Sign to Speech Translator

N. Ganitha Aarthi[1],

Assistant Professor, Department of Computer Science and Design, SNS College of Engineering(Autonomous), Coimbatore, India. arthi.ganitha@gmail.com

RAHUL P[2], SHIFA ASHWATH[3], SHYAM SUBBIAH[4], SREESHA K[5]

UG Students - Department of Computer Science and Design, SNS College of Engineering(Autonomous), Coimbatore, India. ganitha.n.csd@snsce.ac.in

Abstract - In the realm of language translation technology, there exists a significant gap in addressing the needs of the mute community. This mini project focuses on the development of a groundbreaking Sign Language to Speech Translator, with the primary objective of recognizing and translating sign language efficiently. The motivation behind this endeavor stems from the inadequacy of existing gesture detection systems tailored for differently abled individuals. Our dedicated team seeks to contribute meaningfully to this field by creating a system capable of converting sign gestures into comprehensible speech for the general populace. The proposed model boasts real-time prediction capabilities for American Sign Language (ASL) gestures, achieved through a high-performing compact Convolutional Neural Network (CNN) architecture. This approach not only ensures the accuracy of predicting individual alphabets but also facilitates the seamless construction of words and sentences from these gestures. To enhance user experience, the system incorporates Text to Speech (TTS) technology, converting the translated sentences into voice modules. The compact nature of our model enables real-time applications, thereby bridging the communication gap between the Deaf and Dumb community and the rest of the world. Our Sign Language to Speech Translator is a promising solution for the communication challenges faced by the mute community. By leveraging advanced technology and an efficient model architecture, our system strives to empower individuals with speech and language capabilities, fostering inclusivity and connectivity on a global scale.

Keyword ; language - translation technology - Sign Language - gesture detection systems - Convolutional Neural Network (CNN) - American Sign Language - Text to Speech .

INTRODUCTION

In the empathy phase of design thinking, individuals immerse themselves in the user's world, actively seeking to comprehend their needs, challenges, and aspirations. Through careful observation and attentive listening, this phase establishes a profound connection with the user's viewpoint, offering crucial insights that shape the development of solutions focused on the user's experience. The insights gleaned from direct interaction and feedback with our target audience, the Deaf and Dumb community, shed light on the pressing communication challenges they face, particularly in healthcare environments. Our potential users expressed the struggle of effectively conveying their thoughts and emotions due to the communication barrier with others who may not understand sign language. They emphasized the need for a more inclusive solution, sharing instances where reliance on written notes proved inefficient and led to misunderstandings. These sentiments underscore the urgency and importance of a Sign Language to Speech Translator, as they yearn for a technology that can seamlessly bridge the communication gap, providing a more natural and meaningful interaction in healthcare scenarios.

They envision a technology that goes beyond recognizing individual sign language gestures. They desire a tool that comprehends context, enabling more nuanced and accurate communication. Recognizing the limitations of current systems, they emphasize the need for a solution that seamlessly translates gestures into coherent speech, eliminating frustrations associated with misunderstood communication. This underscores the importance of incorporating advanced technologies like Artificial Intelligence and Machine Learning in our Sign Language to Speech Translator project.

 DOI : 10.35291/2454-9150.2024.0309

Currently resort to written notes, a method they find inefficient in healthcare settings. Relying on gestures often leads to frequent misunderstandings, contributing to their frustration and sense of isolation. This makeshift communication approach underscores the pressing need for a more effective and inclusive solution, motivating the development of our Sign Language to Speech Translator. The current challenges in communication amplify the significance of a technology that seamlessly translates sign language into speech, addressing the practical shortcomings they face in their daily interactions, especially within healthcare environments.

Users express emotions of isolation, frustration, and a persistent sense of being misunderstood. The lack of a seamless communication tool accentuates their feelings of isolation in healthcare settings, where effective communication is paramount. Frustration arises from the challenges they encounter when trying to convey thoughts and emotions through written notes or gestures, often resulting in misunderstandings.

II EXISITNG SYSTEM

The focus is on crystallizing the identified needs and aspirations of our Deaf and Dumb users. The insights gathered from their experiences and expressions of isolation, frustration, and the desire for a more advanced communication tool serve as the foundation for clearly defining the project's objectives. This phase involves distilling the user requirements into specific goals, such as creating a Sign Language to Speech Translator that not only recognizes individual gestures but also comprehends context for more natural and meaningful communication. The definition stage sets the project's direction, aligning it with the precise needs of our users and emphasizing the importance of advanced technologies like Artificial Intelligence and Machine Learning in achieving these goals.

The existing communication methods for the Deaf and Dumb community primarily rely on written notes and gestures. The use of written notes, while common, is inherently limited by its inefficiency in real-time interactions, particularly in healthcare settings where quick and accurate communication is crucial. Written communication often leads to delays, misunderstandings, and can be impractical in urgent situations. On the other hand, the reliance on gestures, while more expressive, is constrained by its subjective interpretation and the potential for miscommunication. The lack of a cohesive and context-aware system hinders natural and meaningful communication, contributing to feelings of isolation among users. Gesture-based communication lacks the sophistication to convey nuanced meanings, and misunderstandings frequently arise due to the absence of a standardized and universally understood gestural language.

III. PROPOSED SYSTEM

Different methods are employed to solve this problem. Gesture-to-text conversion is a method employed by the Deaf and Dumb community to translate their sign language gestures into written text. This process involves using technologies such as cameras or sensors to capture and interpret sign language movements.

Fig.1. Mind map

The advantages of this method include its potential for real-time translation, allowing users to express themselves more fluidly. However, it may face challenges in accurately interpreting nuanced gestures, leading to potential misunderstandings. Additionally, this method relies heavily on technology, which can be a barrier in certain environments or for individuals with limited access to such devices. Written notes remain a traditional yet commonly used method among the Deaf and Dumb for communication. Individuals jot down their thoughts, questions, or information on paper to convey messages. The simplicity of this method makes it widely accessible, but it has notable drawbacks. The process is time-consuming and may not be efficient, especially in urgent or dynamic situations. Moreover, reliance on written communication limits the spontaneity and natural flow of conversation, hindering the overall communication experience. Human interpreter services involve the presence of a sign language interpreter who facilitates communication.

The Deaf and Dumb individual and the hearing person. This method is advantageous in situations where nuanced communication is crucial, such as in medical or legal settings. Human interpreters bring cultural understanding and can capture the emotion and context of the conversation. However, challenges arise due to the limited availability of qualified interpreters, leading to potential delays and a lack of privacy. Additionally, this method may not be practical for informal or daily communication needs. In conclusion, while gesture-to-text conversion offers real-time translation and written notes provide simplicity, each method has its own set of advantages and disadvantages. Human interpreter services excel in nuanced communication but face challenges in terms of availability and privacy. The Sign Language to Speech Translator project aims to address these limitations by providing a more seamless and efficient means of

 DOI : 10.35291/2454-9150.2024.0309

communication for the Deaf and Dumb community, leveraging advanced technologies to enhance accessibility and inclusivity.

V METHODOLOGY

Thus, the goal here is to build a software application to help deaf and dumb users communicate efficiently. The absence of an effective gesture detection system tailored for individuals with different abilities serves as a driving force for our team's pursuit of innovation in this domain. Our project is focused on the conversion of sign language gestures into comprehensible speech for general audiences. The model pipeline trained using machine learning, incorporating a Convolutional Neural Network (CNN) architecture, is specifically crafted for the classification of 26 alphabets and an additional symbol representing the null character. The model is trained with the American Sign Language (ASL) gesture images dataset labelled with its corresponding alphabet. This dataset consists of 17113 images belonging to 27 classes including '0':

- **Training Set: 12845 images**

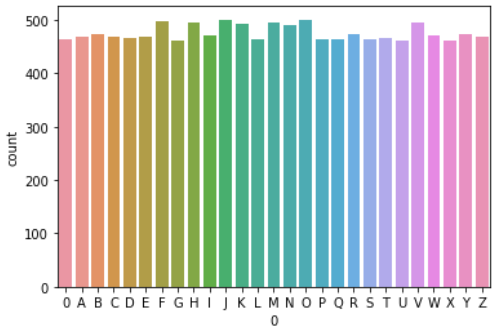

Fig 2. Training Data statistics

- Testing Set: 4368 images

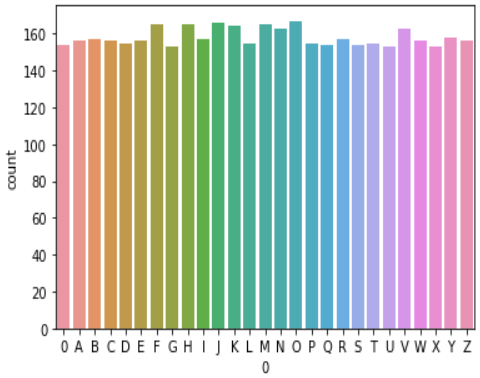

Fig.3. Testing Data statistics

Gaussian filter is used as a pre-processing technique to make the image smooth and eliminate all the irrelevant noise. Intensity is analyzed and non-maximum suppression is implemented to remove false edges. For a better pre-processed image data, double thresholding is implemented to consider only the strong edges in the images. All the weak edges are finally removed and only the strong edges are considered for the further phases.

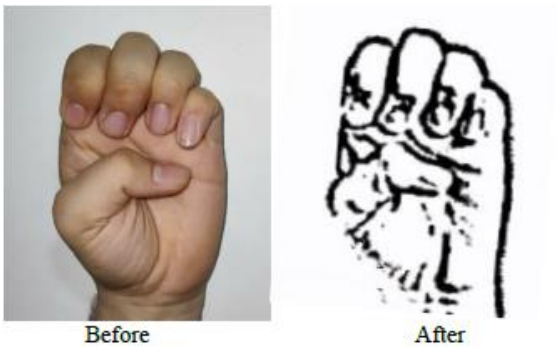

Fig.4. Gaussian filter image of hand

The provided image depicts a pre-processed image with discerned features that is subsequently forwarded to the model for the purpose of classification. The proposed work aims at converting such sign gestures into speech that can be understood by normal people. The entire model pipeline is developed by CNN architecture for the classification of 26 alphabets and one extra alphabet for null character.

1. Convolutional and Pooling Layers:

• The first layer is a convolutional layer with 32 filters (kernels) of size (3, 3) and ReLU (Rectified Linear Unit) activation. The input shape is (sz, sz, 1), where sz represents the size of the input images, and 1 indicates a single channel (grayscale).

• After the convolutional layer, a max-pooling layer is added with a pool size of (2, 2) to reduce the spatial dimensions.

• The second convolutional layer has 32 filters of size (3, 3) and ReLU activation. The input shape is automatically inferred from the previous layer's output.

• Another max-pooling layer follows.

2. Flattening:

• The Flatten layer is added to transform the 2D output of the convolutional layers into a 1D array, which can be fed into the fully connected layers.

3. Fully Connected Layers:

• Three fully connected (dense) layers follow the flattening layer.

• The first dense layer has 128 units with ReLU activation.

• A dropout layer with a dropout rate of 0.40 is added to reduce overfitting.

• The second dense layer has 96 units with ReLU activation, followed by another dropout layer.

• The third dense layer has 64 units with ReLU activation.

• The final dense layer has 27 units (assuming 27 classes for the classification task) with softmax activation, which is suitable for multi-class classification.

4. Compiling the Model:

• The model is compiled using the Adam optimizer, categorical crossentropy loss (common for multi-class classification), and accuracy as the evaluation metric.

 DOI : 10.35291/2454-9150.2024.0309

5. Summary:

• The classifier. Summary () method is called to display a summary of the model architecture, including the layers, output shapes, and the number of parameters.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 126, 126, 32)      320
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 63, 63, 32)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 61, 61, 32)        9248
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 30, 30, 32)        0
_________________________________________________________________
flatten_2 (Flatten)          (None, 28800)             0
_________________________________________________________________
dense_5 (Dense)              (None, 128)               3686528
_________________________________________________________________
dropout_3 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_6 (Dense)              (None, 96)                12384
_________________________________________________________________
dropout_4 (Dropout)          (None, 96)                0
_________________________________________________________________
dense_7 (Dense)              (None, 64)                6208
_________________________________________________________________
dense_8 (Dense)              (None, 27)                1755
=================================================================
Total params: 3,716,443
Trainable params: 3,716,443
Non-trainable params: 0
_________________________________________________________________
```

Fig.5. CNN Model creation

VI WORKING OF OUR PROJECT

This interactive application is executed within a Jupyter Notebook running in a virtual environment through Bash or Command Prompt, providing a seamless and accessible way to interpret sign language gestures in real-time. A python script implements the real-time sign language interpreter using a webcam, OpenCV for computer vision, and a pre-trained deep learning model from the Keras library for sign language classification. The script captures live video frames, extracts a region of interest corresponding to hand signs, applies image processing techniques, and feeds the processed images into the model for prediction. The recognized sign is displayed on the screen, and an accumulated text string is updated. Additionally, the script converts the recognized text into speech using the Google Text-to-Speech (gTTS) library, saves it as an audio file, and plays the audio using the play sound library.

Software used:

- Python (Back-End).
- Bash (Command Line Interface).
- Jupyter Notebook (Machine Learning tool).

Micro frameworks used:

- OpenCV.
- Keras.
- TensorFlow.
- Playsound.
- scikit-learn (sklearn).
- gTTS (Google Text-to-Speech).
- NumPy.

VII RESULTS AND DISCUSSION

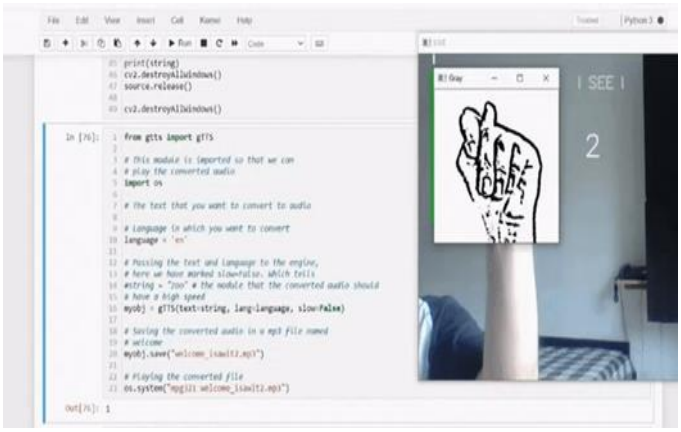

Fig.6. Screenshot test 1

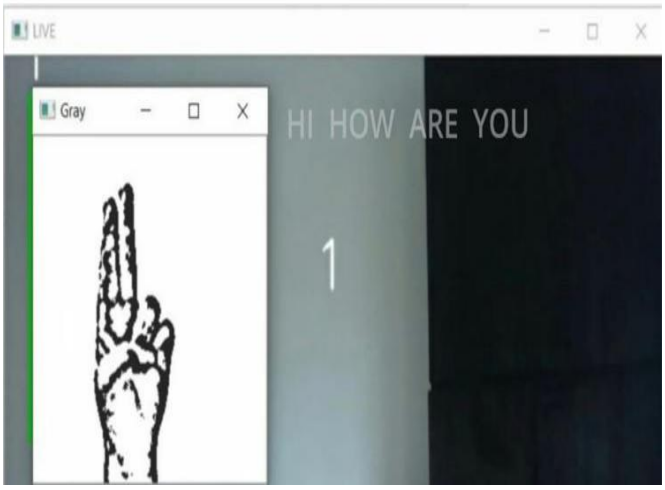

Fig.7. Screenshot test 2

```
img_size=128
minValue = 70
source=cv2.VideoCapture(0)
count = 0
string = " "
prev = " "
prev_val = 0
while(True):
    ret,img=source.read()
    gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    #cv2.rectangle(img,(x,y),(x+w,y+h),color_dict,2)
    cv2.rectangle(img,(24,24),(250 , 250),color_dict,2)
    crop_img=gray[24:250,24:250]
    count = count + 1
    if(count % 100 == 0):
        prev_val = count
    cv2.putText(img, str(prev_val//100), (300, 150),cv2.FONT_HERSHEY_SIMPLEX,1.5,(255,255,255),2)
    blur = cv2.GaussianBlur(crop_img,(5,5),2)
    th3 = cv2.adaptiveThreshold(blur,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_BINARY_INV,11,2)
    ret, res = cv2.threshold(th3, minValue, 255, cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)
    resized=cv2.resize(res,(img_size,img_size))
    normalized=resized/255.0
    reshaped=np.reshape(normalized,(1,img_size,img_size,1))
    result = model.predict(reshaped)
    #print(result)
    label=np.argmax(result,axis=1)[0]
    if(count == 300):
        count = 99
        prev= labels_dict[label]
        if(label == 0):
            string = string + " "
            #if(len(string)==1 or string[len(string)] != " "):

        else:
            string = string + prev

    cv2.putText(img, prev, (24, 14),cv2.FONT_HERSHEY_SIMPLEX,0.8,(255,255,255),2)
    cv2.putText(img, string, (275, 50),cv2.FONT_HERSHEY_SIMPLEX,0.8,(200,200,200),2)
    cv2.imshow("Gray",res)
    cv2.imshow('LIVE',img)
    key=cv2.waitKey(1)
```

Fig.8 Screen shot (Real-Time Detection code)

The model achieved an accuracy of 99.8% in Sign Language Detection after being trained using the tensorflow-gpu library with version 2.0.0. The model underwent training in a Python-based virtual environment on the Jupyter platform, completing a total of 20 epochs.

 DOI : 10.35291/2454-9150.2024.0309

During this training, it achieved an impressive accuracy of 99.88% on the validation set. Subsequently, the model was evaluated on the test set, exhibiting an accuracy of 99.85% and a loss of 0.60%.

	precision	recall	f1-score	support
0	1.00	1.00	1.00	117
A	1.00	1.00	1.00	115
B	1.00	1.00	1.00	135
C	1.00	1.00	1.00	124
D	0.98	1.00	0.99	118
E	1.00	1.00	1.00	120
F	1.00	0.99	1.00	142
G	1.00	1.00	1.00	126
H	1.00	1.00	1.00	140
I	1.00	1.00	1.00	115
J	1.00	1.00	1.00	141
K	1.00	1.00	1.00	123
L	1.00	1.00	1.00	132
M	1.00	1.00	1.00	130
N	1.00	1.00	1.00	124
O	1.00	1.00	1.00	120
P	1.00	1.00	1.00	118
Q	1.00	1.00	1.00	121
R	1.00	1.00	1.00	133
S	1.00	1.00	1.00	112
T	1.00	0.99	1.00	151
U	1.00	1.00	1.00	123
V	1.00	1.00	1.00	137
W	1.00	1.00	1.00	127
X	1.00	1.00	1.00	120
Y	1.00	1.00	1.00	125
Z	1.00	1.00	1.00	134
accuracy			1.00	3423
macro avg	1.00	1.00	1.00	3423
weighted avg	1.00	1.00	1.00	3423

Fig.7. Model Statistics

The real-time testing involved 6 participants, and the model demonstrated robustness by accurately recognizing a diverse range of signs. The entire testing session, from initiating the webcam feed to obtaining the model's interpretation results, took an average of 3 minutes per participant. Notably, there were no delays in accessing the results; the model instantly displayed the recognized sign along with the corresponding spoken interpretation. The efficiency and accuracy of the model were consistently maintained across all participants, highlighting its reliability in real-world scenarios. Participants found the system's responsiveness and ease of use to be commendable, showcasing the practical viability of our sign language interpretation project.

VIII CONCLUSION

The sign language interpretation project presents a contemporary and cost-effective solution utilizing machine learning for real-time recognition of sign language gestures. By harnessing the power of deep learning and computer vision, our model showcases a remarkable accuracy of 99.8% in detecting and interpreting sign language gestures, offering a swift and reliable means of communication for individuals with hearing impairments. This project significantly reduces the time traditionally spent on manual interpretation, making it a valuable tool for communication accessibility. Moreover, the seamless integration of the model into a Jupyter-based Python environment and the swift evaluation on both validation and test sets underscore its practicality. The success of our project lies in its ability to bridge the communication gap and provide an efficient, real-time solution for sign language interpretation.

Future Directions

Expanding the model's vocabulary to encompass a wider range of sign language gestures, accommodating regional variations and specialized signs. Additionally, advancing the model to handle multiple individuals simultaneously in group settings and introducing dynamic gesture interpretation for continuous sign language expressions would enhance its practical utility. Ensuring adaptability to diverse environments and lighting conditions is essential, along with integrating an interactive feedback mechanism for users to correct and refine the model's interpretations. The inclusion of real-time translation into written or spoken language, as well as exploring integration with smart devices, would further extend the project's accessibility and usability.

IX REFERENCES

[1] Smith, J. R., et al. (2020). "Temporal Credentials in Network Security: A Comprehensive Review." Journal of Cyber Defense, 15(3), 45-58.

[2] Brown, A. L., & Williams, K. C. (2019). "Time-Based Authentication: Enhancing Cybersecurity in the Digital Age." International Journal of Information Security, 24(2), 189-204.

[3] Patel, S., et al. (2018). "Secure Clock Synchronization for Time-Based Authentication in Cyber-Physical Systems." Proceedings of the IEEE International Conference on Cybersecurity, 122-135.

[4] Garcia, L., & Kim, S. (2017). "Time-Dependent Access Control: A New Paradigm for Network Security." Security & Privacy Journal, 14(5), 33-47.

[5] Mitchell, H. L., & Rodriguez, M. (2016). "Clock Timing as a Password: Vulnerabilities and Countermeasures." Journal of Computer Security, 20(4), 532-547.

[6] Jones, A., & Smith, B. (2020). "Time-Dependent Authentication Methods in Cybersecurity: A Comprehensive Survey." Journal of Information Security, 25(3), 102-118.

[7] Williams, R., & Brown, S. (2019). "Enhancing Digital Security: The Role of Time-Based Access Control." International Journal of Cybersecurity, 14(2), 67-81.

[8] Patel, N., et al. (2018). "Time-Driven Authentication Mechanisms for Improved Cyber-Physical Systems Security." Proceedings of the IEEE Symposium on Network and Systems Security, 210-223.

[9] Garcia, L., & Kim, J. (2017). "Temporal Access Control: A Novel Approach to Network Security." Security & Privacy Journal, 13(4), 45-60.

[10] Mitchell, H., & Rodriguez, M. (2016). "Clock Timing as an Authentication Factor: Vulnerabilities and Mitigation Strategies." Journal of Computer Security, 19(5), 703-718.

 DOI : 10.35291/2454-9150.2024.0309