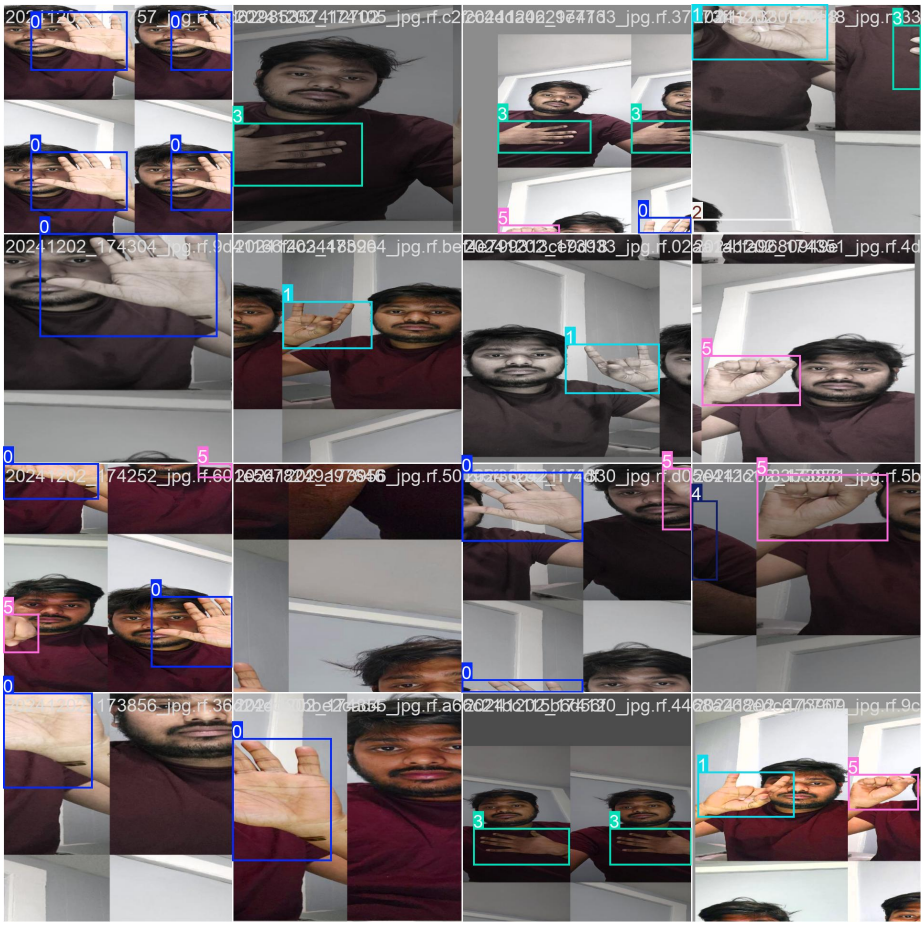

Figure 3: Visualization of model performance.Train batch data.

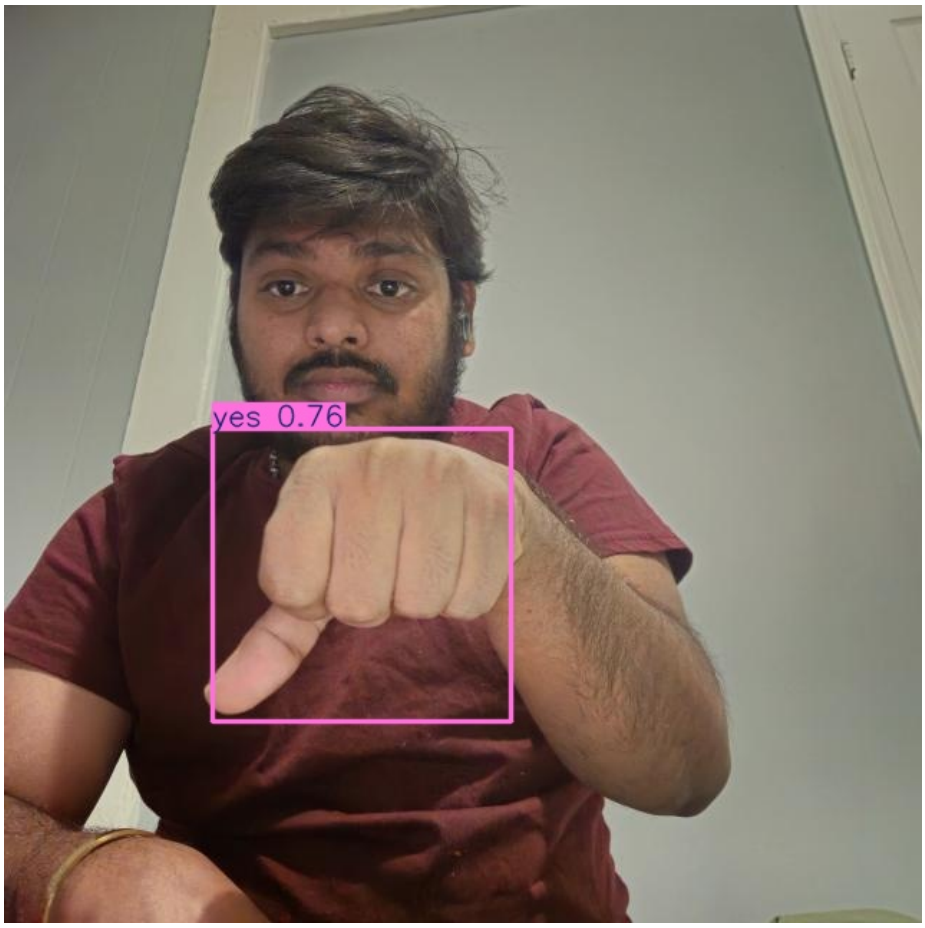

Figure 4: Visualization of model performance.Example sign detection.

4 Analysis and Discussion

The model performs well on distinct gestures but struggles with ambiguous or noisy inputs. Comparative analysis with baseline models indicates a substantial improvement in classification accuracy. Challenges include handling ambiguous gestures and optimizing the system for real-time applications.

5 Conclusion and Future Work

The results of the recognition had a successful way of translating the gestures into speech for recognizing static sign language. Future research will focus on: Continuous Gesture Recognition: Use recurrent neural networks (RNNs) or transformers to understand the timing of movements. Multilingual Support:

Enables the system to understand various sign languages from different regions in a much better way. Edge Device Deployment: Allows the model to work on a low-resource device for real-time use.

6 References

1. Deep Learning for Sign Language Recognition: Current Techniques, Benchmarks, and Open Issues. IEEE Xplore.
2. Unraveling a Decade: A Comprehensive Survey on Isolated Sign Language Recognition. CVPR Open Access.
3. Sarhan, A., et al. Recent Advancements in RGB-Based Sign Language Recognition.
4. ChaLearn Looking at People Challenge 2021 on Isolated Sign Language Recognition.
5. Recent Advances on Deep Learning for Sign Language Recognition. Computer Modeling in Engineering Sciences, 2024. DOI: https://doi.org/10.32604/cmes.2023.045731.
6. Advancements in Sign Language Recognition: A Comprehensive Review and Future Prospects. IEEE Xplore, 2024. DOI: https://ieeexplore.ieee.org/document/10670380.