IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

RECOGNISING AND CONVERTING SIGN LANGUAGE INTO SPEECH

[1]Vaishnavi Athrey, [2]Dr. Deepashree Devaraj
[1]Final Engineering Student, [2]Assistant Professor
[1]Name of Department of 1st Author,
[1]Name of organization of 1st Author, City, Country

Abstract : This paper presents a comprehensive study on recognizing and converting sign language to speech using machine learning models based on Convolutional Neural Networks (CNNs) with a vision-based approach and the OTSU method. Sign language is a vital means of communication for the deaf community, and bridging the communication gap between the deaf and hearing communities is crucial for inclusivity and accessibility. We propose an innovative approach that combines the power of CNNs, a vision-based approach, and the OTSU method for robust and accurate sign language recognition and translation into spoken language.

***Keywords* – Machine Learning models, OTSU methods, Vision- based approach, Convolutional Neural Network, ReLU, Softmax**

I. INTRODUCTION

The challenges faced by the deaf community in communication are significant, as they primarily rely on sign language as their primary means of expression. However, communication barriers arise when interacting with individuals who do not understand sign language. This creates a need for sign language recognition and conversion systems that can facilitate communication between the deaf and hearing communities.

Machine learning models, particularly Convolutional Neural Networks (CNNs), have shown immense potential in recognizing and converting sign language, revolutionizing communication between the deaf and hearing communities. CNNs have emerged as powerful tools for visual recognition tasks, owing to their ability to capture and analyze complex visual patterns. By enabling real-time translation of sign language into spoken language, these systems enhance accessibility, promote inclusion, and empower deaf individuals to fully participate in various aspects of society, ultimately fostering a more inclusive and accessible world.

II. RELATED WORK

Sign language recognition and conversion have been active areas of research, with various vision-based approaches and machine learning (ML) models being explored. This review provides an overview of the existing research in this field, highlighting the advancements and contributions made by different approaches.

1. Vision-Based Approaches:

- Handcrafted Feature Extraction: Earlier research focused on handcrafted feature extraction methods, such as edge detection, contour analysis, and hand shape modeling. These methods relied on predefined rules and heuristics to recognize sign language gestures.
- Template Matching: Some studies employed template matching techniques, where predefined templates of signs were compared with input images or video frames to identify the corresponding sign.

2. Machine Learning Models:

- Support Vector Machines (SVM): SVMs were applied to classify sign language gestures based on extracted features. Feature extraction techniques included edge histograms, shape descriptors, and motion information.
- Deep Learning with CNNs: The advent of deep learning and CNNs revolutionized sign language recognition. CNNs were employed to automatically learn features from sign language images or video frames, achieving significant improvements in recognition accuracy.

The exploration of transfer learning and the creation of diverse and large-scale sign language datasets have shown promise in addressing these challenges. Future research should focus on further improving accuracy, robustness, and real-time performance, thereby contributing to the development of effective sign language recognition and conversion systems.

III. RESEARCH METHODOLOGY

The vision-based approach for hand gesture recognition leverages computer vision techniques to analyze and interpret hand movements and postures in order to recognize and understand sign language gestures. This approach involves a series of steps that collectively enable the system to accurately recognize and classify hand gestures.

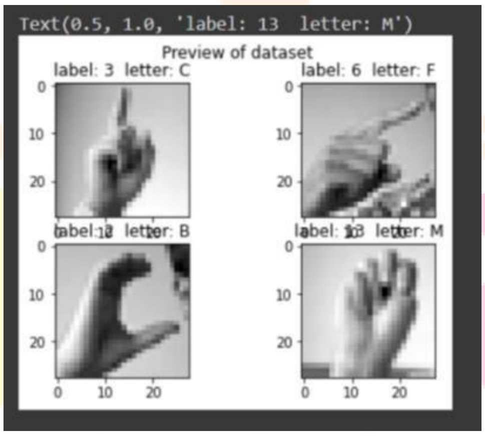
The following sections outline the key components and processes involved in the vision-based approach:

1. Image Acquisition: The first step is to acquire images or video frames that capture the hand gestures. This can be done using various devices, such as cameras or depth sensors, ensuring proper lighting and positioning for optimal image quality.

2. Preprocessing: Preprocessing techniques are applied to the acquired images or frames to enhance the visibility of hand gestures and eliminate noise or unwanted artifacts. These techniques may include contrast enhancement, noise removal, and image normalization, as discussed earlier.

3. Hand Detection and Localization: Hand detection algorithms are employed to identify and localize the hand region within the acquired images or frames. These algorithms utilize various techniques such as skin color segmentation, edge detection, or machine learning-based methods to isolate the hand from the background or other objects.

4. Hand Feature Extraction: Once the hand region is localized, relevant features are extracted to capture the distinctive characteristics of hand gestures. These features can include hand shape, finger positions, contours, or motion information. Feature extraction techniques can range from simple methods such as pixel intensity or shape descriptors to more complex methods like scale-invariant feature transform (SIFT) or histogram of oriented gradients (HOG).



```
Text(0.5, 1.0, 'label: 13  letter: M')
```

Fig 1. Feature Extraction of hand gestures

One of the methods used in this approach is known as the OTSU method. The OTSU method, also known as Otsu's thresholding, is a widely used image segmentation technique that automatically determines an optimal threshold value to separate objects from the background in grayscale images. Developed by Nobuyuki Otsu in 1979, this method has found significant applications in various image processing tasks, including sign language recognition and conversion.

The OTSU method operates on the assumption that the grayscale histogram of an image can be divided into two classes: foreground (objects) and background. It aims to find the optimal threshold that maximizes the between- class variance while minimizing the within-class variance. By identifying the optimal threshold, the method effectively separates the objects from the background, enabling precise and accurate segmentation.

The algorithm of the OTSU method can be summarized as follows:

1. Compute the histogram of the grayscale image, representing the distribution of pixel intensities.
2. Calculate the cumulative sum of the histogram and obtain the total sum of intensities.
3. Iterate over all possible threshold values from 0 to the maximum intensity level.
4. Compute the between-class variance for each threshold value, considering the intensities on both sides of the threshold.
5. Select the threshold that maximizes the between-class variance, resulting in the optimal threshold value.

By automatically determining an optimal threshold, it enables accurate separation of objects from the background, improving subsequent analysis and facilitating precise interpretation of sign language gestures.

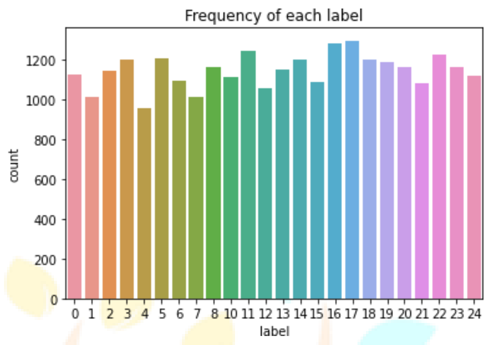


Fig 2. Thresholding using OTSU Method

IV. DATASET COLLECTION AND PREPROCESSING

Collecting a diverse and representative sign language dataset is crucial for training and evaluating machine learning models in sign language recognition and conversion. Our dataset is a well-curated dataset that encompasses variations in gestures, signers, and lighting conditions to ensure robust and generalizable model performance.

American Sign Language (ASL) is a complete, natural language that has the same linguistic properties as spoken languages, with grammar that differs from English. ASL is expressed by the movement of hands and faces. This dataset consists of 27,455 images of handsigns, each image is of 28 x 28 size and in grayscale format. The dataset format is patterned to match closelywith the classic MNIST. Images in the dataset belong toa label from 0–25 representing letters from A-Z (but nocases of 9=J or 25=Z as they involve hand motion).

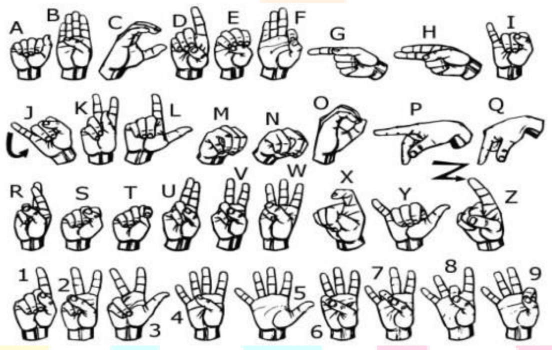


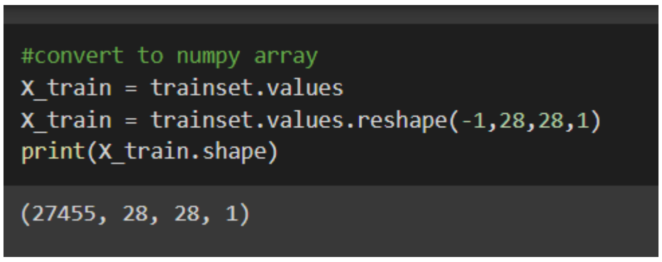
Fig 3. ASL Dataset used

Dataset Preprocessing:

Preprocessing plays a crucial role in preparing the sign language dataset for effective training and analysis. Thissection outlines the key preprocessing steps, including image enhancement, noise removal, and normalization techniques, applied to the dataset.

1. Image Enhancement:

- Reshaping Image: We reshaped the image to improve the visibility of sign language gestures. As theimages input into the model must be of a 28x28 size, allcaptured frames must be reshaped.

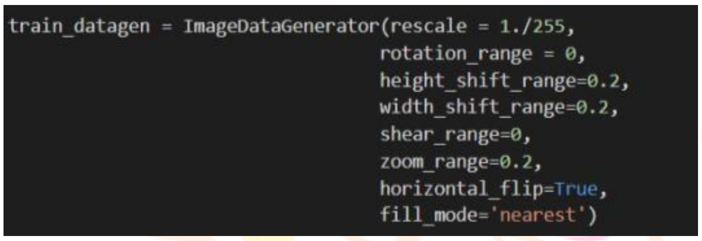


```
#convert to numpy array
X_train = trainset.values
X_train = trainset.values.reshape(-1,28,28,1)
print(X_train.shape)

(27455, 28, 28, 1)
```

Fig 4. Reshaping of data

- Scaling Image: For the neural network, the processing time would reduce if we scaled the data from0-255 to 0-1. Hence, we applied scaling techniques to enhance the edge details and scale it appropriately.



```
train_datagen = ImageDataGenerator(rescale = 1./255,
                                   rotation_range = 0,
                                   height_shift_range=0.2,
                                   width_shift_range=0.2,
                                   shear_range=0,
                                   zoom_range=0.2,
                                   horizontal_flip=True,
                                   fill_mode='nearest')
```

Fig 5. Scaling of data

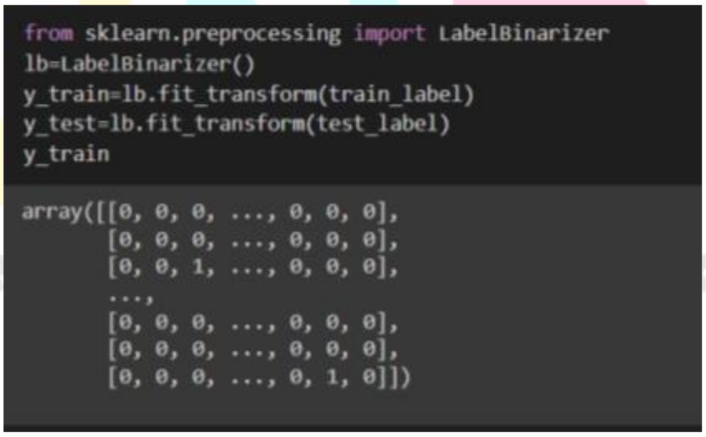
2. Noise Removal:

- Denoising Filters: We applied spatial filters, such asGaussian or median filters, to remove noise and smooththe images while preserving important details.

- Background Subtraction: If the background wasrelatively static, we employed background subtraction techniques to eliminate the stationary elements from theimages, emphasizing only the moving hand gestures.

3. Label Binarizer:

Label Binarizer is a class that accepts categorical data as input and returns a numpy array to ensure faster and easier image processing.



```
from sklearn.preprocessing import LabelBinarizer
lb=LabelBinarizer()
y_train=lb.fit_transform(train_label)
y_test=lb.fit_transform(test_label)
y_train

array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 1, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 1, 0]])
```

Fig 6. Applying Label Binarizer to the data

V. CNN ARCHITECHTURE FOR FEATURE EXTRACTION

The chosen Convolutional Neural Network (CNN) architecture plays a crucial role in extracting relevant features from sign language images and enabling accurate recognition. This section provides a detailed explanation of the selected CNN architecture, highlighting its different layers, parameters, and network design.

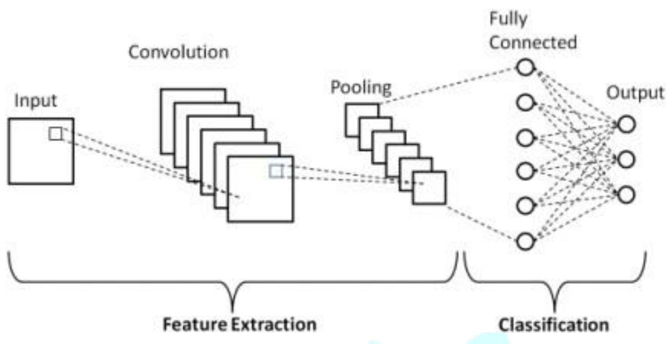


Fig 7. Convolutional Neural Network layers

1. Convolutional Layers

The CNN begins with a series of convolutional layers that extract features from the input sign language images. Each convolutional layer consists of multiple filters or kernels that perform convolutions on the inputimage, capturing spatial patterns and local features.

The number of filters and their sizes are key parametersthat determine the complexity and expressive power ofthe CNN. Convolutional layers are typically followed by activation functions, such as ReLU (Rectified LinearUnit), to introduce non-linearity.

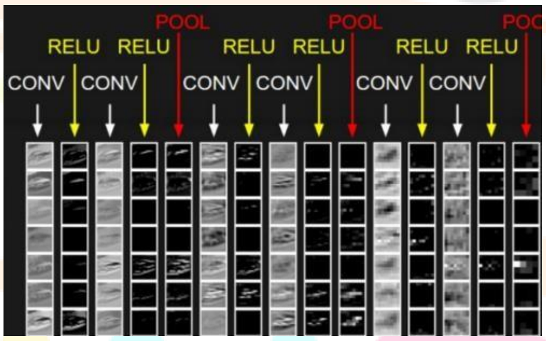


Fig 8. Usage of ReLU

2. Pooling Layers:

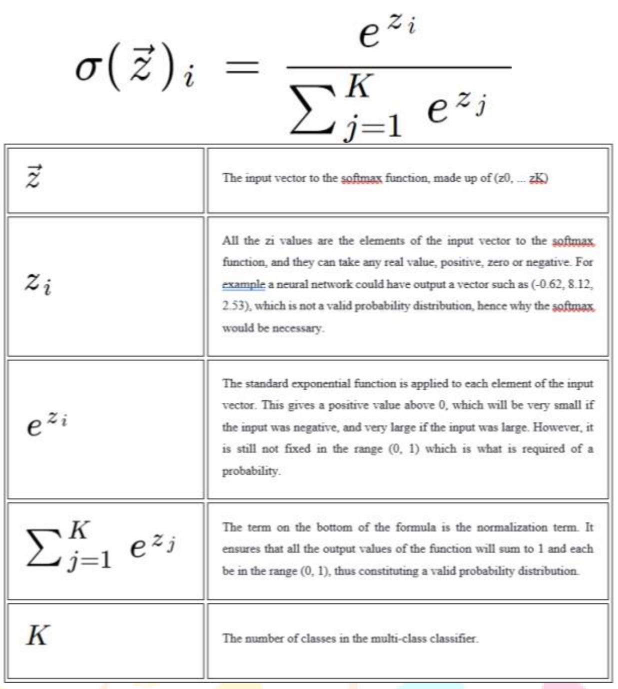
Pooling layers are inserted between the convolutional layers to reduce the spatial dimensions of the feature maps while preserving the important features. Pooling helps in down sampling the feature maps, making the network more robust to variations in hand gesture positions and allowing higher-level feature extraction.

3. Fully Connected Layers:

Following the convolutional and pooling layers, fully connected layers are employed to process the extracted features and make predictions. Fully connected layers are composed of nodes or neurons, where each neuron is connected to all the neurons in the previous layer. The number of neurons in the fully connected layers and their arrangement are important architectural parameters that determine the model's capacity to learn complex relationships.

4. Output Layer:

The final layer of the CNN is the output layer, which produces the predicted classes or labels for the sign language gestures. The number of neurons in the outputlayer corresponds to the number of classes or gestures to be recognized. Activation functions such as softmax are often used in the output layer to generate class probabilities and facilitate multi-class classification.



$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

$\vec{z}$	The input vector to the softmax function, made up of (z0, ... zK)
z_i	All the zi values are the elements of the input vector to the softmax function, and they can take any real value, positive, zero or negative. For example a neural network could have output a vector such as (-0.62, 8.12, 2.53), which is not a valid probability distribution, hence why the softmax would be necessary.
e^{z_i}	The standard exponential function is applied to each element of the input vector. This gives a positive value above 0, which will be very small if the input was negative, and very large if the input was large. However, it is still not fixed in the range (0, 1) which is what is required of a probability.
$\sum_{j=1}^{K} e^{z_j}$	The term on the bottom of the formula is the normalization term. It ensures that all the output values of the function will sum to 1 and each be in the range (0, 1), thus constituting a valid probability distribution.
K	The number of classes in the multi-class classifier.

Fig 9. Softmax Formula

The chosen Convolutional Neural Network (CNN) architecture is justified based on its ability to effectivelycapture relevant visual patterns and hierarchical featuresin sign language gestures. This justification lies in the inherent strengths of CNNs in analyzing images and their suitability for complex pattern recognition tasks.

VI. PROCEDURE

The following enlists the detailed procedure of the development of a Sign Language to Speech converting system:

1. Labeled Sign Language Datasets:

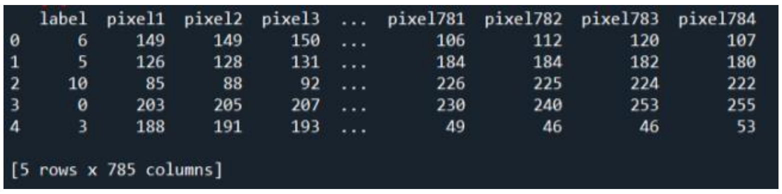
To train the ML model, a diverse and representative dataset of sign language gestures is required, as discussed earlier. The dataset includes a wide range of gestures performed by different signers, covering variations in hand shapes, movements, and expressions. Each sample in the dataset is labeled with thecorresponding sign language gesture, providing the ground truth for training the model.

2. Feature Extraction:

Before training the ML model, the sign language imagesor frames need to be preprocessed and transformed intosuitable feature representations. This typically involvestechniques such as image enhancement, noise removal,and normalization, as discussed earlier. The goal is to extract relevant visual features that capture the distinctive characteristics.

3. Model Training:

The ML model, based on the selected CNN architecture,is trained using the labeled sign language dataset. During training, the model learns to map the input signlanguage images to their corresponding labels through an iterative optimization process. The objective is to minimize the discrepancy between the predicted outputsand the ground truth labels, using loss functions such ascategorical cross-entropy.



```
   label  pixel1  pixel2  pixel3  ...  pixel781  pixel782  pixel783  pixel784
0      6     149     149     150  ...       106       112       120       107
1      5     126     128     131  ...       184       184       182       180
2     10      85      88      92  ...       226       225       224       222
3      0     203     205     207  ...       230       240       253       255
4      3     188     191     193  ...        49        46        46        53

[5 rows x 785 columns]
```

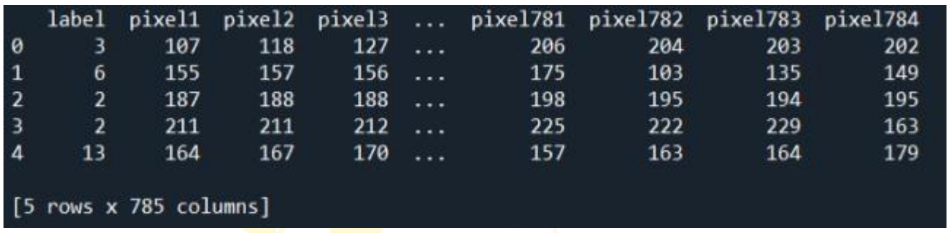
Fig 10. Train Set

4. Hyperparameter Selection:

Hyperparameters play a critical role in the trainingprocess and influence the performance of the ML model. These hyperparameters include learning rate, batch size, number of epochs, weight initialization schemes, regularization techniques, and optimizerselection. The selection and tuning of hyperparameters are performed through experiments, cross-validation, and validation set performance evaluation.

5. Validation and Evaluation:

To assess the performance of the trained ML model, a separate validation dataset is used. The validation dataset, distinct from the training dataset, helps in estimating the model's generalization ability and identifying potential overfitting.



```
   label  pixel1  pixel2  pixel3  ...  pixel781  pixel782  pixel783  pixel784
0      3     107     118     127  ...       206       204       203       202
1      6     155     157     156  ...       175       103       135       149
2      2     187     188     188  ...       198       195       194       195
3      2     211     211     212  ...       225       222       229       163
4     13     164     167     170  ...       157       163       164       179

[5 rows x 785 columns]
```

Fig 11. Test Set

By utilizing a supervised learning approach, the ML model learns to recognize and convert sign language gestures based on the labeled training data. It leveragesthe provided annotations to associate visual patterns with specific sign language labels. The model's ability to generalize unseen sign language gestures is enhancedby the diversity and representativeness of the training dataset. The iterative training process and hyperparameter selection aim to optimize the model's performance and achieve high accuracy in gesture recognition and conversion.

VII. RESULTS AND DISCUSSION

Evaluation metrics play a crucial role in assessing the performance of a sign language recognition andconversion system. They provide quantitative measuresof the system's accuracy, robustness, and effectiveness in understanding and converting sign language gestures.In this particular paper, the results are provided in the form of accuracy and precision.

1. Accuracy:

Accuracy is a fundamental metric that measures the overall correctness of the system's predictions. It iscalculated as the ratio of correctly recognized gestures to the total number of gestures in the dataset. Theaccuracy obtained at the end of this project was 99.6%.

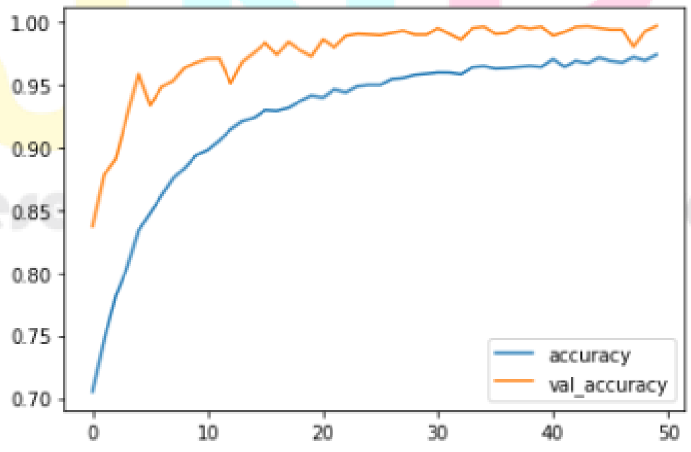


Fig 12. Accuracy Graph

2. Precision and Loss:

Precision and loss measure the proportion of correctly recognized positive gestures out of all predicted positivegestures. The loss encountered was 0.92.

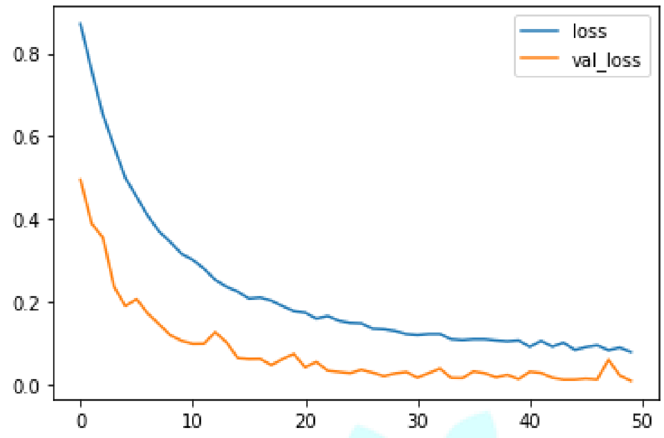


Fig 13. Loss Graph

VIII. FUTURE WORK

The field of sign language recognition and conversion using CNNs and the vision-based approach holds great potential for further advancements. Here are some potential enhancements and future directions to improvethe effectiveness and capabilities of the system:

1. Transfer Learning and Pretrained Models: Leveraging transfer learning techniques by utilizingpretrained CNN models trained on large-scale image datasets, such as ImageNet, can enhance the recognitionperformance. Fine-tuning these pretrained models on the sign language dataset can help improve accuracy, especially with limited training data.

2. Temporal Modeling: Considering the temporal aspectof sign language gestures can capture the dynamics andmotion information. Employing recurrent neural networks (RNNs), long short-term memory (LSTM) networks, or temporal convolutional networks (TCNs) can model the temporal dependencies and improve the system's ability to recognize complex gestures.

3. Integration of Natural Language Processing (NLP): Combining sign language recognition with natural language processing techniques can enable the system to convert sign language gestures directly into spoken language or textual representation. This integration canfacilitate more effective communication between sign language users and non-signers.

By focusing on these potential enhancements and futuredirections, researchers and developers can advance the field of sign language recognition and conversion usingCNNs and the vision-based approach.

IX. CONCLUSION

The paper aimed to address the challenge of recognizingand converting sign language gestures into speech usingmachine learning models based on convolutional neuralnetworks (CNNs) and a vision-based approach. The integration of the OTSU method and the detailed explanation of the CNN architecture and training procedure offer valuable guidance for researchers and developers working on similar systems. The performance evaluation and comparison with existing approaches validate the effectiveness of the proposed approach. The paper's findings lay the foundation for further advancements in the field, promoting more accurate, real-time, and inclusive sign language recognition and conversion systems.

X. REFERENCES

1. H. Li et al., "Real-Time American Sign Language Recognition Using Convolutional Neural Networks," inIEEE Transactions on Human-Machine Systems, vol. 47, no. 1, pp. 114-124, 2017.

2. A. Tripathi and R. Chellappa, "Sign Language Recognition using Dynamic Time Warping and Convolutional Neural Networks," in 2018 IEEE WinterConference on Applications of Computer Vision(WACV), 2018, pp. 1351-1360.

3. M. K. Singh and V. P. Namboodiri, "Sign LanguageRecognition using Temporal Residual Networks," in 2019 IEEE International Conference on Image Processing (ICIP), 2019, pp. 1790-1794.

4. M. Huenerfauth et al., "Automated Animation of American Sign Language Facial Expressions: A Surveyof the State of the Art," in IEEE Transactions on Human-Machine Systems, vol. 49, no. 5, pp. 424-434, 2019.

5. M. Siddiqui et al., "Sign Language Recognition usingDeep Learning Techniques: A Comprehensive Review," in 2021 IEEE International Symposium on

6. S. Ganorkar et al., "Real-Time Sign Language Recognition using Convolutional Neural Network and Long Short-Term Memory," in 2020 IEEE InternationalConference on Consumer Electronics (ICCE), 2020, pp.1-6.

7. A. Bansal et al., "Sign Language Recognition using Deep Learning: A Review," in 2019 IEEE 5th International Conference on Recent Advances in Information Technology (RAIT), 2019, pp. 1-5.

8. R. Roy et al., "Sign Language Recognition using Deep Learning Techniques: A Survey," in 2020 IEEE International Conference on Recent Advances in Electronics and Communication Technology (ICRAECT), 2020, pp. 195-200.

9. S. Kumar et al., "Real-Time Indian Sign Language Recognition using Convolutional Neural Network," in 2019 IEEE 5th International Conference for Convergence in Technology (I2CT), 2019, pp. 1-6.

10. P. Kumar et al., "Automatic Indian Sign Language Recognition using Convolutional Neural Network," in 2017 IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics Engineering (UPCON), 2017, pp. 1-5.