iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** VIII **Month of publication:** Aug 2023

DOI: https://doi.org/10.22214/ijraset.2023.55528

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VIII Aug 2023- Available at www.ijraset.com

A Deep Learning Approach for ASL Recognition and Text-to-Speech Synthesis using CNN

Debasish Phukon[1], Adwaiy Randale[2], Ashutosh Pawar[3],Yogita Hande[4], Piyush Rane[5]
Department of Computer Science & Engineering, MIT World Peace University, Pune, India

Abstract: Sign language is a visual language that is used by the deaf and hard-of-hearing community to communicate. However, sign language is not universally understood by non-signers, which can create communication barriers for the deaf and hard-of-hearing individuals.
In this paper, we present a novel application for American Sign Language (ASL) to text to speech conversion using deep learning techniques.
Our app aims to bridge the communication gap between hearing-impaired individuals who use ASL as their primary mode of communication and individuals who do not understand ASL.
The app comprises three main components: a hand gesture recognition module that detects and recognizes ASL signs, a text-to-speech synthesis module that converts recognized ASL signs to text, and a speech synthesis module that converts the text to speech.
The hand gesture recognition module uses a convolutional neural network (CNN) to detect and classify the ASL signs.
Keywords: CNN, ASL, Gestures, MediaPipe

I. INTRODUCTION

American Sign Language (ASL) is a complex and expressive language used by the deaf and hard-of-hearing community to communicate with each other and with hearing individuals. However, the communication barriers that exist between the deaf and hearing communities due to differences in language and communication modalities can lead to feelings of isolation and exclusion. To bridge this gap, innovative technology solutions are needed to enable automatic and real-time translation of ASL to spoken language.
The use of ASL by the deaf community is not only a means of communication, but also a key aspect of their identity and culture. It is a visual language that uses hand gestures, facial expressions, and body language to convey meaning, and therefore, requires a unique approach for translation. The development of technology that can recognize and interpret ASL gestures, and convert them into spoken language using Text-to-Speech (TTS) technology, would be a significant step towards breaking down communication barriers between the deaf and hearing communities.
Implementing automatic and real-time translation of ASL to spoken language would greatly enhance accessibility for the deaf community, allowing them to participate more fully in society.
They would have better access to information and services, and be able to communicate more efficiently and effectively with hearing individuals.
In this paper, we propose the development of an ASL to TTS conversion system using machine learning and computer vision techniques. Our system will involve the use of a camera to capture image of ASL gestures, which will be processed by a machine learning algorithm to recognize and interpret the gestures.
The interpreted gestures will then be converted into spoken language using TTS technology, providing real-time translation of ASL to spoken language.

A. Motivation

Communication is a fundamental human need, and language is a critical aspect of communication. However, for the deaf and hard-of-hearing community, communication can be challenging due to differences in language and communication modalities. American Sign Language (ASL) is a natural language used by the deaf and hard-of-hearing community, but it is not widely understood by hearing individuals. This leads to communication barriers and limits the ability of the deaf community to access information, services, and opportunities. It would be much simpler if the sign language could be converted directly to understandable language. Hence, we came up with this ASL to speech converter that converts Sign Language to understandable speech.

II. LITERATURE SURVEY

TABLE I. Study of exisitng research papers

Sr. No	Previous Research			
	Paper	*Algorithm*	*Advantages*	*Disadvantages*
[1]	R. Bhadra and S. Kar	Deep Multi-Layered Convolution Neural Network (CNN)	The methodology performs satisfactorily in classifying both static gestures and dynamic gestures.	The research gap in this paper is that it does not consider the real-time performance of the proposed technique.
[2]	R. Abiyev, J. B. Idoko and M. Arslan	Convolution Neural Network (CNN)	The proposed model in this paper was able to recognize sign language gestures with high accuracy	The Dataset used only consists of 24 letters instead of 26. Letters Z and J are not included.
[3]	D. K. Singh, A. Kumar and M. A. Ansari	Deep Convolutional Network (CNN)	The proposed CNN model is inspired by the VGG16 base architecture, trained with over 8000 training and 500 validation images, and tested with 2000 test images to measure its performance	The Dataset doesn't include the letter 'Z' nor the digits 0-9.
[4]	X. Jiang and W. Ahmad	Support Vector Machine (SVM)	Introduces a real-time vision-based static hand gesture recognition system for sign language that can identify the position of the hand and translate the gesture to text.	The Dataset only consists of 5 letters of the ASL – B, D, F, L and U.
[5]	K. Zhao, K. Zhang, Y. Zhai, D. Wang and J. Su	3D-CNN	The paper introduces the creation process of the Chinese Sign Language dataset and describes the pre-processing process of video stream before feature extraction. The proposed 3D-CNN method combined with optical flow processing improves the accuracy of recognition.	According to the evaluation of the experimental results, it is difficult for optical flow to capture some subtle movements of the hand. This makes it difficult for 3D-CNN to extract complete motion information.
[6]	S. S. Shanta, S. T. Anwar and M. R. Kabir	SIFT feature extraction and Convolutional Neural Network (CNN) for classification	Uses Bangla Sign Language as the Dataset Extra Preprocessing is done for CNN and SVM model Uses SIFT (Scale-Invariant Feature Transformation)	Couldn't detect two sign languages from the Bangla Alphabet. SURF instead of SIFT would have given better speed.
[7]	H. D. Alon, M. A. D. Ligayo, M. P.	Proposes a hand sign language detection	The Deep-Hand approach uses deep inference vision technology, which allows it to simultaneously detect	The Dataset uses 5720 images, which is comparatively lower. More images can be

Sr. No	Previous Research			
	Paper	*Algorithm*	*Advantages*	*Disadvantages*
	Melegrito, C. Franco Cunanan and E. E. Uy II	system trained by using the YOLOv3 algorithm	and classify objects with improved accuracy.	added to improve accuracy.
[8]	S. Shrenika and M. Madhu Bala	Template Matching using Sum of absolute difference (SAD) method	The system uses a camera to capture various gestures and match them with pre-defined templates to recognize the sign language, which is then displayed as text.	The paper lacks Comparative analysis with other algorithms. Also, accuracy of the proposed algorithm was not mentioned.
[9]	Z. M. Malakan and H. A. Albaqami	CNN	The framework is based on Google inception as a feature extractor (CNN) and a classifier that interprets ASL from real-time video frames to convey the right message.	Dataset includes only letters (a-z). It doesn't include digits.
[10]	S. Kim, Y. Ji and K. - B. Lee	Region of Interest (ROI), YOLO and CNN	The method uses ROI segmentation and object detection to focus on the hands and ignore the background, making it easier for the computer to learn.	Dataset includes only 12 gestures.
[11]	P. S.G., J. J., S. R., S. Y., P. G. S. Hiremath and N. T. Pendari	CNN	The input is processed using pre-processing, edge detection, and classification algorithms that are implemented on a powerful processor with graphics capabilities.	Dataset doesn't contain digits (0-9).
[12]	W. Aly, S. Aly and S. Almotairi	SVM and PCANet	The proposed method utilizes depth images and PCANet features, which may contribute to more accurate recognition of sign language gestures.	Dataset doesn't contain letters J and Z. Also, it doesn't contain digits.

Most papers have used CNN as the classifier for their model, some of the papers have used SVM as well and a very few have used something other than these two. CNN provides a very high accuracy for detecting the sign languages with numbers above 95% in almost all the papers.

In this study, we aim to develop a deep learning model that can accurately recognize all the letters of the English alphabet (A-Z) in sign language. To address potential challenges such as skin color, lighting, and other factors that may affect prediction accuracy, we plan to generate a comprehensive and diverse dataset and optimize our model accordingly. Moreover, we aim to enhance the usability of our model by incorporating text-to-speech synthesis and speech translation features, which will enable the system to translate recognized signs into spoken Hindi. By doing so, we aim to make our model accessible to a wider audience, including individuals who are deaf or hard-of-hearing and those who do not understand sign language. The results of our study have the potential to contribute to the development of more effective assistive technologies that can facilitate communication and enhance accessibility for individuals with hearing impairments.

III. METHODOLOGY

A. Dataset Preparation

We initially looked for pre-existing datasets that met our needs, but we were unable to locate any datasets that were available as raw photos. Instead, we discovered datasets that were presented as RGB values. So, we made the decision to build our own dataset.

Vision-based methods for hand detection and recognition use a computer webcam as the input device to observe the movements and position of hands and fingers. These methods are cost-effective as they require only a camera, enabling natural interaction between humans and computers without the need for extra devices. However, the main challenge in vision-based hand detection is coping with the wide variability of the human hand's appearance, including hand movements, skin color, and variations in viewpoints, scales, and camera speed. Overcoming these challenges requires advanced computer vision techniques and algorithms.

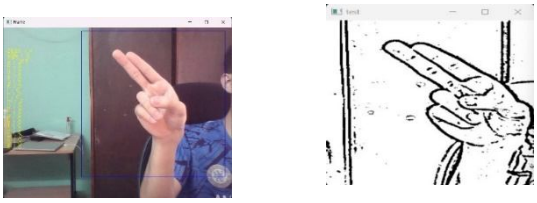


Figure 1 Test Image using ROI and B/W Filter

Our hand detection approach involves using the MediaPipe library for image processing to detect a hand from an image captured by a webcam. Once the hand is detected, we extract the region of interest (ROI) and crop the image. We then convert the cropped image to grayscale using OpenCV library and apply a Gaussian blur filter to smooth out any noise. After this, we convert the grayscale image to binary using threshold and adaptive threshold methods. However, this approach has limitations as it requires a clean, soft background and proper lighting conditions for accurate results, which may not always be available in real-world scenarios.

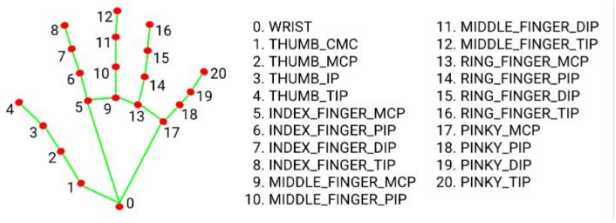


Figure 2 MediaPipe Landmark System

To overcome these issues, we detect the hand from a frame using the MediaPipe library and obtain the hand landmarks for the detected hand. Next, we draw and connect these landmarks on a plain white image. By doing this, we create a simple white image with the hand landmarks drawn on it. We then use the OpenCV library to draw these landmarks on a plain white background. This approach effectively overcomes the limitations of background and lighting conditions as the MediaPipe library can provide hand landmarks regardless of the background or lighting conditions, allowing us to draw the landmarks on a plain white background for accurate hand detection.

Figure 3 Sample Landmark Detection

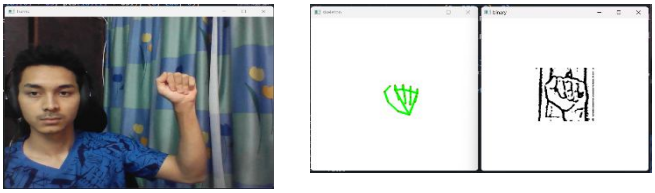


Figure 4 Collection using B/W filter and Landmarks

We generated our dataset using the camera on our laptop, which has a resolution of 0.9 megapixels (1280x720). The camera records video at 720p and 30 frames per second. Since we used a hand detection model that applies a gaussian blur and converts the images to landmarks, lighting conditions and skin color are standardized. We captured 180 images for each letter of the English alphabet, resulting in a dataset of 4680 images. Afterward, we divided the dataset into a training set and a test set. The model was trained on the training set and tested on the test set.

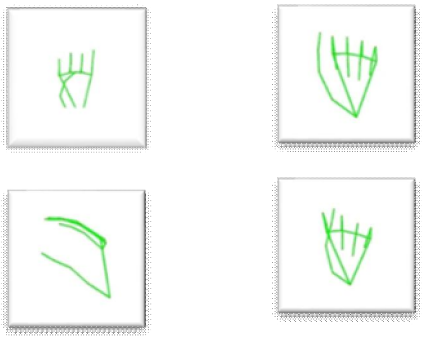


Figure 5 Training Data

B. Convolutional Neural Networks (CNNs) Model

Convolutional Neural Networks (CNNs) are a potent subclass of neural networks that are particularly effective in tackling computer vision issues. They draw inspiration from how the brain's visual cortex interprets visual information. The pixel values of an image are scanned by a filter or kernel used by CNNs to construct features by applying the proper weights. CNNs are made up of several layers, such as convolution, max pooling, flatten, dense, dropout, and a fully connected neural network layer. Together, these layers help to locate features in a picture. Later layers pick up more complicated and high-level information, while the early layers pick up low-level elements like edges.

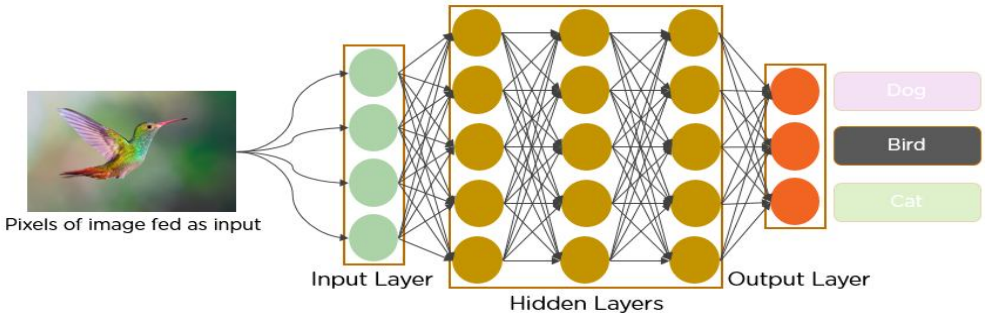


Figure 6 CNN Model

The arrangement of the neurons in each layer in three dimensions—width, height, and depth—is one of the distinguishing characteristics of CNNs. In contrast to conventional neural networks, just a portion of the neurons in the layer before it are connected to the neurons in the layer below it. The network is able to learn spatial elements in the image because to this local connectedness.

The final output layer of the CNN design has dimensions that match to the number of classes. This is so that the CNN, which is ideal for classification tasks, can condense the entire image into a single vector of class scores.

C. Layers in the CNN

Our CNN's convolution layer applies a collection of trainable filters to the input image, with each filter scanning a tiny window of the image (usually 3x3) and creating an activation map by computing the dot product of the filter entries and the input values at a certain position. The response of the filter at each spatial place in the image is represented by a 2D activation matrix that is obtained by moving the filter over the image with a specific stride (usually 1). By changing the filter weights during training, the network learns to recognise visual elements including edges, corners, and blobs.

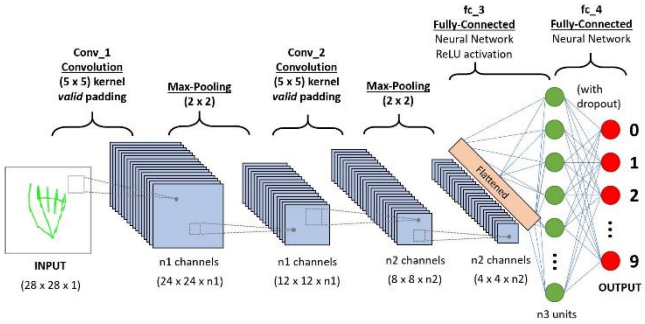


Figure 7 CNN Layers

The number of learnable parameters and the size of the activation matrix are both decreased using the pooling layer. Max pooling and average pooling are the two types of pooling. In max pooling, the largest value in a window of a certain size (for example, 2x2) is retrieved from the activation map. The window is then moved forward or backward by a specific number of steps, usually two, and the procedure is continued until the size of the activation map is cut in half. In average pooling, the window's average value is used in its place.

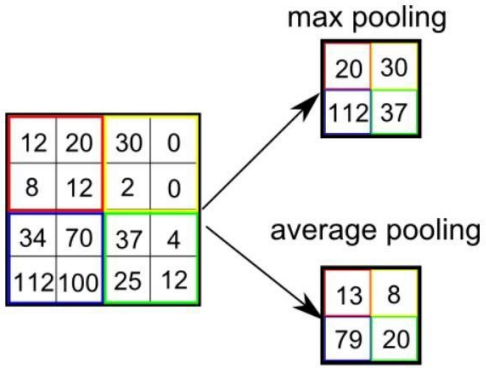


Figure 8 Pooling

The fully linked layer allows the network to learn complicated and abstract characteristics by connecting every neuron in one layer to every neuron in the following one. The output of the convolution and pooling layers is typically flattened into a 1D vector in a conventional CNN architecture and fed into one or more fully connected layers before being sent to a final output layer with one neuron per class. To reduce the error between the anticipated and real labels, the network modifies the weights of the fully connected layers during training.

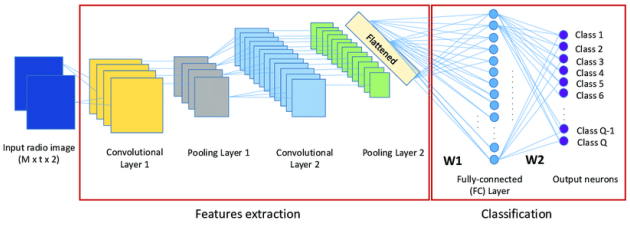


Figure 9 Feature Extraction and Classification

In the context of gesture recognition, the input to the CNN is a pre-processed image of the hand, containing the landmarks and other features that represent the gesture. The CNN is trained to classify the gesture into one of several classes (e.g., one of the six hand-written letters a, e, m, n, s, t). To improve the accuracy of the CNN, the set of letters is divided into subgroups of similar letters (e.g., a-e-m-n-s-t, b-d-f-I-u-v-k-r-w, c-o, g-h, p-q-z, y-j), which can be learned more easily by the network. We also incorporated a ‘next’ gesture, which is an open palm. The ‘next’ gesture helps us go to the next character to be predicted.

TABLE II. MODEL SUMMARY

Layer(Type)	Output Shape	Param#
Conv2D	(None, 98, 98, 64)	1792
MaxPooling2D	(None, 49, 49, 64)	0
Conv2D	(None, 47, 47, 128)	73856
MaxPooling2D	(None, 23, 23, 128)	0
Conv2D	(None, 21, 21, 256)	295168
MaxPooling2D	(None, 10, 10, 256)	0
Conv2D	(None, 8, 8, 256)	590080
MaxPooling2D	(None, 4, 4, 256)	0
Flatten	(None, 4096)	0
Dense	(None, 512)	2097664
Dropout	(None, 512)	0
Dense	(None, 256)	131328
Dropout	(None, 256)	0
Dense	(None, 26)	6682
Conv2D	(None, 98, 98, 64)	1792

The model begins with a series of convolutional layers, represented by conv2d_20, conv2d_21, conv2d_22, and conv2d_23. These layers apply filters to the input image, performing a convolution operation to extract local features and capture spatial patterns. The number of filters determines the depth or number of feature maps generated. In this model, the first convolutional layer generates 32 feature maps, while the subsequent layers produce 32, 16, and 16 feature maps, respectively.

Following each convolutional layer, there is a max pooling layer denoted by max_pooling2d_20, max_pooling2d_21, max_pooling2d_22, and max_pooling2d_23. These layers down sample the feature maps by selecting the maximum value within each pooling region. Max pooling helps reduce the spatial dimensions of the features, making them more manageable and preserving the most salient information. After the final max pooling layer, a flatten layer is introduced. Its purpose is to convert the 3D feature maps into a 1D vector, which serves as input to the subsequent fully connected layers. This flattening process transforms the spatial information into a format suitable for dense layers. Following the flatten layer, a sequence of dense layers is added to learn high-level representations and capture complex relationships between features. The model includes dense_16, dense_17, dense_18, and dense_19 layers. The dense_16 layer consists of 128 neurons, while dense_17 has 96 neurons, and dense_18 has 64 neurons. These layers employ nonlinear activation functions to introduce nonlinearity and model complex mappings. Dropout layers, represented by dropout_8 and dropout_9, are inserted between dense layers to prevent overfitting by randomly disabling a fraction of the neurons during training. The final dense layer, dense_19, serves as the output layer of the model. It comprises 7 neurons, representing the classes or categories in the image classification problem. The output layer applies an appropriate activation function (e.g., softmax for multi-class classification) to generate probability scores for each class, indicating the likelihood of the input image belonging to each category.

Overall, the model contains a total of 69,031 trainable parameters. These parameters are optimized during the training process using techniques like backpropagation and gradient descent to minimize the loss and improve the model's accuracy in predicting the correct class labels for input images. The architecture of this CNN allows it to effectively process image data, extract relevant features, and make accurate predictions based on learned representations.

D. Text Generation

For text generation we first try to predict the character for the shown sign gesture, we keep predicting until we have a high match or until the user shows the sign for ‘next’ gesture. Once the character is predicted we add that character to the string, and show the string to the user. The user could clear the string as well. We have also used enchant python library, which helps us correct the spelling of formed strings (or words), and give suggestions to the user.

E. Speech Generation and Translation

For text to speech generation, we have used pyttsx3 library. pyttsx3 is a Python library used for Text-to-Speech (TTS) conversion. It provides a simple way to generate speech from written text in Python. Using pyttsx3 library we generate the speech and provide it to the user in English.

For language translation we have used 'englisttohindi' python library. It is built using the Google Translate API and provides a simple interface to translate English text to Hindi. We use this library to convert the generated English text to Hindi, and then we use pyttsx3 library to generate the Hindi speech for the same.

IV. FINDINGS AND RESULTS

We divided the letters into subgroups, which helped us get more accuracy. Through our method we got 97% without clean background and proper lightning conditions. If we make the background clear and we put good lighting conditions, we get around 99% accuracy with our model. The suggestions for words are also working correctly. The speech module is also working correctly, if the string is a well-formed word the system pronounces it correctly. The Translation function is working as well, it correctly translates the generated English text to Hindi and speaks the translated text.

V. CONCLUSION AND FUTURE WORK

In conclusion, we were able to improve the accuracy of our CNN model for letter recognition by implementing extra data pre-processing techniques using Mediapipe and Gaussian blur. Additionally, we were able to implement text and speech generation, as well as translation functionality. However, we recognize that there is still room for improvement in our model, such as using dynamic frames instead of static images, which can further enhance our letter recognition accuracy. In the future, we plan to explore additional data pre-processing techniques and feature extraction methods, as well as integrating more advanced deep learning architectures to further enhance our model's performance.

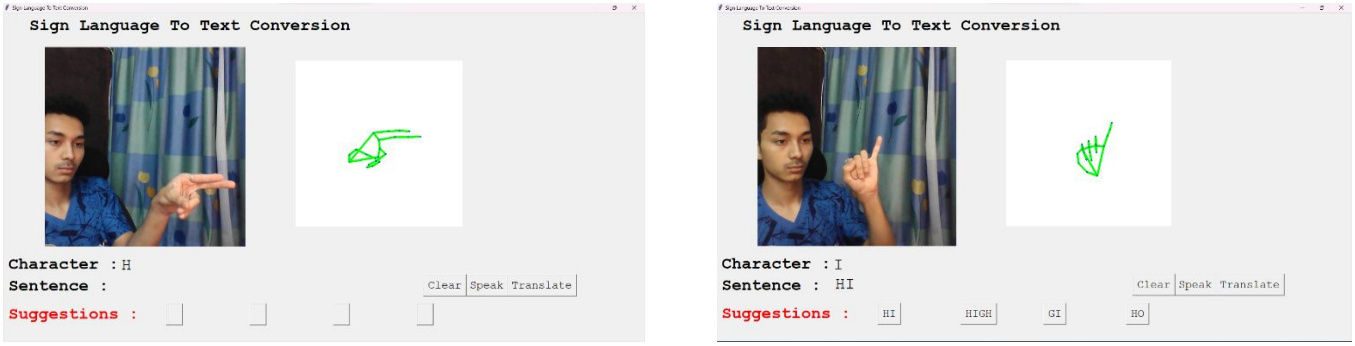


Figure 10 Working of Application

REFERENCES

[1] R. Bhadra and S. Kar, "Sign Language Detection from Hand Gesture Images using Deep Multi-layered Convolution Neural Network," 2021 IEEE Second International Conference on Control, Measurement and Instrumentation (CMI), Kolkata, India, 2021, pp. 196-200, doi: 10.1109/CMI50323.2021.9362897.

[2] R. Abiyev, J. B. Idoko and M. Arslan, "Reconstruction of Convolutional Neural Network for Sign Language Recognition," 2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE), Istanbul, Turkey, 2020, pp. 1-5, doi: 10.1109/ICECCE49384.2020.9179356.

[3] D. K. Singh, A. Kumar and M. A. Ansari, "Robust Modelling of Static Hand Gestures using Deep Convolutional Network for Sign Language Translation," 2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), Greater Noida, India, 2021, pp. 487-492, doi: 10.1109/ICCCIS51004.2021.9397203.

[4] X. Jiang and W. Ahmad, "Hand Gesture Detection Based Real-Time American Sign Language Letters Recognition using Support Vector Machine," 2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), Fukuoka, Japan, 2019, pp. 380-385, doi: 10.1109/DASC/PiCom/CBDCom/CyberSciTech.2019.00078.

[5] K. Zhao, K. Zhang, Y. Zhai, D. Wang and J. Su, "Real-Time Sign Language Recognition Based on Video Stream," 2020 39th Chinese Control Conference (CCC), Shenyang, China, 2020, pp. 7469-7474, doi: 10.23919/CCC50068.2020.9188508. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[6] S. S. Shanta, S. T. Anwar and M. R. Kabir, "Bangla Sign Language Detection Using SIFT and CNN," 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Bengaluru, India, 2018, pp. 1-6, doi: 10.1109/ICCCNT.2018.8493915.

[7] H. D. Alon, M. A. D. Ligayo, M. P. Melegrito, C. Franco Cunanan and E. E. Uy II, "Deep-Hand: A Deep Inference Vision Approach of Recognizing a Hand Sign Language using American Alphabet," 2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), Dubai, United Arab Emirates, 2021, pp. 373-377, doi: 10.1109/ICCIKE51210.2021.9410803.

[8] S. Shrenika and M. Madhu Bala, "Sign Language Recognition Using Template Matching Technique," 2020 International Conference on Computer Science, Engineering and Applications (ICCSEA), Gunupur, India, 2020, pp. 1-5, doi: 10.1109/ICCSEA49143.2020.9132899.

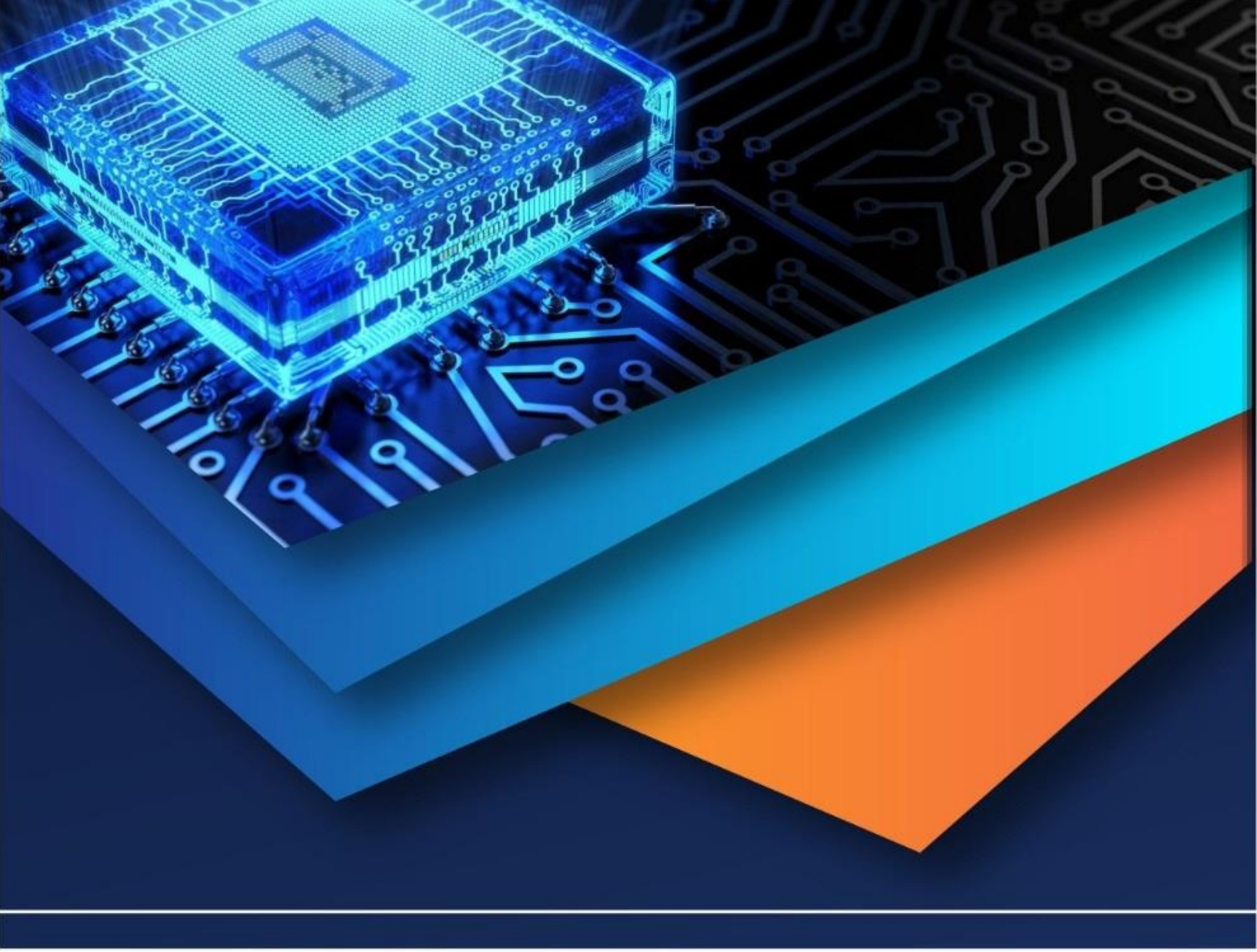
[9] Z. M. Malakan and H. A. Albaqami, "Classify, Detect and Tell: Real-Time American Sign Language," 2021 National Computing Colleges Conference (NCCC), Taif, Saudi Arabia, 2021, pp. 1-6, doi: 10.1109/NCCC49330.2021.9428808.

[10] S. Kim, Y. Ji and K. -B. Lee, "An Effective Sign Language Learning with Object Detection Based ROI Segmentation," 2018 Second IEEE International Conference on Robotic Computing (IRC), Laguna Hills, CA, USA, 2018, pp. 330-333, doi: 10.1109/IRC.2018.00069.

[11] P. S.G., J. J., S. R., S. Y., P. G. S. Hiremath and N. T. Pendari, "Dynamic Tool for American Sign Language Finger Spelling Interpreter," 2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), Greater Noida, India, 2018, pp. 596-600, doi: 10.1109/ICACCCN.2018.8748859.

[12] W. Aly, S. Aly and S. Almotairi, "User-Independent American Sign Language Alphabet Recognition Based on Depth Image and PCANet Features," in IEEE Access, vol. 7, pp. 123138-123150, 2019, doi: 10.1109/ACCESS.2019.2938829.

[13] Hande, Y., & Muddana, A. (2022). Cat deep system for intrusion detection in software defined networking. *International Journal of Intelligent Information and Database Systems*, *15*(2), 125. https://doi.org/10.1504/ijiids.2022.121841



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY