Volume 1, No. 1, Issue 2

Sign-To-Text Conversion

Ayush Kumar
Computer Science Engineering Department, Chandigarh University, Mohali, Punjab. Email: ayushkumarak560@gmail.com

Akanksha Singh
Computer Science Engineering Department, Chandigarh University, Mohali, Punjab. Email: akanksha.e14998@cumail.com

Piyush Verma
Computer Science Engineering Department, Chandigarh University, Mohali, Punjab. Email:pv535111@gmail.com

Abstract— People who are born deaf or hard of hearing are viewed as liabilities by the community and find it extremely difficult to support themselves. They need specialized training in order to communicate using sign language, but the issue still exists since in India, the general public was not provided with enough sign language instruction, making it difficult for them to be understood. a method based on learning that transforms American Sign Language (ASL) into comprehensible text. Convolutional neural networks (CNNs) are used to extract identifying characteristics based on comparisons with datasets. From there, they generate predictions about the alphabet, resulting in word formations that represent the intended message. Using data processing, augmentation, and normalization approaches, the suggested model was trained on this dataset. Using a test dataset, the trained models were assessed and shown to be very accurate in identifying the ASl symbols. Through Sign-To-Text translation, this method offers a quick, accurate, and affordable way to translate American Sign Language into Text. This strategy may enhance the manner in which an individual and those with speech impairments communicate**.**

***Index Terms*- CNN, deep learning. Pooling, max pooling, TensorFlow**

INTRODUCTION

The process by which individuals exchange their feelings, ideas, and opinions about a situation or about their life is called communication. On the other hand, some people find it more difficult to communicate than average people. For these folks, communicating their ideas and messages requires the use of sign language. To get an idea of how hard it is, consider this easy question: How many individuals can communicate using American Sign Language (ASL) or any other sign language? With that knowledge, youcan see how hard it is for them to interact with regular people. "Sign language is equal to speech, capable of both the rigorous and the poetic, of philosophical analysis and passionate expression." [1] Oliver Sacks is well-known for saying this on the elegance of sign language. The capacity to communicate verbally using gestures and signs is utilized by those who are nonverbal or hard of hearing. Never try to learn sign language so that deaf individuals can communicate with non-deaf people. Deaf people thus experience social isolation. However, if a computer could be made to interpret sign language into text format, the gap between the hearing community andthe broader public may be lessened. In line with a World Health Organization report, 430 million people, or more than 5% of the global population, needed rehabilitation to treat their incapacitating hearing loss. One in ten individuals, or more than 700 million, will have a debilitating hearing loss by the year 2050 [2]. People who identify as "deaf" typically have substantial hearing loss, which means they have little to no hearing. They communicate frequently by using sign language. This "Sign to test conversion" comes into play since we need an inventive, user-friendly method for this sizable segment of the population to communicate with regular people in order to empower them.

Recent developments in machine learning have led to significant progress in the detection and comparison of images, with models demonstrating remarkable accuracy. Gesture-to-Text conversion by the University of Rochester, which uses the OpenCV platform to create an accurate system that can translate gestures and movements into written text, has advanced significantly beyond previous studiesin sign language to text conversion[3]. With an accuracy of 89.5% and a 1.50 SER score utilizing a bidirectional technique, the 3D Avatar technique for Continuous Sign Movement using Speech/Text is a revolutionary AI-based avatar creation method that works exclusively on ISL and predicts the entire word rather than just the alphabets [4]. The self-made KoSign Sign Language Translation Project has a sizable dataset and uses both text and video recognition techniques; nevertheless, its accuracy is now only 75%, and it mayrise in the future as the project is still in development [5]. Following the AR/VR methodology, ASL Recognition for Virtual and Augmented Reality achieves excellent accuracy but is highly dependent on hardware and saves data, raising privacy issues [6]. Another intriguing initiative is How2Sign, which has an enormous amount of original material in the form of voice, motion gestures, films, and more. [7] All of these initiatives demonstrate radically distinct methodologies and aid in the development of this project's concept. As deep learning and neural network research progress, more practical and efficient methods for "sign-to-text conversion" may be created.

The goal of this research paper is to provide a novel method for employing CNNs to translate sign language into text. This effort aims to ensure that the model is resilient and scalable to varied sign language movements, while also attempting to break down the communication barrier between a deaf person and a normal person through a system with accuracy and speed.

Background od Dataset

When creating machine learning models, datasets are crucial because the more images used for comparison, the more accurate the model will be. American Sign Language (ASL), which is extensively used by the deaf and hard-of-hearing community worldwide, is the primary focus of the dataset. It was Nikhil, the dataset author, who took on the chore of compiling and labelling the photographs in signlanguage. [8]

Because the photos were chosen with care, a variety of hand forms, motions, and sign gestures that are often employed in ASL communication are shown. Nikhil worked on ASL (American Sign Language), using the dataset from GitHub and developing his own dataset. Training and testing data are contained in two files inside the dataset. There are 12845 photos in the training set, divided across 27 classes. 4268 photos from 27 courses—26 alphabet classes and one for blank or space—are included in the assessment data.

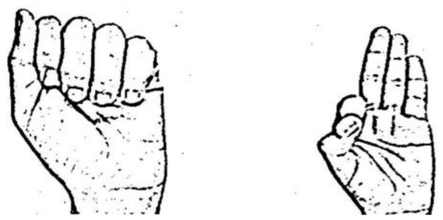
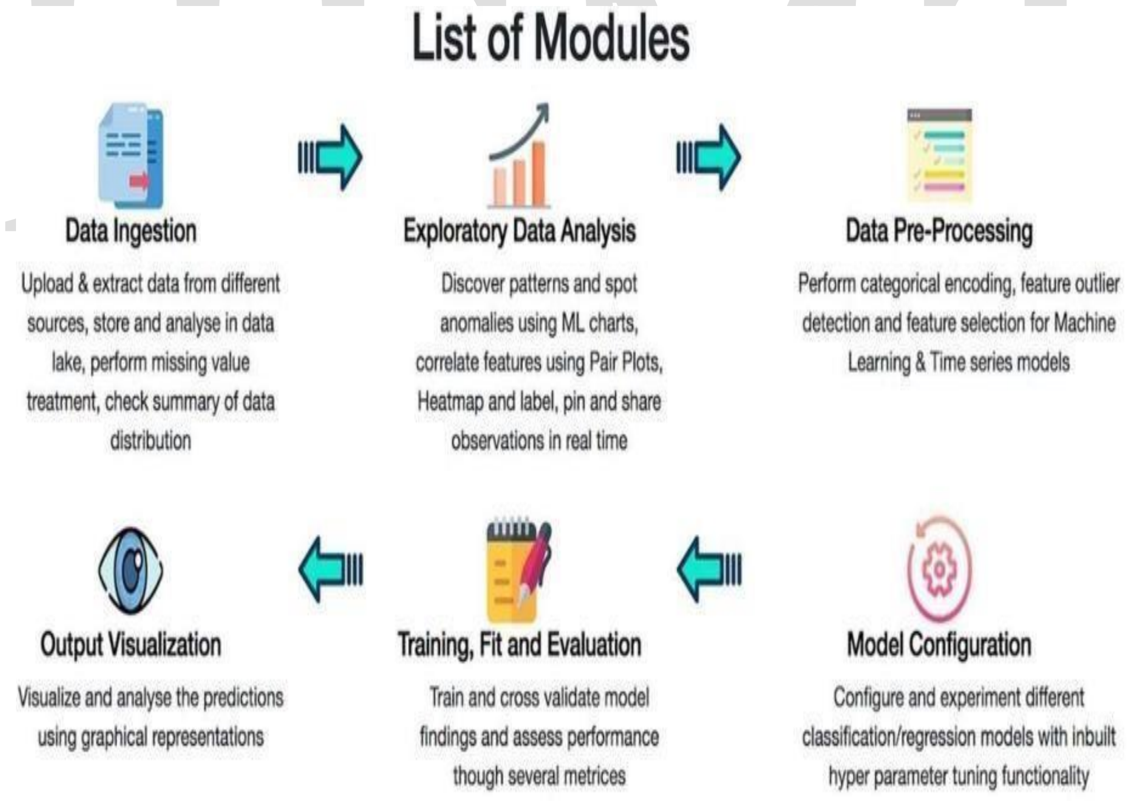


Fig: 1

The collection includes grave-scaled pictures of ASL hand movements, as seen in Fig 1. There are 12,845 photos in all, broken down into 27 groups, in the training data folder. In American Sign Language, each picture corresponds to a certain sign gesture. A variety of hand forms, hand motions, and face emotions are included in these gestures. The dataset offers a thorough depiction of sign language motions and includes a variety of ASL-related topics. 4,268 photos make up the testing data folder, which is arranged into the same 27 classes as the training data. This separation makes sure that the training and testing sets have the same class distribution, which makes it easier to fairly assess and validate the model's performance.

There are 27 classes in the dataset: 26 are alphabet classes and 1 is a class for the blank or space gesture. The alphabet classes allow the recognition and conversion of finger-spelled words and phrases by representing each letter of the English alphabet in ASL. For sign-to- text conversion to handle spaces or pauses in phrases, the blank class must be present. American Sign Language (ASL), which is extensively used by the deaf and hard-of-hearing community worldwide, is the primary focus of the dataset. It was Nikhil, the dataset author, who took on the chore of compiling and labelling the photographs in sign language. Because the photos were chosen with care, a variety of hand forms, motions, and sign gestures that are often employed in ASL communication are shown.

Proposed Framework.


List of Modules
Data Ingestion
Upload & extract data from different sources, store and analyse in data lake, perform missing value treatment, check summary of data distribution
Exploratory Data Analysis
Discover patterns and spot anomalies using ML charts, correlate features using Pair Plots, Heatmap and label, pin and share observations in real time
Data Pre-Processing
Perform categorical encoding, feature outlier detection and feature selection for Machine Learning & Time series models
Model Configuration
Configure and experiment different classification/regression models with inbuilt hyper parameter tuning functionality
Training, Fit and Evaluation
Train and cross validate model findings and assess performance though several metrices
Output Visualization
Visualize and analyse the predictions using graphical representations

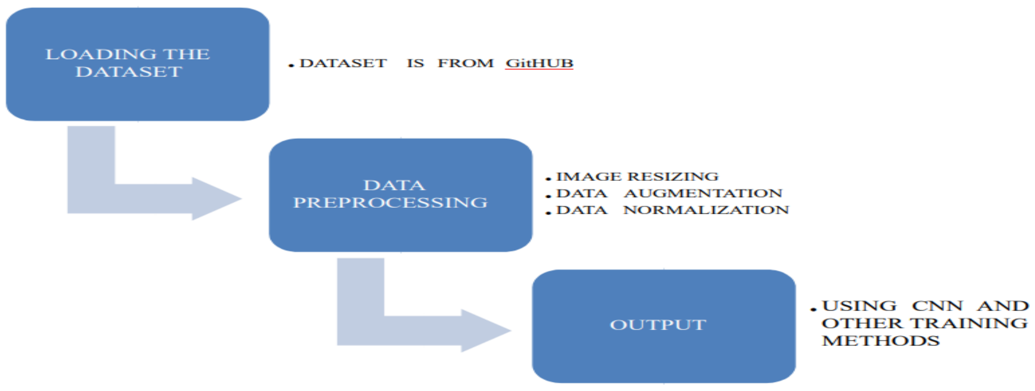


Fig: 2

Data Collection: It is comparable to the requirements analysis phase, when all necessary data and information are gathered and all project needs are considered in order to complete the project. The database is chosen at this stage for additional processing in later stages. The information and data needed for the project are acquired and examined at this point. This phase is crucial since the machine learning model's efficacy and accuracy depend on the caliber of the data that was gathered.

Data Processing: This stage involves processing all of the information and data gathered during the data collection phase in order tomove the project forward. This is the designing phase, when all needs are thoroughly examined to ensure the project develops properly. Following collection, the data is cleaned up and formatted so that machine learning algorithms may use it. Techniques for data transformation, pre-processing, and cleansing are used in this step

Model Selection/ Training: Data training models are chosen following database and dataset analysis in the data processing stage using these data models. To create the ideal model training and obtain the most appropriate outputs, the database is processed.

Development: Additional development takes place based on models and training model procedures from the preceding phase. During this phase, a software framework related to the training outcomes and data model findings is constructed.

Testing: A different set of data that wasn't utilized for training is used to assess the trained model. The testing phase's goal is to evaluate the model's functionality and capacity for generalization on hypothetical cases that mimic real-world situations. The project is put through a number of procedures to ensure that it is free of errors and functions properly.

Evaluation: Following training, the model's accuracy in translating the ASL alphabet into text was assessed using the testing dataset. It was discovered that the model has an extremely high accuracy of 98.92%. In order to make sure the model wasn't overfitting the training set, the loss function was also examined. Furthermore, the accuracy of the model was tested using data that had not been seen before using some custom photographs. These photos could be reliably classified by the model, demonstrating its stability and generalizability.

In summary, this study's CNN model has shown promising results in identifying hand motions in American Sign Language (ASL). Notably, applying preprocessing methods like picture scaling, data augmentation, and data normalization greatly increased the model's efficacy. The exceptional precision of the model indicates its potential for trustworthy hand gesture identification. Additionally, the suggested model is easily adaptable to other sign languages such as ISL with little modifications, proving its usefulness as a versatile tool for sign-to-text conversion.

I. Model Used

The Architecture is divided into layers as follows:

Convolutional Neural Network (CNN):

The layered design of convolutional neural networks (CNNs)[9] sets them apart from conventional neural networks. There are three dimensions to the arrangement of neurons in each layer of a CNN: height, breadth, and depth. Rather than being entirely linked, the neurons in a particular layer are only partially connected to a portion of the preceding layer. A final output layer at the conclusion of the CNN architecture has a set number of classes. This is so that the CNN can categorize the input picture in accordance with the predetermined classes by reducing the entire image to a single vector of class scores.

Pooling Layers:

One kind of layer that is frequently utilized in convolutional neural networks (CNNs) is the pooling layer[10]. They work on the output of convolutional layers, which come before them, to reduce the feature map's spatial dimensions without losing any crucial information.

The feature map is divided into tiny, non-overlapping areas known as pooling regions or receptive fields in order for the pooling layer to function. After that, a pooling function, such max pooling or average pooling, is applied to each receptive field in order to compress it into a single number. The network's next tier receives this compressed output after that.

By reducing the size of the feature map, pooling layers assist to improve computation efficiency and lessen the likelihood of overfitting. Additionally, they aid in improving the network's translation invariance, which enables it to identify the same item anywhere it may be seen in the picture.

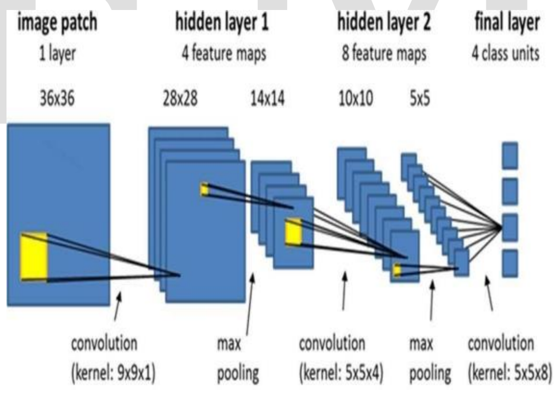


Fig- 3: Convolutional Neural Network processing

TensorFlow:

TensorFlow [11] is an all-inclusive open-source machine learning platform that provides a versatile ecosystem of tools, libraries, and resources to help academics push the limits of the field and developers create and implement machine learning-powered applications with ease.

Keras:

A Python-based high-level neural network framework called Keras [12] serves as a TensorFlow wrapper. It contains pre-built implementations of frequently used neural network features, such as layers, goals, activation functions, and optimizers, as well as tools for processing text and picture input. It is used to quickly design and test neural networks with little to no scripting.

OpenCV:

Open-Source Computer Vision, also known as OpenCV [13], is an open-source programming library that offers a number of real-time computer vision capabilities, such as video capture, object and face identification, image processing, and analysis. Although its main interface is built in C++, a broader spectrum of developers can use it because interfaces for Python, Java, and MATLAB/OCTAVE arealso available.

II. Methodology

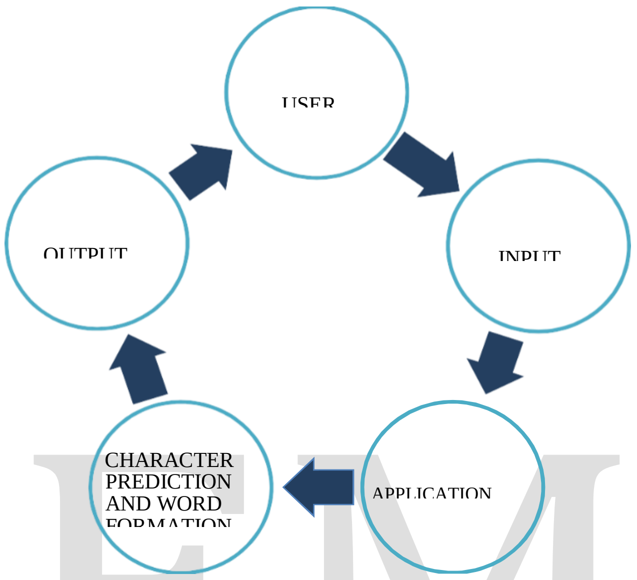


FIG: 4

Conv2D layer:

It is a CNN's main constituent. A Conv2D layer uses a set of filters to extract certain features from an input image. Every filter looks at the image, multiplying and adding each element separately. A feature map that shows the spatial distribution of the filter in the input image is the layer's output.

ReLu layer:

The Rectified Linear Unit (ReLU) activation function is used to give the Conv2D layer's output nonlinearity. ReLu [14] is a simple and effective function that preserves positive numbers while reducing negative integers to zero.

MaxPooling2D layer:

It is a down sampling process that keeps important information while shrinking the spatial size of the feature map. The subsequent layer receives the greatest value from a rectangular area of the feature map.

Flattening layer:

In order to feed it into the fully linked layers, it transforms the output of the final MaxPooling2D layer into a one-dimensional vector.

Fully connected layer:

This common neural network layer connects all of the neurons in the current layer to all of the neurons in the layer below. After passing through the fully connected layer with the flattened feature vector, a vector of class scores is obtained.

Softmax layer:

The last layer of the CNN turns class scores into probabilities using the softmax approach [15]. It is reasonable to interpret the result as a probability distribution across all conceivable classes since the softmax function guarantees that the total of all class probabilities equals one.

Layer 1:

1. After feature extraction, apply the Gaussian Blur filter and threshold to the captured frame to obtain the processed picture.
2. After processing the picture, the CNN model is used to make predictions. If a letter is identified in more than 50 frames, it is printed and used to build the word.
3. The blank sign is used to represent the space between words.

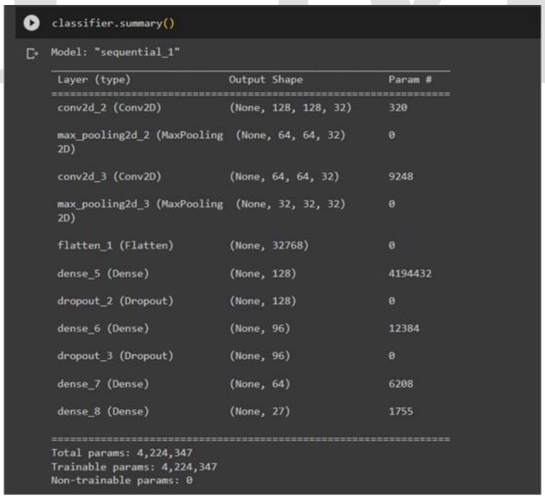
Layer 2:

1.We identify other symbol sets that yield comparable detection results.

2. Next, we use classifiers designed specifically for those sets to classify between those sets.

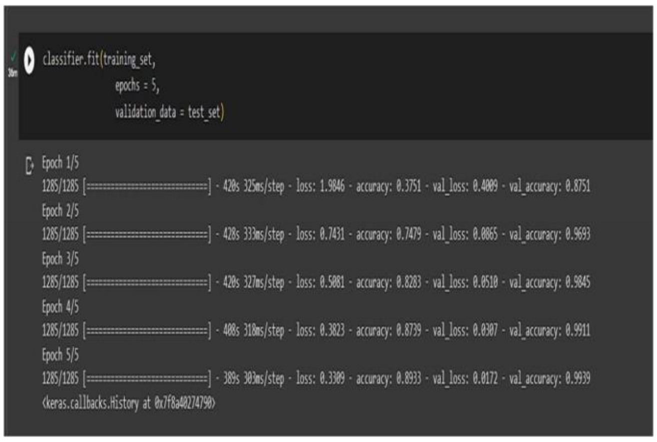
Results

The accuracy of the model is a very high 98.92%. One of the most important assessment metrics for machine learning models is accuracy, which quantifies how well the model performs in accurately predicting the target variable or class labels. The accuracy score assesses how well the model can translate sign language movements into their equivalent written representations and classify them. A high accuracy means that the model has effectively learnt and assimilated the characteristics and patterns of various sign movements, allowing for precise text translation and prediction. The CNN model is a dependable tool for jobs involving the conversion of sign language to text because of its high accuracy, which indicates that it is capable of identifying and discriminating between distinct signmotions.



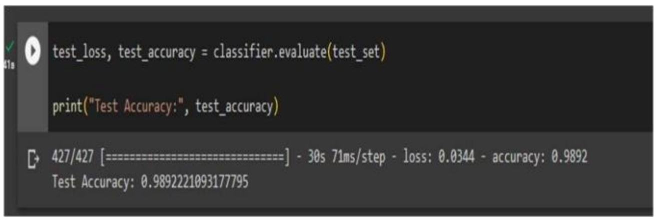
```
classifier.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_2 (Conv2D)           (None, 128, 128, 32)      320

 max_pooling2d_2 (MaxPooling  (None, 64, 64, 32)       0
 2D)

 conv2d_3 (Conv2D)           (None, 64, 64, 32)        9248

 max_pooling2d_3 (MaxPooling  (None, 32, 32, 32)       0
 2D)

 flatten_1 (Flatten)         (None, 32768)             0

 dense_5 (Dense)             (None, 128)               4194432

 dropout_2 (Dropout)         (None, 128)               0

 dense_6 (Dense)             (None, 96)                12384

 dropout_3 (Dropout)         (None, 96)                0

 dense_7 (Dense)             (None, 64)                6208

 dense_8 (Dense)             (None, 27)                1755

=================================================================
Total params: 4,224,347
Trainable params: 4,224,347
Non-trainable params: 0
```



```
classifier.fit(training_set,
               epochs = 5,
               validation_data = test_set)

Epoch 1/5
1285/1285 [==========================] - 420s 325ms/step - loss: 1.9846 - accuracy: 0.3751 - val_loss: 0.4009 - val_accuracy: 0.8751
Epoch 2/5
1285/1285 [==========================] - 428s 333ms/step - loss: 0.7431 - accuracy: 0.7479 - val_loss: 0.0865 - val_accuracy: 0.9693
Epoch 3/5
1285/1285 [==========================] - 420s 327ms/step - loss: 0.5081 - accuracy: 0.8283 - val_loss: 0.0510 - val_accuracy: 0.9845
Epoch 4/5
1285/1285 [==========================] - 408s 318ms/step - loss: 0.3823 - accuracy: 0.8739 - val_loss: 0.0307 - val_accuracy: 0.9911
Epoch 5/5
1285/1285 [==========================] - 389s 303ms/step - loss: 0.3309 - accuracy: 0.8933 - val_loss: 0.0172 - val_accuracy: 0.9939
<keras.callbacks.History at 0x7f8a40274790>
```

FIG: 5



```
test_loss, test_accuracy = classifier.evaluate(test_set)

print("Test Accuracy:", test_accuracy)

427/427 [==============================] - 30s 71ms/step - loss: 0.0344 - accuracy: 0.9892
Test Accuracy: 0.9892221093177795
```

FIG: 6

We ran five epochs; the accuracy in the first one was 0.8751, and the accuracy increased with each additional epoch, rising from 0.9693 in the second epoch to 0.9939 in the fifth. Furthermore, the validation loss decreases to 0.0172 from 0.4009. The Accuracy function, which is used to calculate the model's overall accuracy, displays an accuracy of 98.92%. The CNN model's capacity to generate accurate predictions and successfully identify the underlying patterns and characteristics in the sign language data is demonstrated by its high accuracy. This achievement shows that the model has gained a thorough comprehension of different sign movements, which allows it to convert them into text with accuracy.

The ability to recognize small variations in hand gestures, postures, and facial expressions is crucial for sign language understanding. The CNN model's high accuracy shows how good it is at recognizing and differentiating these fine-grained features. Once the model has mastered the ability to identify and understand the unique properties of various sign movements, it can convert them into their corresponding textual counterparts.

CONCLUSION AND FUTURE SCOPE

An extensive sign language dataset was gathered and preprocessed at the start of the study. This dataset included pictures of several sign language movements and the written labels that went with them. To improve their quality and prepare them for additional analysis, the photos underwent preprocessing.

After that, a CNN architecture was created and put into practice utilizing Python and well-known deep learning frameworks like PyTorch and TensorFlow. The prepared dataset was used to train the CNN model, with a focus on maximizing accuracy and performance. Handling difficulties including background noise, illumination, and variable gestures was carefully considered.

To sum up, this project report offers a unique method for converting sign language to text by utilizing Python and Convolutional Neural Networks (CNN). The project's main goal is to correctly translate sign language motions into their equivalent written representations. It starts with gathering and preprocessing datasets, then uses well-known deep-learning frameworks to design and build a CNN architecture. The trained model can identify and categorize sign language motions with a high accuracy of 98.92%. Additionally, the proposed system offers an intuitive user interface for real-time applications, facilitating efficient communicationbetween sign language users and non-users.

Future directions in sign-to-text conversion provide various viable avenues to pursue further advancements in the discipline.

1. Enhancing Gesture Recognition: Future research efforts may concentrate on enhancing the precision and resilience of gesture recognition. Investigating cutting-edge deep learning architectures that may better capture the temporal relationships in sign language movements, such transformer models or recurrent neural networks (RNNs), may be one way to do this.

2. Expanding the Dataset: The generality and effectiveness of the sign-to-text conversion system may be enhanced by expanding the dataset's size and variety. An inclusive and thorough system may result from gathering further examples of sign language gestures from different people, racial and cultural origins, and persons.

3. Real-time Tracking and Translation: Enhancing the system's usability and responsiveness can be achieved by integrating real-time gesture tracking and translation capabilities. More fluid and organic communication might be feasible by using computer vision techniques to monitor gestures in real-time video feeds and deliver an immediate translation into text.

4. Multi-modal Approaches: Investigating multi-modal strategies that integrate auditory and visual inputs might enhance the system's accuracy and resilience even more. The inclusion of auditory cues, such as hand gestures or voice responses, to sign language can enhance the understanding of gestures by adding more context.

5. Adaptation to User Preferences: In the future, developing the system to accommodate different user preferences and sign language dialects may prove to be a beneficial approach. Enabling users to customize the system based on their preferred signing style or geographical differences can significantly improve user happiness and communication efficacy.

6. Mobile Applications and Accessibility: The availability and accessibility of the sign-to-text conversion system may be increased by creating mobile applications for tablets and smartphones. As a result, people would be able to take the system with them and use it to communicate in a variety of settings.

7. Integration with Augmented Reality (AR): A more engaging and natural user experience may be achieved by using augmented reality (AR) technologies to overlay text translations right onto the live video stream. There's no need to flip between screens or devices when seeing the translated text superimposed on the live video broadcast.

8. Collaboration and Communication Tools: It might be beneficial to include tools that facilitate user communication and cooperation. This might include features that help those who are hard of hearing or deaf connect with others more successfully, including video conferencing or real-time chat.

REFERENCES

[1]Oliver Sacks Quote: "Sign language is the equal of speech, lending itself equally to the rigorous and the poetic, to philosophical analysis o ." (n.d.).

https://quotefancy.com/quote/879000/Oliver-Sacks-Sign-language-is-the-equal-of-speech-lending-itself-equally-to-the-rigorous

[2]World Health Organization: WHO. (2023). Deafness and hearing loss. www.who.int. https://www.who.int/news-room/fact- sheets/detail/deafness-and-hearing-loss

[3] Kaliyamoorthi, M., Patidar, A., Walia, P., & Roy, A. B. (2018). Hand Gesture Detection and Conversion to Speech and Text.

ResearchGate. Retrieved from

https://www.researchgate.net/publication/329305443_Hand_Gesture_Detection_and_Conversion_to_Speech_and_Text

[4] Chakladar, D. D., Kumar, P., Mandal, S., Roy, P. P., Iwamura, M., & Kim, B. (2021). 3D Avatar Approach for Continuous Sign Movement Using Speech/Text. Applied Sciences, 11(8), 3439. https://doi.org/10.3390/app11083439

[5] Huerta-Enochian, M. (2022, June 1). KoSign Sign Language Translation Project: Introducing The NIASL2021 Dataset. Retrieved from https://aclanthology.org/2022.sltat-1.9/

[6] Mohd Rum, Siti Nurulain & Boilis, Braxton. (2021). Sign Language Communication through Augmented Reality and Speech Recognition (LEARNSIGN). International Journal of Engineering Trends and Technology. 69. 125-130. 10.14445/22315381/IJETT- V69I4P218.

[7] Cornell University Izutov, E. (2020, April 10). ASL Recognition with Metric-Learning based Lightweight Network. Retrieved from https://arxiv.org/abs/2004.05054

[8] Duarte, A. (2020, August 18). How2Sign: A Large-scale Multimodal Dataset for Continuous American Sign Language. Retrieved from https://arxiv.org/abs/2008.08143

[9] GeeksforGeeks. (2023). Convolutional Neural Network CNN in Machine Learning. GeeksforGeeks. Retrieved from https://www.geeksforgeeks.org/convolutional-neural-network-cnn-in-machine-learning/

[10] Djordjevic, I. B. (2021). Quantum Machine Learning. Elsevier eBooks, 619–701. https://doi.org/10.1016/b978-0-12-821982-9.00007-1

[11] TensorFlow. (n.d.). TensorFlow. Retrieved from https://www.tensorflow.org/

[12] Team, K. (n.d.). Keras: Deep Learning for humans. Retrieved from https://keras.io/

[13] OpenCV. (2023, May 9). Home - OpenCV. Retrieved from https://opencv.org/

[14] Brownlee, J. (2020). A Gentle Introduction to the Rectified Linear Unit (ReLU). MachineLearningMastery.com. Retrieved from https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/

[15] Brownlee, J. (2020). Softmax Activation Function with Python. MachineLearningMastery.com. Retrieved from https://machinelearningmastery.com/softmax-activation-function-with-python/