FINAL PROJECT REPORT

SIGN LANGUAGE DETECTION

Akshaya Mamidipalli

811292765

Abstract:

To set those who are deaf or mute apart from the rest of society, deafness and mutism frequently serve to highlight these individuals. With different motions denoting a particular letter, number, or sentence, most of the communication takes place in either Indian Sign Language (ISL) or American Sign Language (ASL). While some deaf persons use hearing aids, few can afford them, and not all use sign language. All these issues are resolved by sign language since it is more effective, simple, and flexible. The goal of the project is to create a hand motion recognition system that communicates the importance of hope for the entire population. The research focuses on creating a hand gesture recognition system that communicates the message of hope for the entire society and its relevance since it is more effective, simple, and flexible. The goal of this work is to identify movements of signs using deep learning and to enhance current systems to provide software that is both remarkable and efficient.

Introduction:

Gestures, which use body language or hand movements in sign language, are the most well-known and comprehended form of communication. The deaf and mute communities employ sign language, a type of gesture communication that allows people to express themselves and shines a light on the darkness. However, one needs to understand the signs and gestures employed to convey these viewpoints in order to appreciate them. People in today's world are so preoccupied with themselves that they rarely stop to consider other people. The deaf and mute groups are left to bear the consequences in these situations. Recent advances in artificial intelligence have led to countless gains in several fields, such as robotics and healthcare, which motivate us to work toward a better society. HCI, or human-computer interaction, is one such field of study. Facial recognition technology and application fields are the focus of HCI. In order to pave the way for more significant discoveries, the system's picture analysis aims to assist and train the CNN (Convolutional Neural Network) to decipher the indicators.

It is frequently stressed that communication is the key to all opinions since it reflects a person's basic principles of communicating ideas, thoughts, and viewpoints. Nonetheless, a significant portion of the world's population has speech problems, hearing loss, or both. By providing the deaf-mute population with an optimistic environment, this project seeks to close the communication gap between us and them. Converting the received data images into grayscale format, appropriately processing them, and guaranteeing that the images gathered are faithful to the Sign Language are some of the system's primary features. The Deep Learning Neural Network recognition system is the central concept.

Background:

To help people with hearing problems communicate, sign language is essential. However, accuracy and real-time processing are two issues that plague many of the sign language recognition methods now in use. To improve the effectiveness and dependability of sign-language identification systems, this study uses deep learning neural networks.

Motivation:

The project's goal is to help deaf and dumb people communicate better. This allows a machine to predict the hand gestures of people who are deaf or stupid. Here, we are developing a model that predicts the results of different hand gestures. To do this, we will process photos and use a deep learning method to obtain training datasets. Every field is greatly impacted by communication, and the way that people interpret thoughts and expressions is what draws researchers to close this gap for all living things.

Related Works:

Zafar Ahmed Ansari and Gaurav Harit carried out an important work to properly classify Sign Language gestures. Finger-finger-finger spellings of alphabets, integers, and other frequently used terms are among the 140 categories into which they have split the globe. The dataset is created using a Kinect sensor. 640 × 480-pixel RGB photos are gathered, along with their depth information. Each pixel's depth values are recorded by the image's depth data. Every pixel in the image has a corresponding value in the depth data file. Since all of the sample members were standing with their hands outstretched, the hand in the picture will have the least depth. We verified this by masking pixels with the same depth values as the rest of the image.

Divya Deora and Nikesh Bajaj employ Principal Component Analysis (PCA) to identify sign symbols. The article also suggests using neural networks for recognition. The quality was low because the data was taken with a 3-megapixel camera. For every sign, they gathered 15 images, which they then stored in their database. The small dataset was one of the causes of their disappointing findings. They segmented and divided the RGB into components and performed a basic border pixel analysis. They assert that neural networks can produce a superior result, even though using the fingertip approach in conjunction with PCA produced a favourable result.

To identify sign language, Lionel et al. employed a Convolutional Neural Network (CNN). To automate sign language recognition, two processes were completed: action categorization and feature extraction. An Artificial Neural Network is utilized to finish the second phase, whereas a CNN is employed to finish the first. Twenty distinct Italian gestures were signed by 27 participants in the sample under a variety of circumstances. Video recordings are made with the Microsoft Kinect. 6600 images were used in the development process, of which 4600 were used for training and the rest for validation. In post-production, the upper portion of the hand and body was cropped out of the photos. For each gesture, the model just needs to learn one side. Max pooling is used to extract and classify features. Each of the two CNNs that make up this system feeds its output into an ANN. The ANN makes use of the output from both CNNs. A GPU was used to train the model, while a CPU was used to augment the data. Data distortion encompassed transformation, rotation, and zooming.

Challenges:

Data problems and the intricacies of sign language interpretation are just two of the challenges that come with developing a robust deep learning system for sign language identification.
a) Limited datasets: It can be challenging to compile a comprehensive and diverse data set of

sign language gestures because of regional differences, variations in signing techniques, and the need for inclusivity.
b) Diverse signing styles: It is challenging to create a one-size-fits-all strategy that works for all sign language users due to their diverse signing styles and variances.
C) Real-time processing: Although it can be computationally demanding and lead to latency issues, real-time processing is crucial for efficient communication in sign language detecting systems.
d) Limited availability: A lot of time and resources can be spent annotating large datasets for deep learning model training, particularly when sign language datasets are involved.
E) User flexibility: Users may have unique expressions and signing styles, which makes it challenging to create a system that adapts to each user's unique characteristics.

Technical Approach:

Overview:

TensorFlow and Keras are used in the code to create a Convolutional Neural Network (CNN) for picture classification. It includes training, performance evaluation, model architecture, data preparation, data augmentation, and library imports. The model is made up of dense layers, convolutional layers with batch normalization, max pooling, and dropout for regularization. Along with image and label visualization, the system also has a confusion matrix for performance analysis on specific classes. Along with establishing parameters like batch size, image dimensions, and epochs, the code also mounts Google Drive in Google Colab for file access. The script offers information on how the model behaves and performs.

Steps Involved:

Importing Libraries:

Bring in the required Python libraries, such as Matplotlib for data visualization, Pandas for data manipulation, NumPy for numerical calculations, and TensorFlow and Keras for machine learning.

Preparing Data

- Use image_dataset_from_directory in TensorFlow to load and preprocess image data.
- Divide the data into sets for training and validation.
- Show the names of the classes together with the number of photos in each class.

Data Augmentation:

- Using Keras' ImageDataGenerator, define a function (func) for picture augmentation.
- Adjust the training pictures and store them in the designated directory.

Model Architecture:

- Convolutional layers with batch normalization, max pooling, dropout for regularization, and dense layers make up the model. It is constructed using the Keras Sequential API.
- There are two distinct model iterations with varying dropout rates.

Model Training and Compilation

- Use accuracy as the measure, sparse categorical cross-entropy loss, and the Adam optimizer to compile the model.
- Use the training set to train the model, the validation set to validate it, and the history object to track the training process. Show the accuracy and loss of training and validation throughout epochs.

Assessment of Performance:

- Assess the final model's accuracy using the validation set.
- Show the elapsed time after training the model for more epochs.

Matrix of Confusion:

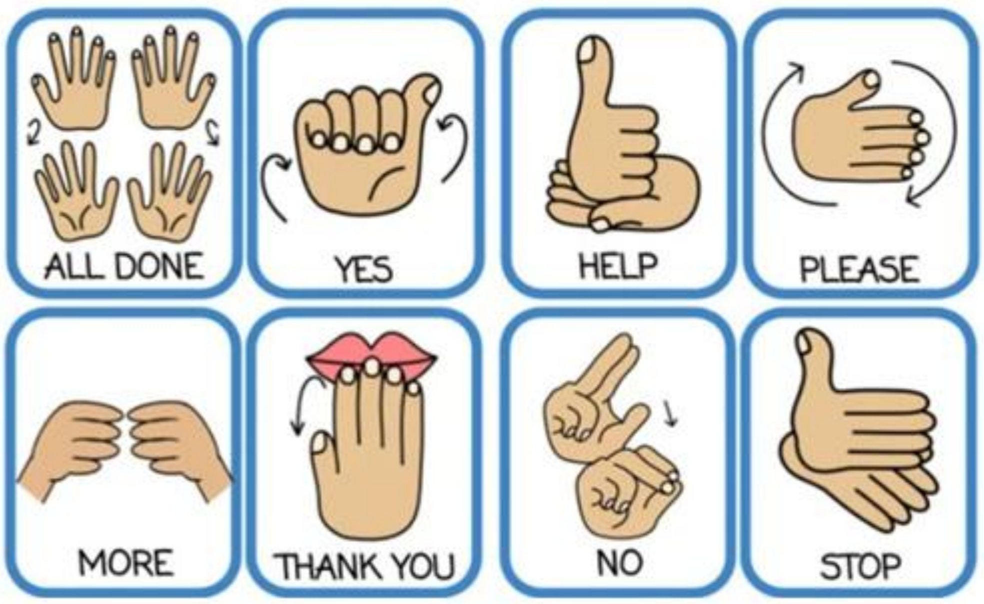
- Create and present a confusion matrix to examine how well the model performs in each class. Visualization of Images and Labels.
- Show the predicted and true labels for a sample of the validation set's photos.

All things considered, the code complies with accepted procedures for creating and refining a CNN for picture categorization. Model versions with varying dropout rates for regularization, visuals for performance evaluation, and data augmentation to improve model generalization are all included. The well-organized script seeks to shed light on the performance and behaviour of the model.

Experiments:

Dataset:

- The dataset we utilized contained 500 validation samples and 1000 training samples for every indicator.
- Experimented with different model setups, such as filters, epochs, and dropout rates.
- Displayed validation accuracy, validation loss, training accuracy, and training loss.



Evaluation Metric:

We are getting the following validation accuracy using the Keras API for convolutional neural networks (CNNs) with TensorFlow as the backend.

- 100 percent following 25 epochs, a 0.25 dropout, batch normalization, and 2D max pooling for spatial sampling.
- When we raise the dropout to 0.5 and decrease the epoch to 5, the validation accuracy is 97.15%.

No. of Epoch's	Dropout	Validation accuracy
25	0.25	100%
25	0.5	100%
10	0.5	100%
5	0.5	97.15%
7	0.6	86.75%
10	0.6	94.44%

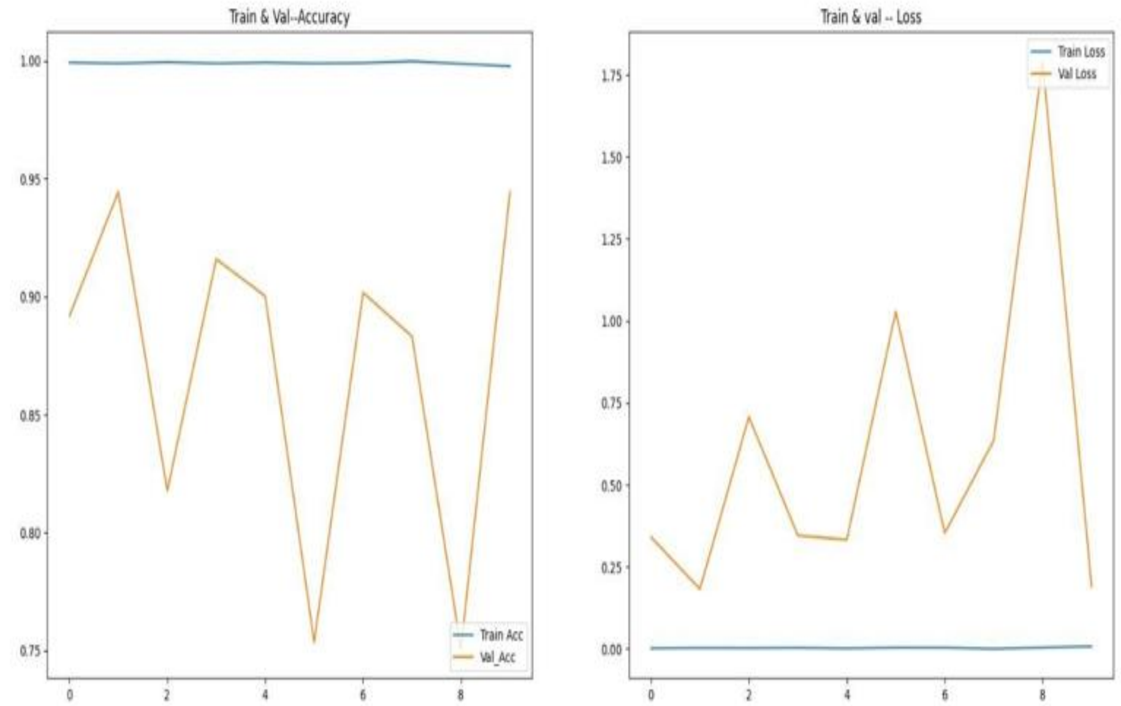
Quantitative Results:

The code's quantitative output includes training and validation accuracy, training and validation loss, post-training validation accuracy, and confusion matrix insights. These findings provide a thorough understanding of the model's functionality and can direct more research and improvement of the sign language identification system.

Qualitative Results:

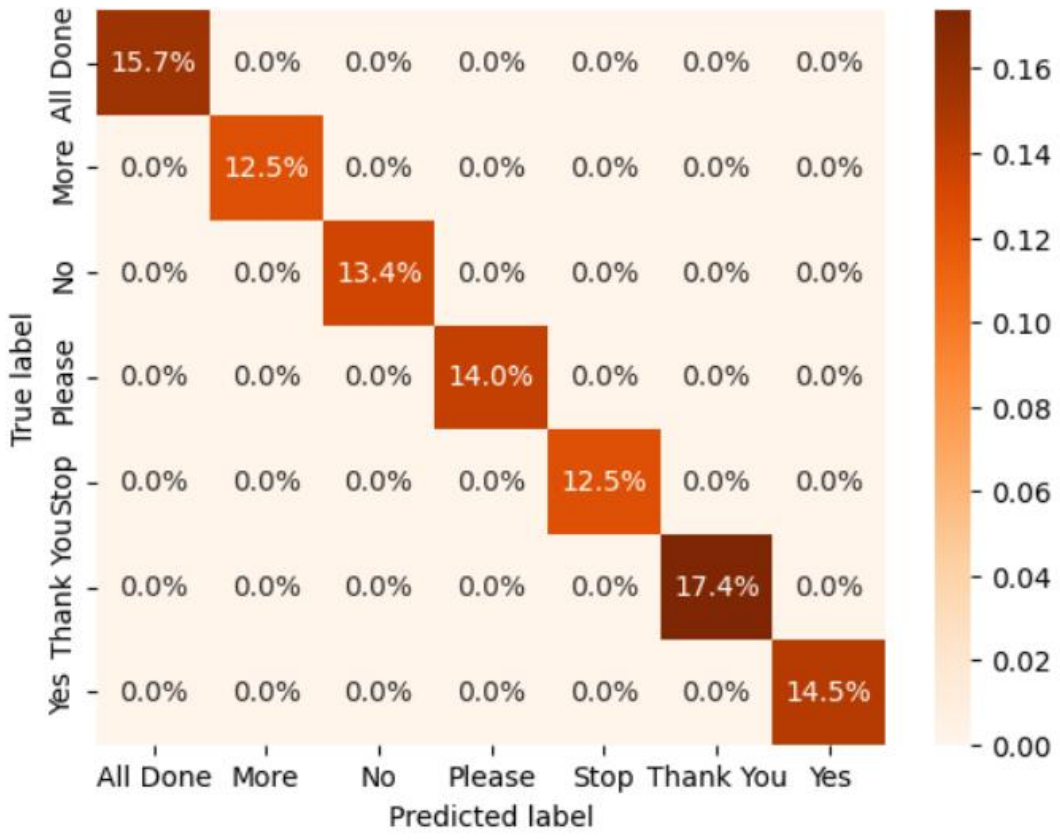
Based on typical situations, this qualitative analysis offers a narrative about the model's advantages, disadvantages, and possible areas for development. The particulars of the dataset and the model's training would determine the actual outcomes.

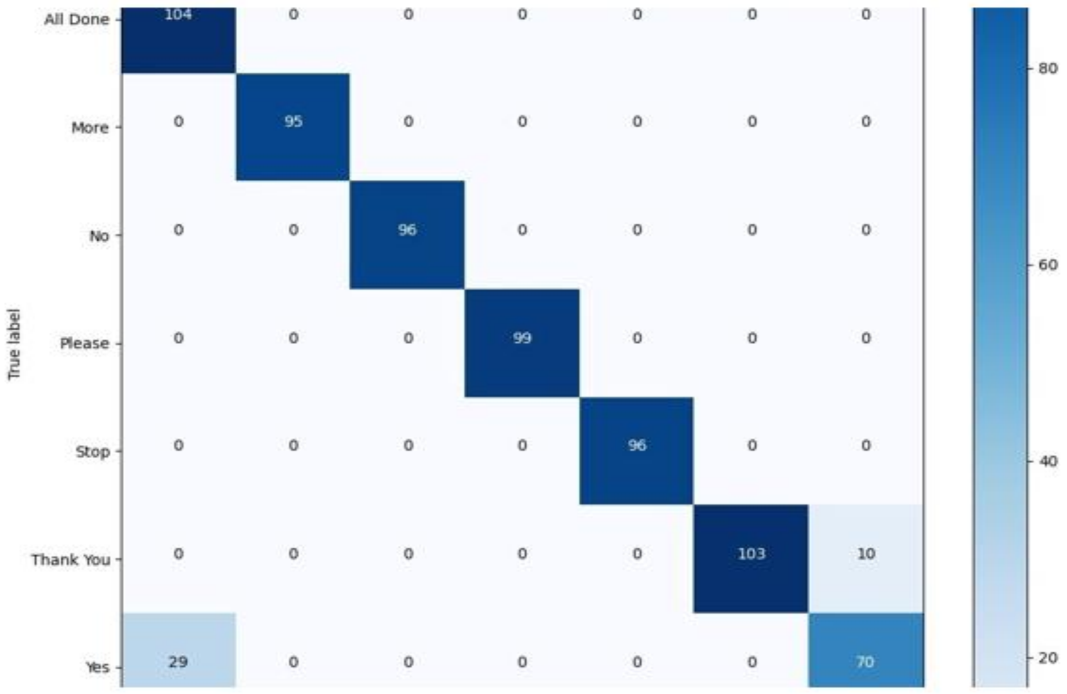
Yes
Yes
More
All Done
More
Please
All Done
Stop
No


Train & Val--Accuracy
Train Acc
Val_Acc
Train & val -- Loss
Train Loss
Val Loss

Confusion Matrix:
When describing how well a classification model performs on a collection of data for which the true values are known, a confusion matrix is a table that is frequently utilized. A predicted class's instances are represented by each row of the matrix, and an actual class's occurrences are represented by each column (or vice versa). If you want to examine how well a classification model performs in terms of true positive, true negative, false positive, and false negative classifications, the confusion matrix is very helpful.

We used a confusion matrix in our project to determine the relationship between each sign, and the only ones that have a relationship with "yes" are "all done" and "thank you."





Conclusion:
TensorFlow and Keras were used in the research to create and train a Convolutional Neural Network (CNN) for sign language recognition. The code implementation included a number of topics, including data preprocessing, model architecture, training, and assessment. The model was created to classify hand signs from photos. Among the major accomplishments were convolutional layers, data augmentation, and data preprocessing using TensorFlow's image_dataset_from_directory. The model showed that it could use convolutional blocks to learn hierarchical features from input photos.

Sparse categorical cross-entropy loss and the Adam optimizer were used for training and assessment. Time analysis documented the execution time for various steps, while performance analysis demonstrated the model's capacity to categorize various sign language classes. Real-world deployment, transfer learning, and hyperparameter tuning are future considerations.

In summary, the created sign language identification model exhibits encouraging outcomes, and more development and investigation of cutting-edge methods can improve precision and generalization. This initiative advances the creation of accessible and inclusive sign language recognition systems.

References:

K. Bantupalli and Y. Xie, "American Sign Language Recognition using Deep Learning and Computer Vision," *2018 IEEE International Conference on Big Data (Big Data)*, Seattle, WA, USA, 2018, pp. 4896-4899, doi: 10.1109/BigData.2018.8622141.

A. Kumar, K. Thankachan and M. M. Dominic, "Sign language recognition," *2016 3rd International Conference on Recent Advances in Information Technology (RAIT)*, Dhanbad, India, 2016, pp. 422-428, doi: 10.1109/RAIT.2016.7507939.

S. Ikram and N. Dhanda, "American Sign Language Recognition using Convolutional Neural Network," *2021 IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCON)*, Kuala Lumpur, Malaysia, 2021, pp. 1-12, doi: 10.1109/GUCON50781.2021.9573782

M. N. Saiful *et al*., "Real-Time Sign Language Detection Using CNN," *2022 International Conference on Data Analytics for Business and Industry (ICDABI)*, Sakhir, Bahrain, 2022, pp. 697-701, doi: 10.1109/ICDABI56818.2022.10041711.

A. Deshpande, A. Shriwas, V. Deshmukh and S. Kale, "Sign Language Recognition System using CNN," *2023 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE)*, Bengaluru, India, 2023, pp. 906-911, doi: 10.1109/IITCEE57236.2023.10091051.

Aishwarya, Aparna and D. J. D’Souza, "Sign Language Recognition," *2021 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)*, Nitte, India, 2021, pp. 104-106, doi: 10.1109/DISCOVER52564.2021.9663629.