2024 International Conference on Emerging Smart Computing and Informatics (ESCI)
AISSMS Institute of Information Technology, Pune, India. Mar 5-7, 2024

2024 International Conference on Emerging Smart Computing and Informatics (ESCI) | 979-8-3503-0661-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/ESCI59607.2024.10497370

Transformative Advancements: Sign Language Conversion to Text and Speech

1stS Balakrishna Reddy
Assistant professor,Department of CSE(DS)
CVR College of Engineering
Hyderabad, India
sama.balakrishnareddy@gmail.com

2nd Ahmed Shahebaaz
Assistant professor,Department of CSE(DS)
CVR College of Engineering
Hyderabad, India
ahmedshahbaaz11@gmail.com

3rd Ratnam Dodda
Sr.Assistant Professor,Department of CSIT
CVR College of Engineering
Hyderabad, India
ratnam.dodda@gmail.com

4th C.Raghavendra
Associate Professor, Department of CSE(CS)
CVR College of Engineering
Hyderabad, India
crg.svch@gmail.com

***Abstract*—The integration of sign language conversion to text and speech represents a transformative technological advancement that serves as a bridge between the deaf and hearing communities. This innovation relies on real-time interpretation, utilizing computer vision and machine learning algorithms to capture and understand the nuances of sign language gestures, including hand movements, facial expressions, and body language. The primary objective is to facilitate seamless communication and comprehension between deaf and hearing individuals.This technology holds immense promise in enhancing accessibility for the deaf community, enabling them to express themselves more fluently in various everyday scenarios. From ordering at a restaurant to participating in classroom discussions, sign language is translated instantly into understandable text and speech. Additionally, it promotes inclusivity by eliminating the need for interpreters, allowing hearing individuals to engage in meaningful conversations with their deaf counterparts.In the realm of education, this technology has the potential to revolutionize the learning of sign language and support the integration of deaf students into mainstream educational environments. As artificial intelligence and computer vision advancements progress, the accuracy and accessibility of sign language conversion are expected to improve further, contributing to a more inclusive and interconnected world for all. This signifies a significant stride in breaking down communication barriers and fostering equal opportunities for deaf individuals.**

***Index Terms*—Sign language conversion,Text-to-speech ,Machine learning algorithms , Deaf and hearing communities, Computer vision**

I. Introduction

In the ever-evolving landscape of technology, the choice of the right tools can be the linchpin differentiating between effective solutions and persistent challenges. When it comes to the field of sign language recognition, the traditional methods often come with limitations that hinder accessibility and inclusivity. Recognizing these hurdles, our project embarks on a journey of innovation and empowerment by delving into the world of sign language recognition technology [1]. Traditional sign language recognition systems have been marred by complexities and constraints, including limited adaptability and accessibility. In response to these challenges, our project seeks to explore a novel solution: the development of advanced sign language recognition technology. This endeavour is rooted in the firm belief that technology should empower, not inhibit, the communication and interaction of deaf individuals with the world at large [2]. We aim to harness technology to make sign language more accessible and inclusive, ensuring that individuals who rely on this form of communication have the freedom to express themselves and participate fully in society. This project serves as a beacon of accessibility, empowerment, and progress in the field of sign language recognition [3].

A. Motivation

The primary motivation for embarking on this project is to address the evolving needs and challenges faced by developers in the realm of software development. Traditional Integrated Development Environments (IDEs) have long been powerful tools, but they come with their own set of obstacles, such as intricate installation procedures, compatibility issues, and the restrictions to specific devices, which can hinder the efficiency and flexibility of developers in their work. Our project is ultimately motivated by the goal of providing developers with the information and resources they need to make wise choices regarding their coding environments. By offering insights into the benefits and factors to take into account of web-based code editors, we hope to make a positive impact on the software development landscape and make it more effective, inclusive, and flexible. Our goal is to give developers the freedom to code as they see fit, enabling them to work more productively and efficiently. [4] [5].

B. Literature Survey

The research initiative led by Luv (IIT2016085), Nitin Raj Singh (IIT2016132), Ravi Chandra (IIT2016141), Sheldon

Tauro (IIT2016137), and Siddhant Gautam (IIT2016069), students of IIT in 2019, is focused on the development of a computer application designed to recognize and interpret real-time hand gestures in American Sign Language (ASL) through video feeds. This endeavor is underpinned by an extensive literature survey to inform the application's framework and methodology [6].

This project's main goal is to process and comprehend complex hand movements in ASL by utilizing cutting-edge machine learning models. The intended result is real-time generation of corresponding text outputs on the screen to improve accessibility and communication for ASL-dependent people. Through the use of this creative method, sign language and text can be seamlessly combined to enable deaf and hard-of-hearing people to communicate in a variety of social, professional, and educational contexts. [7].

The comprehensive literature survey conducted by the team has contributed to the refinement of their methodologies, ensuring that the application aligns with the latest advancements in the field. The potential impact of this project extends beyond the immediate realm of ASL, with implications for accessibility in technology, education, and communication at large. By synthesizing existing knowledge and incorporating cutting-edge techniques, this research initiative is poised to make a significant contribution to the field of assistive technology. The collaborative efforts of the dedicated team members underscore the value and impact of this solution within the ASL community and beyond.

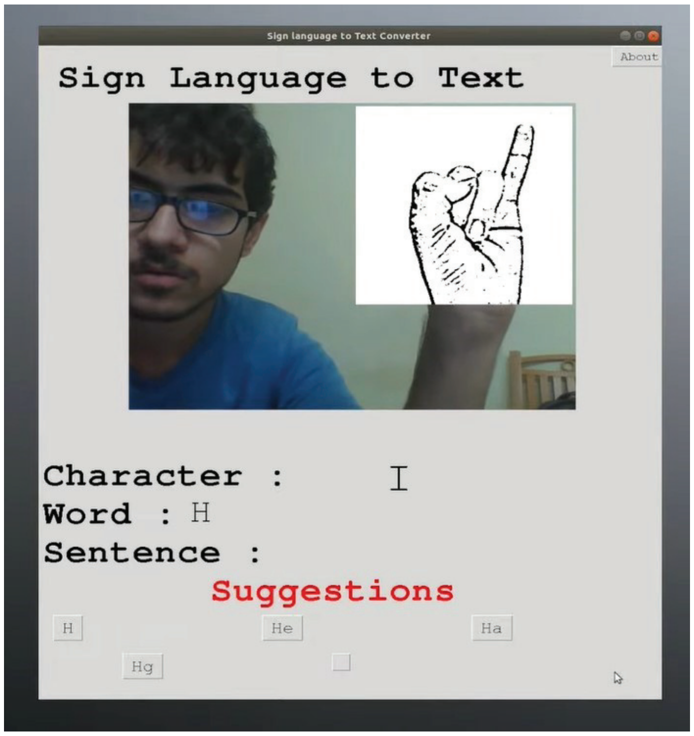


Fig. 1. Existing Model

C. Limitations of Existing Work

The project aimed at creating a computer application for real-time American Sign Language (ASL) recognition has shown promise, but it does come with certain limitations. Here are some of the key limitations of the ASL recognition model:

1) Lighting Sensitivity: The model's performance is heavily dependent on good lighting conditions. Poor lighting can significantly affect its accuracy, limiting its usability in various environments.

D. Plain Background Requirement

It is essential to have a plain background for precise gesture detection. In real-world scenarios, where backgrounds are frequently complex and dynamic, this limitation may not be feasible. [8].

1) Text-to-Speech Conversion Absence: The system cannot convert the recognized sign language gestures into audible speech. This restricts its usability for individuals who may rely on spoken language as their primary mode of communication [9].

2) Word Recommendation Feature: The absence of a word recommendation feature means that the model does not offer assistance or suggestions for sign language users, which could be particularly useful for individuals who are less proficient in ASL or for enhancing communication efficiency [10].

II. Methodology

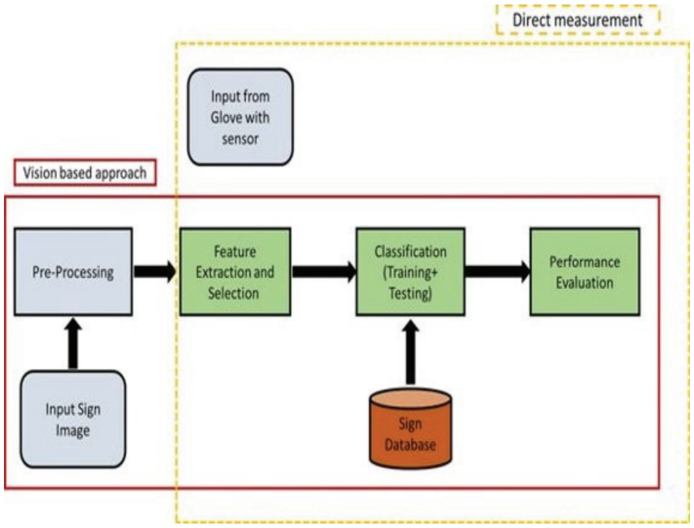


Fig. 2. System Architecture

1) The system architecture for sign language recognition is a comprehensive framework that seamlessly integrates multiple components to provide accurate and efficient sign language recognition. Input from Sensor is responsible for capturing sign language gestures through sensors embedded in a glove or wearable device [11].
2) A number of essential elements that process visual sign language cues are included in the vision-based approach.

Visual sign language cues can be captured as images or video streams using input sign image. [12].

3) To raise the data quality, the preprocessing part refines and cleans the input sign images. The process of feature extraction and selection involves identifying the most discriminative features for classification by extracting pertinent features from the preprocessed images. [12].
4) In order to identify sign language gestures against real-time or database-based sign language inputs, the classification module uses machine learning techniques.
5) The accuracy and effectiveness of the sign language recognition system are evaluated by the performance evaluation component. Metrics like error rates, processing times, and recognition accuracy may be included. [13].

A. Gesture Classification and Convolutional Neural Network (CNN)

A class of neural networks called CNN is very helpful in resolving computer vision issues. Their source of inspiration was the actual visual perception that occurs in the brain's visual cortex. They employ a filter or kernel to iterate over all of the image's pixel values and perform calculations by establishing suitable weights to allow for the identification of particular features. Convolution, max pooling, flatten, dense, dropout, and a fully connected neural network layer are some of the layers that CNN is composed of. When combined, these layers provide a very potent tool for feature recognition in images. Low-level features are detected by the initial layers, which then progressively start to detect higher-level features that are more complex. [14]. The neurons in CNN layers are arranged in three dimensions: width, height, and depth, in contrast to regular neural networks. Rather than being fully connected to every other neuron in the layer, a layer's neurons will only be connected to a small portion of the layer (window size) preceding it. Furthermore, since we will have reduced the entire image to a single vector of class scores at the end of the CNN architecture, the final output layer would have dimensions (number of classes). [15]. A tiny window size,

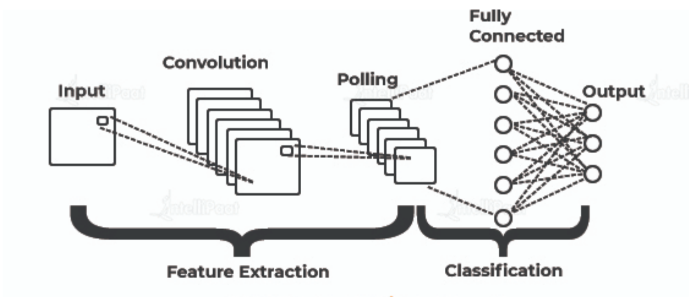


Fig. 3. Convolutional Neural Network

usually 5 by 5, that extends to the input matrix's depth is used in the convolution layer. Learnable window-sized filters make up the layer. We calculated the . product of the filter entries and input values at a specific position and slid the window by stride size [usually 1] for each iteration. This process results in the creation of a 2-dimensional activation matrix, which indicates the response of the matrix at each spatial position [16].

Pooling Layer: In order to reduce the size of the activation matrix and, eventually, the learnable parameters, we employ a pooling layer. There are two different kinds of pooling: textbfMax Pooling: In this type of pooling, we take a window size (for instance, a 2*2 window) and only take the maximum of four values. We'll close this window and carry on until we eventually have an activation matrix that is half the size it was originally. b. Average Pooling: This method involves averaging all values within a given window [17]. We divided

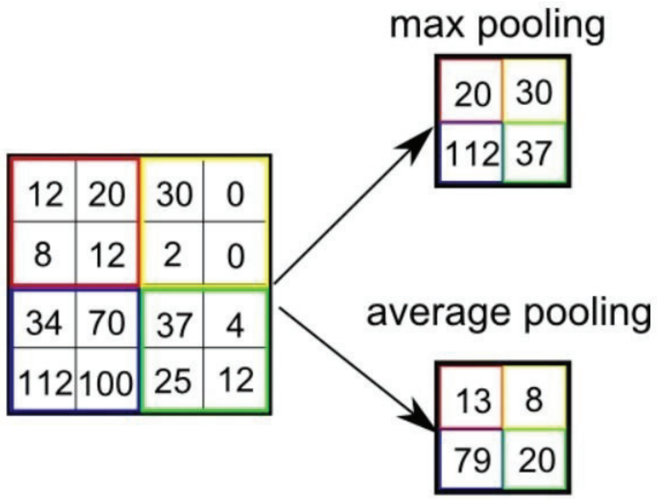


Fig. 4. Types of pooling in CNN

the whole 26 different alphabets into 8 classes in which every class contains similar alphabets.

There will be a probability assigned to each gesture label. The predicted label will be considered to be the one with the highest probability. Thus, we will further classify into single alphabets such as a, e, m, n, s, or t after the model uses mathematical operations on hand landmarks to classify [aemnst] in a single class. Ultimately, our approach produced improved accuracy both with and without a clear background and optimal lighting. We obtain even more accurate results when the lighting is good and the background is clear [18]. **Text To Speech Translation:** One of the features of the system is its ability to convert recognized gestures into spoken words. In order to accomplish this, we have incorporated the pyttsx3 library, which successfully produces audible speech from the recognized words. The text-to-speech output is a useful feature even though it's a fairly simple method because it closely resembles natural, in-person conversations. This feature makes the sign language recognition system more approachable and user-friendly, facilitating smooth user-system interaction while also improving the user experience and promoting efficient communication.

Algorithm 1: Convolutional Neural Network

```
Input: Input data $X$, labels $Y$, number of epochs $N$
Output: Trained CNN model
// Define CNN architecture
1 ConvLayer(32 filters, kernel size (3, 3),
    activation='relu', input shape=(height, width,
    channels));
2 ConvLayer(64 filters, kernel size (3, 3),
    activation='relu');
3 Flatten();
4 Dense(128 units, activation='relu');
5 Dense(num_classes, activation='softmax');
// Compile the model
6 Compile(optimizer='adam',
    loss='categorical_crossentropy',
    metrics=['accuracy']);
// Train the model
7 Fit(X, Y, epochs=N);
```

III. Results and Discussion

Python: Python is chosen as the programming language for its simplicity and wide range of libraries. Its clean syntax and extensive support for data processing make it a versatile choice for developing a complex project like sign language recognition [19]. **TensorFlow:** TensorFlow is a popular deep learning framework that's used for training machine learning models. In this project, it's employed to create, train, and fine-tune the neural network [20]. **OpenCV:** OpenCV is a Python library dedicated to image processing. In the context of this project, it is invaluable for tasks like capturing video input, processing image data, detecting hand gestures, and tracking the movements of the hands in real-time video streams [21]. **MediaPipe:** MediaPipe is a framework designed for real-time computer vision applications. It plays a vital role in tracking human hand gestures in real-time video feeds. MediaPipe provides pre-trained models and tools for efficient hand tracking, making it easier to extract meaningful information from video streams [22]. **Pyttsx3:** Pyttsx3 is a Python library that simplifies the integration of text-to-speech functionality. In this project, it allows the system to convert text-based output, such as translations or explanations, into audible speech, making the project more accessible to users [23]. **Tkinter :** Tkinter is a built-in Python library for creating graphical user interfaces (GUIs). It provides a set of widgets and tools for designing user-friendly interfaces that enable users to interact with the sign language recognition system. With Tkinter, you can create buttons, labels, text boxes, and other GUI elements [24]. **Enchant:** Enchant is a library that provides a unified interface for spell-checking. While not directly related to sign language recognition, it can be used to ensure the accuracy of any text-based output or input in the project. This is especially important if the system generates or displays text, such as translations or captions [25].

A. Data Acquisition

Data acquisition is a critical phase in a sign language recognition project. It involves the collection of a diverse dataset of hand gestures, encompassing variations in skin tones and capturing scenarios with different viewpoints, scales, and camera speeds. This diversity ensures that the machine learning model trained on this dataset can effectively recognize and generalize hand gestures under various real-world conditions. The dataset's comprehensiveness is pivotal for the success of the project, as it allows the model to adapt to the diverse ways sign language can be expressed, ultimately making the system more inclusive and accurate for a wide range of users [26].

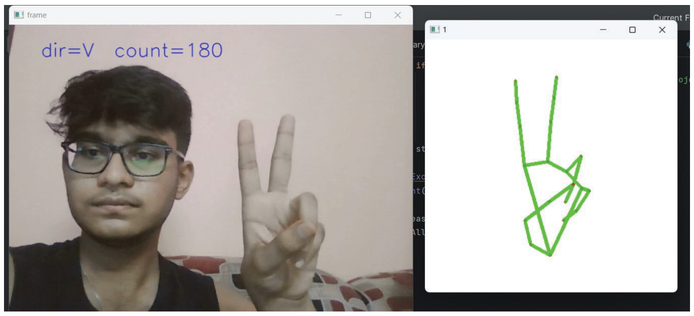


Fig. 5. Data Collection

B. Data Preprocessing and Feature Extraction

In the critical stage of data pre-processing and feature extraction for a sign language recognition project, the primary aim is to refine and prepare the input data for effective analysis. It commences with the identification and isolation of the region of interest (ROI) within the captured image, typically encompassing the hand gestures being performed. Leveraging the powerful OpenCV library, this selected area is meticulously cropped, enabling a more focused and precise analysis of the hand's movements. Subsequently, the grayscale transformation is applied to the cropped image, a fundamental step that simplifies the data while retaining crucial visual information necessary for in-depth analysis. To enhance the image quality, a Gaussian blur is introduced, effectively reducing noise and ensuring a smoother, more uniform appearance [27].

Following grayscale conversion, a thresholding method is employed to convert the image into a binary representation, effectively segregating the hand from the background. This binary image serves as the canvas for the identification and extraction of hand landmarks, which encompass vital points on the fingers and palm. A precise and consistent representation of the hand's position and shape is then produced by drawing and connecting these landmarks on a blank white canvas. The system will eventually be able to reliably and precisely recognize and interpret a wide range of sign language gestures thanks to this thorough and standardized visual representation of the hand, which is essential for later feature extraction and analysis. [28].

Gesture Recognition Software: This specialized software harnesses the power of cutting-edge computer vision algo-

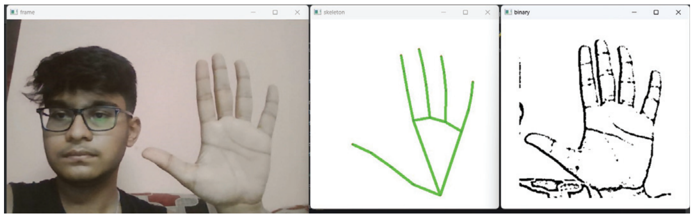


Fig. 6. Data Preprocessing

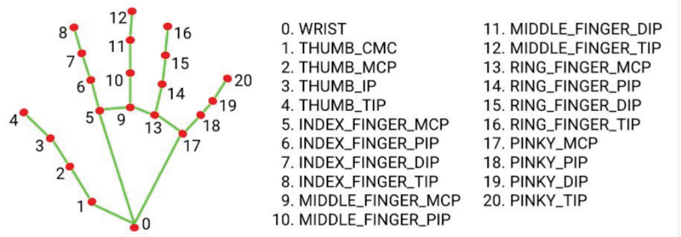
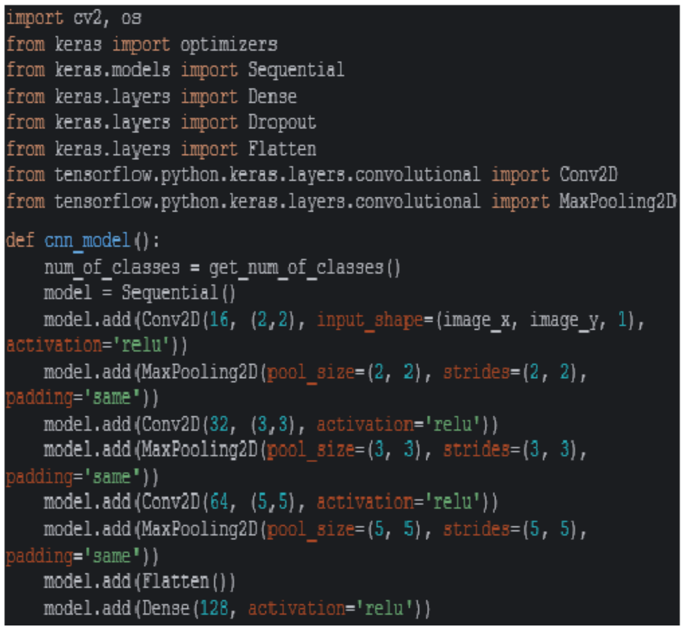


Fig. 7. Mediapipe Landmarks



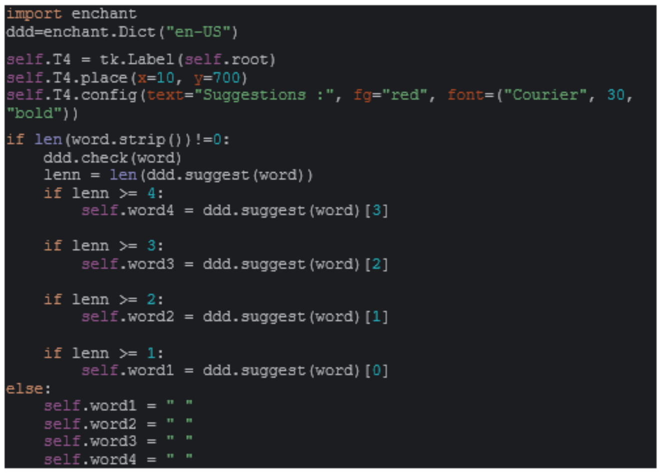
```python
import cv2, os
from keras import optimizers
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
from tensorflow.python.keras.layers.convolutional import Conv2D
from tensorflow.python.keras.layers.convolutional import MaxPooling2D

def cnn_model():
    num_of_classes = get_num_of_classes()
    model = Sequential()
    model.add(Conv2D(16, (2,2), input_shape=(image_x, image_y, 1),
activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2),
padding='same'))
    model.add(Conv2D(32, (3,3), activation='relu'))
    model.add(MaxPooling2D(pool_size=(3, 3), strides=(3, 3),
padding='same'))
    model.add(Conv2D(64, (5,5), activation='relu'))
    model.add(MaxPooling2D(pool_size=(5, 5), strides=(5, 5),
padding='same'))
    model.add(Flatten())
    model.add(Dense(128, activation='relu'))
```

Fig. 8. Gesture Recognition Software

rithms and machine learning models to proficiently recognize and decipher sign language gestures. In this intricate process, the software relies on robust libraries. Notably, OpenCV plays a pivotal role in capturing the images, excelling in image processing, and providing a reliable foundation for recognizing hand movements and gestures. TensorFlow and PyTorch, on the other hand, are selected for their proficiency in training Convolutional Neural Network (CNN) models, an indispensable part of the recognition process. These libraries empower the software to learn and understand the intricate patterns and variations in sign language expressions, facilitating accurate interpretation and seamless communication for the users of the system.

Natural Language Processing (NLP) Software: NLP (Natural Language Processing) software is a critical component in the sign language recognition system, tasked with transforming recognized signs into coherent written text. This intricate process entails several steps, including tokenization, part-of-speech tagging, and language modelling. To ensure the system provides an accurate and coherent interpretation of sign language, we've integrated the versatile enchant module. Enchant provides a valuable dictionary and suggestion framework, which aids in improving the accuracy and quality of the text generated from the sign language gestures. It not only enhances the precision of tokenization and language modelling but also offers helpful suggestions, enabling users to correct any potential errors and ensure the final text output is both meaningful and linguistically accurate. This integration of the enchant module reflects a commitment to user-friendliness and communication efficacy, ensuring that the system not only translates sign language accurately but also presents the results in a format that is readily understood by all.

Speech Synthesis Software: Speech Synthesis Software is a pivotal component in our sign language recognition system, responsible for the transformation of converted text

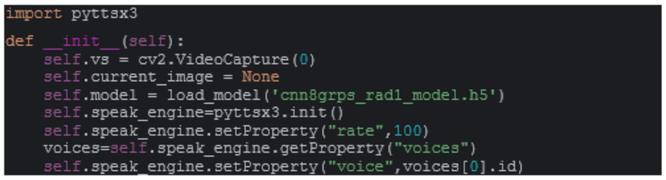


```python
import enchant
ddd=enchant.Dict("en-US")

self.T4 = tk.Label(self.root)
self.T4.place(x=10, y=700)
self.T4.config(text="Suggestions :", fg="red", font=("Courier", 30,
"bold"))

if len(word.strip())!=0:
    ddd.check(word)
    lenn = len(ddd.suggest(word))
    if lenn >= 4:
        self.word4 = ddd.suggest(word)[3]

    if lenn >= 3:
        self.word3 = ddd.suggest(word)[2]

    if lenn >= 2:
        self.word2 = ddd.suggest(word)[1]

    if lenn >= 1:
        self.word1 = ddd.suggest(word)[0]
else:
    self.word1 = " "
    self.word2 = " "
    self.word3 = " "
    self.word4 = " "
```

Fig. 9. NLP Software

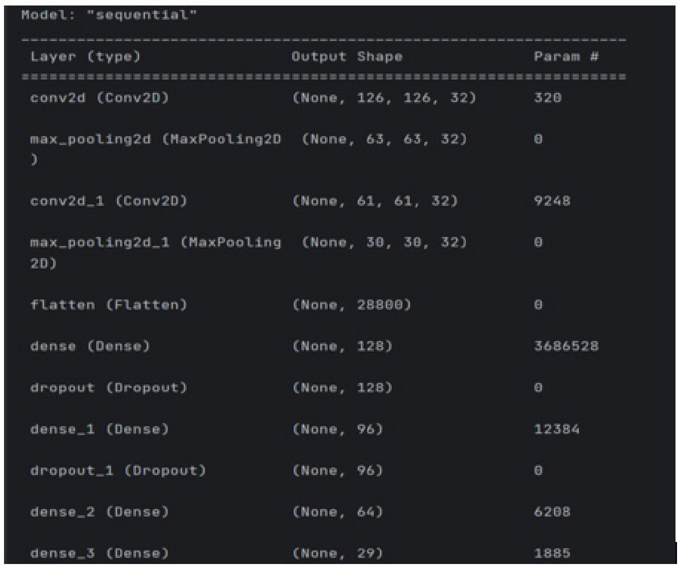
into spoken language. To fulfill this role, we've integrated the versatile ”pyttsx3” library, which serves as a reliable and efficient tool for text-to-speech conversion. With pyttsx3, the system can generate natural and intelligible speech from the translated text, providing a seamless and inclusive experience for users. This text-to-speech capability is invaluable as it bridges the gap between the system and its users, simulating real-life dialogue and enabling effective communication. This comprehensive approach underlines the commitment to providing a versatile and user-friendly sign language recognition system.

User Interface (UI): User Interface (UI) play a fundamental role in ensuring the accessibility and usability of any software application, including our sign language recognition system. To create intuitive and user-friendly interfaces, we



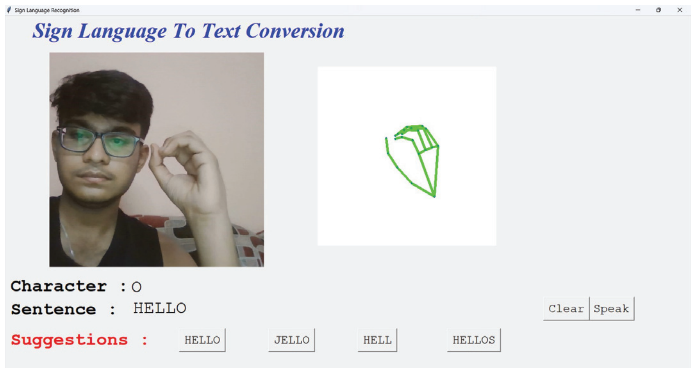
```
import pyttsx3

def __init__(self):
    self.vs = cv2.VideoCapture(0)
    self.current_image = None
    self.model = load_model('cnn8grps_rad1_model.h5')
    self.speak_engine=pyttsx3.init()
    self.speak_engine.setProperty("rate",100)
    voices=self.speak_engine.getProperty("voices")
    self.speak_engine.setProperty("voice",voices[0].id)
```

Fig. 10. Speech Synthesis Software



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 126, 126, 32)      320

 max_pooling2d (MaxPooling2D  (None, 63, 63, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 61, 61, 32)        9248

 max_pooling2d_1 (MaxPooling  (None, 30, 30, 32)       0
 2D)

 flatten (Flatten)           (None, 28800)             0

 dense (Dense)               (None, 128)               3686528

 dropout (Dropout)           (None, 128)               0

 dense_1 (Dense)             (None, 96)                12384

 dropout_1 (Dropout)         (None, 96)                0

 dense_2 (Dense)             (None, 64)                6208

 dense_3 (Dense)             (None, 29)                1885
```

Fig. 11. Model training



Sign Language To Text Conversion

Character : O
Sentence : HELLO

Fig. 12. Caption

have adopted the widely-used Tkinter library, which is a Python library for developing graphical user interfaces (GUIs). Tkinter offers a versatile and efficient toolkit for building interactive and visually appealing interfaces, making it an excellent choice for our project. With Tkinter, we have crafted interfaces that facilitate seamless interactions between users and the sign language recognition system. The library provides a range of widgets such as buttons, labels, text boxes, and more, allowing us to design a user-friendly environment that simplifies the user's experience. Tkinter's capability to create platform-independent and responsive interfaces aligns. **Model Training:**

The process of training a Convolutional Neural Network (CNN) model for sign language recognition involves several essential steps. It begins with initializing the model with random weights and feeding it a dataset of sign language images. During training, the model computes the loss, which quantifies the disparity between its predicted outputs and the actual labels. Backpropagation and optimization algorithms, such as gradient descent, iteratively adjust the model's weights to minimize this loss. The model is trained over multiple epochs, with careful adjustments of hyperparameters like learning rate and batch size as necessary. Concurrently, validation is crucial, as the model's performance on a separate validation dataset is periodically assessed. Adjustments to the model are made based on validation results to prevent overfitting. Once training is complete, the model undergoes testing on a previously unseen testing dataset, where metrics like accuracy, precision, recall, and F1 score are calculated to gauge its efficacy in recognizing sign language gestures. Fine-tuning and hyperparameter optimization may follow, exploring various parameters and model architectures to enhance performance. The trained CNN model is then deployed in a sign language recognition application for real-time gesture interpretation, with continuous monitoring and updates to adapt to evolving sign language gestures or to improve overall accuracy.

IV. Conclusion and Future Scope

In conclusion, the conversion of sign language into text and speech represents a significant advancement in technology and accessibility. This innovation has the potential to bridge communication gaps between individuals who use sign language and those who do not, thereby promoting inclusion and fostering better understanding in society. By harnessing the power of machine learning and computer vision, sign language conversion systems have made strides in real-time translation, making it easier for deaf and hard-of-hearing individuals to interact with the broader community. In the future, continued research and development in this field will likely lead to more sophisticated and accurate sign language conversion systems. These advancements will further enhance communication options for individuals who use sign language, breaking down barriers and promoting inclusivity in our increasingly interconnected world. Ultimately, the conversion of sign language into text and speech is a promising endeavor that holds the potential to enrich the lives of many individuals and contribute to a more inclusive and understanding society.

A. *FUTURE SCOPE*

In the future, the scope for sign language conversion into text and speech extends beyond just accurate recognition. We can anticipate the development of immersive augmented reality (AR) experiences, where individuals can use AR glasses or similar devices to receive real-time sign language translations overlaid on their field of view. This technology would allow for seamless communication between deaf or hard-of-hearing individuals and hearing individuals, fostering greater inclusivity in various settings, from classrooms and workplaces to social interactions. Furthermore, the future of sign language conversion holds immense potential in education. Sign language translation systems can play a pivotal role in making education more accessible to deaf and hard-of-hearing students. These systems can provide real-time sign language interpretations of classroom lectures, enabling students to engage fully with the curriculum. creation of accessible educational content, including sign language captions for online videos and interactive sign language tutorials. The continued development of sign language conversion technology will undoubtedly lead to greater inclusivity, improved education, and enhanced communication for the deaf and hard-of-hearing communities.

REFERENCES

[1] P. E. M. de Sá Escudeiro and M. Campos, "Empowering deaf learners: The promise of sign language moocs," in *European Conference on e-Learning*, vol. 22, pp. 418–421, 2023.

[2] S. S. Rautaray, "Real time hand gesture recognition system for dynamic applications," *International Journal of ubicomp (IJU)*, vol. 3, no. 1, 2012.

[3] L. Schrum and B. B. Levin, *Leading 21st-century schools: Harnessing technology for engagement and achievement*. Corwin Press, 2009.

[4] D. De Silva, S. Vidhanaarachchi, K. Siriwardana, S. Gunasekara, U. Piyumantha, and S. Thilakaratne, "Rookiescript: Constructive programming learning space for beginners," 2023.

[5] P. Xu, "A real-time hand gesture recognition and human-computer interaction system," *arXiv preprint arXiv:1704.07296*, 2017.

[6] W. C. Ritchie and T. K. Bhatia, *The new handbook of second language acquisition*. Brill, 2009.

[7] B. Kang, S. Tripathi, and T. Q. Nguyen, "Real-time sign language fingerspelling recognition using convolutional neural networks from depth map," in *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pp. 136–140, IEEE, 2015.

[8] M. De Coster, M. Van Herreweghe, and J. Dambre, "Sign language recognition with transformer networks," in *12th international conference on language resources and evaluation*, pp. 6018–6024, European Language Resources Association (ELRA), 2020.

[9] O. Koller, O. Zargaran, H. Ney, and R. Bowden, "Deep sign: Hybrid cnn-hmm for continuous sign language recognition," in *Proceedings of the British Machine Vision Conference 2016*, 2016.

[10] R. Yang, C. Zhang, R. Gao, and L. Zhang, "A novel feature extraction method with feature selection to identify golgi-resident protein types from imbalanced data," *International journal of molecular sciences*, vol. 17, no. 2, p. 218, 2016.

[11] A. Sultan, W. Makram, M. Kayed, and A. A. Ali, "Sign language identification and recognition: A comparative study," *Open Computer Science*, vol. 12, no. 1, pp. 191–210, 2022.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[14] R. Rastgoo, K. Kiani, and S. Escalera, "Sign language recognition: A deep survey," *Expert Systems with Applications*, vol. 164, p. 113794, 2021.

[15] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "{TensorFlow}: a system for {Large-Scale} machine learning," in *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pp. 265–283, 2016.

[16] G. Simion, V. Gui, and M. Otesteanu, "Vision based hand gesture recognition: A review," *international journal of circuits, systems and signal processing*, vol. 6, no. 4, pp. 275–282, 2012.

[17] D. Ratnam, P. HimaBindu, V. M. Sai, S. R. Devi, and P. R. Rao, "Computer-based clinical decision support system for prediction of heart diseases using naïve bayes algorithm," *International Journal of Computer Science and Information Technologies*, vol. 5, no. 2, pp. 2384–2388, 2014.

[18] D. Kalyani, C. Raghavendra, and K. R. Prasad, "An improved lung cancer prediction system using image processing‖," *International Journal of Recent Technology and Engineering*, vol. 8, 2019.

[19] C. Keskin, A. Erkan, and L. Akarun, "Real time hand tracking and 3d gesture recognition for interactive interfaces using hmm," *Icann/Iconipp*, vol. 2003, pp. 26–29, 2003.

[20] S. Ö. Arık, M. Chrzanowski, A. Coates, G. Diamos, A. Gibiansky, Y. Kang, X. Li, J. Miller, A. Ng, J. Raiman, *et al.*, "Deep voice: Real-time neural text-to-speech," in *International conference on machine learning*, pp. 195–204, PMLR, 2017.

[21] J. YangY̌, J. Shi, Z. Jin, and H. Zhang, "Design and implementation of a large-scale hybrid distributed graphics system," 2002.

[22] A. F. d. S. Neto, B. L. D. Bezerra, and A. H. Toselli, "Towards the natural language processing as spelling correction for offline handwritten text recognition systems," *Applied Sciences*, vol. 10, no. 21, p. 7711, 2020.

[23] B. Joksimoski, E. Zdravevski, P. Lameski, I. M. Pires, F. J. Melero, T. P. Martinez, N. M. Garcia, M. Mihajlov, I. Chorbev, and V. Trajkovik, "Technological solutions for sign language recognition: a scoping review of research trends, challenges, and opportunities," *IEEE Access*, vol. 10, pp. 40979–40998, 2022.

[24] R. Patel, J. Dhakad, K. Desai, T. Gupta, and S. Correia, "Hand gesture recognition system using convolutional neural networks," in *2018 4th international conference on computing communication and automation (ICCCA)*, pp. 1–6, IEEE, 2018.

[25] M. M. Jin, "Real time hand detection and gesture recognition system," 2011.

[26] L. A. Petrides and T. R. Nodine, "Knowledge management in education: defining the landscape.," 2003.

[27] L. Jing, E. Vahdani, M. Huenerfauth, and Y. Tian, "Recognizing american sign language manual signs from rgb-d videos," *arXiv preprint arXiv:1906.02851*, 2019.

[28] J. Bora, S. Dehingia, A. Boruah, A. A. Chetia, and D. Gogoi, "Real-time assamese sign language recognition using mediapipe and deep learning," *Procedia Computer Science*, vol. 218, pp. 1384–1393, 2023.