TELKOMNIKA Telecommunication, Computing, Electronics and Control
Vol. 22, No. 6, December 2024, pp. 1459∼1468
ISSN: 1693-6930, DOI: 10.12928/TELKOMNIKA.v22i6.26498

Real time Indian sign language recognition using transfer learning with VGG16

Sumit Kumar[1], Ruchi Rani[2], Sanjeev Kumar Pippal[3], Ulka Chaudhari[2]
[1]Symbiosis Institute of Technology, Pune Campus, Symbiosis International (Deemed University), Pune, Maharashtra, India
[2]Department of Computer Science and Engineering, School of Computer Engineering and Technology, Dr. Vishwanath Karad MIT World Peace University, Pune, Maharashtra, India
[3]GL Bajaj Institute of Technology and Management, Greater Noida, Uttar Pradesh, India

Article Info

Article history:

Received Jul 24, 2024
Revised Aug 23, 2024
Accepted Sep 20, 2024

Keywords:

Convolutional neural networks
Pre-trained models
Real-time
Sign language
VGG16 model

ABSTRACT

Normal people's interaction and communication are easier than those with disabilities such as hearing and speech, which are very complicated; hence, the use of sign language plays a crucial role in bridging this gap in communication. While previous attempts have been made to solve this problem using deep learning techniques, including convolutional neural networks (CNNs), support vector machine (SVM), and K-nearest neighbours (KNN), these have low accuracy or may not be employed in real time. This work addresses both issues: improving upon prior limitations and extending the challenge of classifying characters in Indian sign language (ISL). Our system, which can recognize 23 hand gestures of ISL through a purely camera-based approach, eliminates expensive hardware like hand gloves, thus making it economical. The system yields an accuracy of 97.5% on the training dataset, utilizing a pre-trained VGG16 CNN optimized by the Adam optimizer and cross-entropy loss function. These results clearly show how effective transfer learning is in classifying ISL and its possible real-world applications.

Corresponding Author:

Sumit Kumar
Symbiosis Institute of Technology, Pune Campus, Symbiosis International (Deemed University)
Pune 412115, Maharashtra, India
Email: er.sumitkumar21@gmail.com

1. INTRODUCTION

Good communication is essential for human interaction; this allows us to share various ideas and build relationships, thus improving understanding. Sign language plays an important role in deaf and hard-hearing communities, as sign language is the main means of communication [1]. These problems may limit these people from good learning opportunities and cause them to miss important information like news and announcements. These automatic sign language systems can help people communicate with other people, thus reducing the communication gap between them. How would it feel to live in a world where we can communicate with every person regardless of their disabilities? Automatic sign language recognition is a way to break the communication barriers to a more friendly society [2]. Various works have been presented in deep learning [3], especially in convolutional neural networks (CNNs) [4], [5]. These algorithms perform remarkably in different tasks, from image classification to object detection to image segmentation and generation. CNNs are good at recognizing patterns, which makes them useful for applications like sign language recognition [6], [7]. For this sign language recognition, a large amount of data is needed for training, which is where the primary problem occurs. Such datasets are limited; moreover, finding a standard sign language dataset for a specific sign language type, like Indian sign language (ISL) or American sign language, is challenging [8].

This research uses a deep learning technique called transfer learning, a proper learning technique to achieve good results for automatic sign language classification for ISL. Transfer learning focuses on the knowledge acquired by a pre-trained model on a large dataset and uses it for a new task with a limited dataset. The pre-trained model is a CNN called VGG16 [9]. CNNs are known for their better performance in image recognition tasks. VGG16, a pre-trained model on a large image dataset like ImageNet, has already learned many features for identifying patterns and shapes within an image [10]. By using this pre-trained model, the research aims to achieve a better way for ISL classification without needing to train a CNN from scratch on a smaller ISL dataset, which will lead to poor accuracy and poor classification. This approach is better than training a model from scratch as training complex CNNs [11], [12] requires extensive and better computational resources and a lot of labeled data, both of which can be limited for specific sign languages like ISL.

This system can be helpful in various fields. Sign language recognition systems in educational institutions can create a more friendly and understandable environment for students [13]. What if classrooms were installed with real-time translation between sign and spoken language, thus improving student and teacher communication? This makes ordinary people not only understand sign language using people but also being able to help those people further wherever needed. Deaf students can access lectures, participate in discussions more appropriately, and compete with other ordinary people. This helps normal people learn how to use sign language and deaf culture. Additionally, these systems can be used to create educational resources for deaf learners, such as sign language tutorials or educational videos with sign language understanding. Sign language recognition can, however, increase communication in public spaces like hospitals, airports, and news broadcasts. Hospitals can utilize these systems to clearly and adequately understand communication during consultations and emergencies. Airports can provide real-time sign language translation for announcements and flight information, helping navigation for deaf and hard-of-hearing travelers. News broadcasts can use this sign language interpretation alongside spoken news, providing equal access to information for the deaf community. These applications can help deaf and hard-of-hearing individuals facing challenges navigate public services confidently and independently. Different cross-lingual sign language recognition [14] can also help these people communicate with similar people with disabilities from various places.

Section 1 represents a brief introduction and section 2 iterates through the short literature review report with different types of tables proposed in the literature. Section 3 outlines the proposed methodology and design. Section 4 analyzes and discusses performance evaluation. Finally, section 5 is the conclusion, briefly summarizing the proposed model.

2. LITERATURE REVIEW

Mohan *et al.* [15] researched the ISL character recognition system. The researchers used a dataset with 33 characters that excluded specific characters like '1' and '2' since they use the same sign as the letters. They have at least 600 images of each of the characters. Their training ratio is 8:1:1. Their system then performs skin segmentation, in which if the max value passes a threshold, that value is used as a parameter else. It is termed as zero. They pass those images through the thresholding and blue function, and then the threshold is used as a mask to generate the results. Every Image is converted using the Keras model and resized to a range of 255. Image arrays are generated and initially hold a shape of (64,64,3). Data augmentation is used to increase the variance in the images. The researchers have trained a CNN model with 20 epochs and a batch size of 500. These models were experimented with to obtain good accuracy by the researchers. A team of other researchers where a gesture is used as an input to the system [16]. Their dataset consisted of both word-level gestures and finger spellings. They have recognized 17 letters of the English characters. Segmentation is performed on the dataset based on color to detect the sign's shape. A binary image is created further. Neural networks and support vector machine (SVM) models are used for classification. The researchers have used an artificial neural network and SVM presented by the researchers [16]. Finally, the artificial neural network classifies the sign with an average of 94.3% and the SVM classifier uses 92.12% accuracy. These researchers present sign language recognition in a specific language, ISL [17]. It identifies 33 hand gestures and poses from the ISL. The system uses a camera to capture the images, and then the frames are transferred to a remote server for further processing. These researchers have used techniques like face detection, object stabilization, and skin color segmentation for hand detection and tracking. The tracking of the hand is performed in each image frame. The features, once recognized, are sent to the classifier to recognize. Thus, the output is sent back to the android device after recognition. The gestures are classified using the hand poses followed by their immediate motion. Then, the model uses Hidden Markov model chins for gesture classification according to the 12 previously defined in the ISL. After the hand detection, the poses are classified using the K-nearest neighbors (KNN) algorithm [17]. Using this method, the accuracy achieved is 99.7% for static hand poses and 97.23% for gesture recognition. The researchers in the paper [18] implement a solution for sign language recognition. They used a camera as input, displaying the detected output in the system. The system uses a dataset of about 600 training images and 80 testing images for each number. The photos are in a grayscale format with 28x28. The activation functions used are Relu and SoftMax. Ten epochs were calculated with its training accuracy and validation accuracy. The model received about 95% accuracy for their model. The dataset used by these researchers is self-made, with 7800 images [19]. The images are rescaled to 80x60 pixels. They have used CNN. They have also used activation functions like Relu and SoftMax. The accuracy achieved is 99%, which could be an overfitting model. These researchers use YOLOv3 [20] for object detection and darknet-53 CNN [21]. Their system was tested with 16 signs for image and 7 video inputs. Their accuracy for static and dynamic signs is 95.7% and 93.1%. These researchers divided 10400 images with 26 classes into 4:1 ratios, with approximately 400 images for each class [22]. They have used pre-trained models like VGG16, VGG19, ResNet, and DenseNet with Inception V3 and MobileNet V2. They have received 99% for the training dataset and 83% for the test set for the VGG16 model [23]-[25].

3. METHOD

The critical steps in the proposed sign language recognition system are some preprocessing steps for images, namely, noise reduction and resizing, that make the dataset ready to be analyzed effectively and efficiently. The image is then resized to 224x224 pixel. The input size needed by the pre-trained VGG16 is utilized for feature extraction. The convolution layers of VGG16 recognize and encode the important features of hand gestures. These features are further used in the final step of the sign classification process, which takes the help of a pre-trained network to make the right sign prediction.

3.1. Image processing

Image preprocessing is crucial before leveraging images for applications such as sign language recognition, as seen in Figure 1. As described, data must be cleaned and arranged appropriately for analysis. This is also interpreted to mean image preprocessing, in which the images are appropriately formatted for use in the computer vision algorithm. This may range from removing background noise to resizing images. Preprocessing is vital since it improves the quality of input data so that the model will concentrate on the right features; it reduces unnecessary noise in data such that, given such noise reduction, the model can learn more efficiently toward better recognition accuracy.

3.2. Normalization

Images are pixel grids, where each pixel carries a numerical value to describe intensity and color, which changes with the type of image format. CNNs face technical challenges when processing raw data in such forms that are essentially pixel values but inconsistent, not to forget problems in activation functions. Thus, normalization is scaled by bringing the pixel values into a standard range. In this particular case, the function ToTensor() takes the pixel values that range from 0 to 255 for typical grayscale images and thus makes a conversion into the desired PyTorch tensor between 0 and 1.

3.3. Resizing

Without size resizing, images tend to increase computational complications while training CNNs, such as increased complexity and difficulty in finding patterns. To overcome this, images are resized to have the same dimensions. The images are resized to 224x224 pixels in this system. Such a size is used as the images have to be sized this way to fit the requirement of the pre-trained VGG16 architecture. The same will be compatible with the pre-trained model if its dimensions are kept the same.

3.4. Feature extraction

Feature extraction acts as a bridge between the preprocessed video and recognizing the signs. Imagine the system looking at a video frame of a hand gesture. Feature extraction, powered by a CNN, is like showing the system a detailed analysis of that frame. CNN doesn't just look at the overall image; it automatically identifies key aspects like finger shapes, hand location, and orientation. These become the ”features” - a compressed and informative summary of the hand gesture. This allows CNN to learn the essence of each sign, preparing the system for the final stage where it can accurately classify the sign language symbol. VGG16’s core functionality lies in its convolutional layers. These layers perform convolutions, essentially sliding learned filters across the image. These filters activate based on specific patterns they encounter in the image data. The filters progressively extract features as the image data progresses through these layers. They identify low-level features like edges, lines, and basic shapes. Subsequent layers then build upon these, combining them into more complex features. This process continues, allowing the network to learn highly relevant features for sign language recognition.

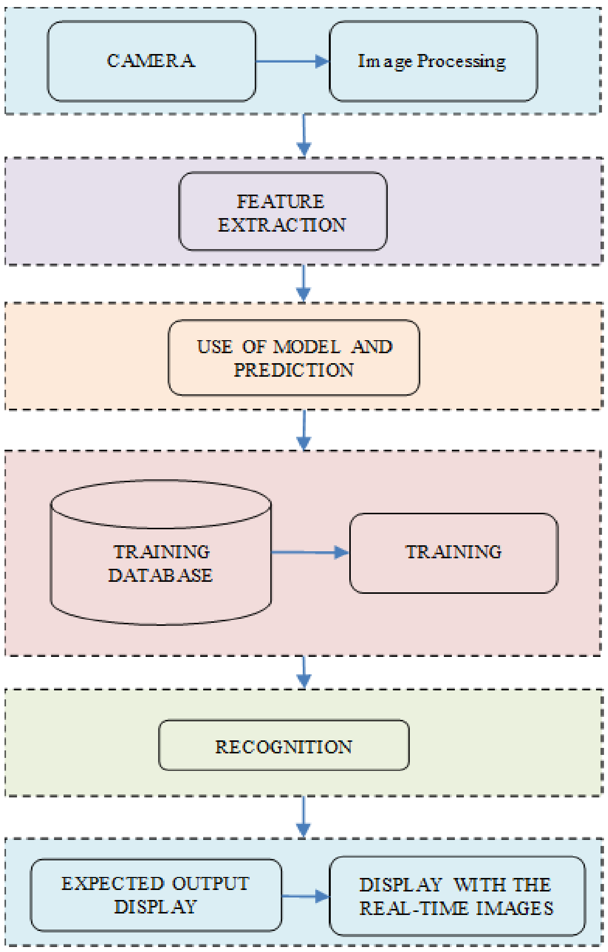

Figure 1. Process flow of the system

3.5. Sign classification

In the final stage, sign classification, the system leverages a pre-trained neural network like VGG16. This network inputs the feature vector, encoding the hand gesture’s key characteristics. During training, VGG16 learned to associate specific patterns within these feature vectors with particular signs from your dataset. Now, VGG16 analyzes the current frame’s feature vector and assigns a probability score to each sign in its vocabulary. The system then outputs the sign with the highest score as the recognized sign, representing its best hand gesture interpretation based on its learned features.

3.6. Dataset details

A public dataset is downloaded from Kaggle (https://www.kaggle.com/datasets/soumyakushwaha/indian-sign-language-dataset) with 23 classes for training and testing. The signs in the dataset are for the alphabet: A, B, C, D, E, F, G, I, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, and Z. To simulate the actual scenario of communication, Gaussian noise was added to every image with blurred and cluttered, colorful backgrounds. The dataset consists of 702 images, each having a resolution of 126x126 pixels. Figures 2 and 3 represent some examples from the dataset. Of the 702 images, 600 images were randomly selected for training, while the remaining 102 images, approximately 14.5% of the dataset, were used for validation.

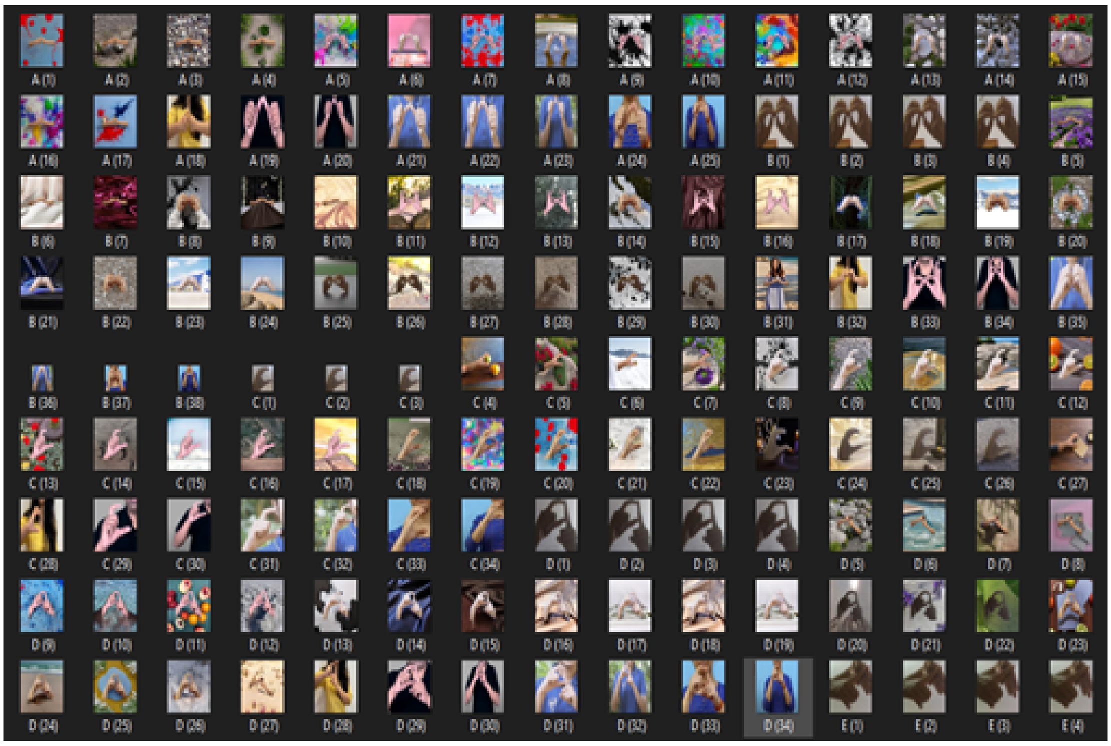
A (1) A (2) A (3) A (4) A (5) A (6) A (7) A (8) A (9) A (10) A (11) A (12) A (13) A (14) A (15)
A (16) A (17) A (18) A (19) A (20) A (21) A (22) A (23) A (24) A (25) B (1) B (2) B (3) B (4) B (5)
B (6) B (7) B (8) B (9) B (10) B (11) B (12) B (13) B (14) B (15) B (16) B (17) B (18) B (19) B (20)
B (21) B (22) B (23) B (24) B (25) B (26) B (27) B (28) B (29) B (30) B (31) B (32) B (33) B (34) B (35)
B (36) B (37) B (38) C (1) C (2) C (3) C (4) C (5) C (6) C (7) C (8) C (9) C (10) C (11) C (12)
C (13) C (14) C (15) C (16) C (17) C (18) C (19) C (20) C (21) C (22) C (23) C (24) C (25) C (26) C (27)
C (28) C (29) C (30) C (31) C (32) C (33) C (34) D (1) D (2) D (3) D (4) D (5) D (6) D (7) D (8)
D (9) D (10) D (11) D (12) D (13) D (14) D (15) D (16) D (17) D (18) D (19) D (20) D (21) D (22) D (23)
D (24) D (25) D (26) D (27) D (28) D (29) D (30) D (31) D (32) D (33) D (34) E (1) E (2) E (3) E (4)

Figure 2. Data for training

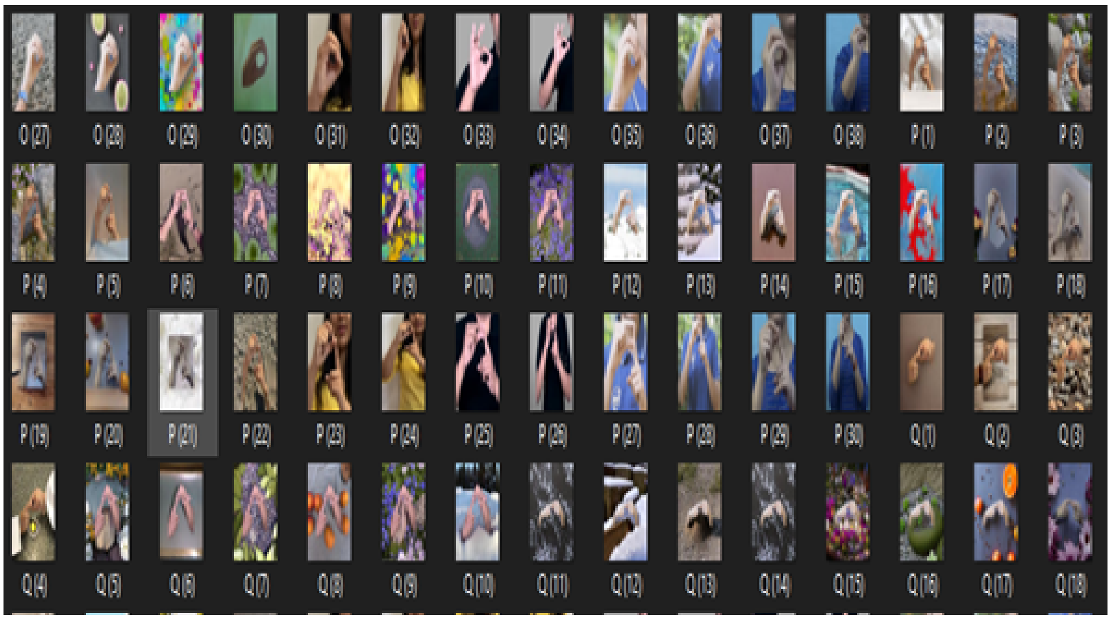
O (27) O (28) O (29) O (30) O (31) O (32) O (33) O (34) O (35) O (36) O (37) O (38) P (1) P (2) P (3)
P (4) P (5) P (6) P (7) P (8) P (9) P (10) P (11) P (12) P (13) P (14) P (15) P (16) P (17) P (18)
P (19) P (20) P (21) P (22) P (23) P (24) P (25) P (26) P (27) P (28) P (29) P (30) Q (1) Q (2) Q (3)
Q (4) Q (5) Q (6) Q (7) Q (8) Q (9) Q (10) Q (11) Q (12) Q (13) Q (14) Q (15) Q (16) Q (17) Q (18)

Figure 3. Dataset used for the testing of the system

3.7. Model details

In the sign language recognition system, VGG16 forms the main framework both for feature extraction and classification. This pre-trained CNN model contains convolutional layers, which apply small filters across images to detect the patterns and features. During training, the large dataset was given to VGG16, starting from simple features such as edges and lines which formed more complex patterns. As seen, VGG16 has interleaved pooling layers between its convolutional layers, as shown in Figure 4. These pooling layers shrink the sizes of the images while maintaining essential features. Consequently, the resulting feature vector captures the unique characteristics of the image.

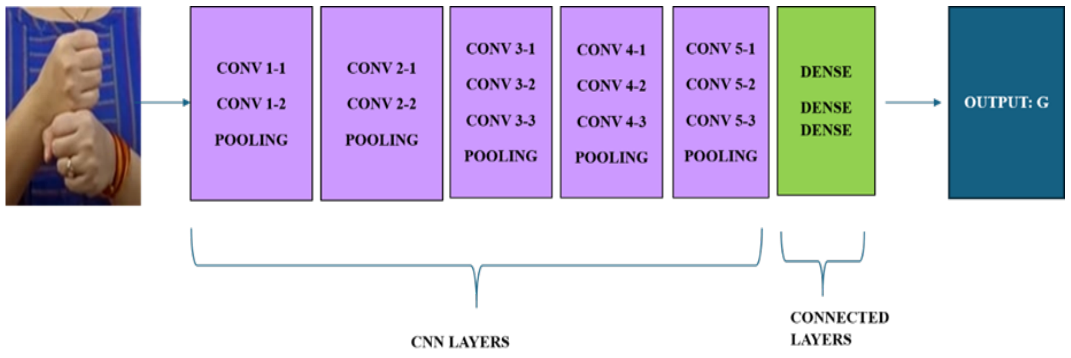

Figure 4. VGG16 convolution layers architecture

Activation functions like ReLU are critical in helping the model learn complex patterns, process data efficiently, and make accurate predictions. ReLU, one of the most widely used activation functions in the hidden layers of neural networks, is computationally less expensive than alternatives like tanh and sigmoid. Another key activation function used in the output layers is SoftMax, as described in (1). SoftMax is particularly useful for classifying images, as it handles multiple classes by assigning probabilities to each.

$$f(x) = \text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}} \tag{1}$$

Here, z represents the raw outputs of the neural network, and $e \approx 2.718$. The ith entry in the SoftMax output vector corresponds to the predicted probability of the input belonging to class i. The model is compiled using Adam optimization and cross-entropy loss. During training, the model undergoes 10 epochs, tracking metrics like training accuracy, training loss, validation accuracy, and validation loss, as seen in Table 1. The model undergoes 10 epochs during training. These metrics provide insights into the model's learning progress.

Table 1. Evaluation metrics

Epoch	Train loss	Valid loss	Train accuracy (%)	Valid accuracy (%)
0	3.18	3.06	5.50	12.75
1	2.77	3.23	15.00	19.61
2	1.33	3.11	39.50	38.24
3	1.0	2.0	67.00	57.84
4	0.54	0.84	84.33	61.76
5	1.56	0.54	91.00	68.63
6	0.63	1.23	95.33	74.51
7	0.02	0.83	95.33	76.47
8	0.12	1.26	96.67	68.63
9	0.82	0.09	97.50	76.47

4. RESULTS AND DISCUSSION

During training, the system continuously adjusts the internal parameters of the neural network to improve its ability to classify signs. The Adam optimizer plays a vital role in this process. It's an optimization algorithm that efficiently updates these parameters based on the network's errors during training. Adam helps the network learn and improve its sign recognition accuracy by iteratively refining these parameters. During training, the loss function measures how well the network's predictions (recognized signs) match the actual signs. The cross-entropy loss function in the system likely calculates the difference between the predicted probability distribution (scores assigned to each sign) and the ideal distribution. Minimizing this cross-entropy loss is the goal during training, as it guides the Adam optimizer to adjust the network's parameters in a direction that leads to better sign classification. This uses Python as a base language for programming. It uses the main libraries like NumPy, pandas, and pytorch. Torch cuda can also be used to execute the program. When training the model, accuracy and loss in model validation data could vary with different cases. In typical cases, the loss decreases, and the accuracy increases as the epoch increases. Three other cases depict the results generated; i) if the validation loss increases and validation accuracy depletes, the model will not learn, ii) the model could be overfitting if the validation loss and accuracy increase, and iii) if the validation loss starts decreasing and validation accuracy increases, the data is being trained correctly. In this system, a section is included that makes predictions on the validation set and provides the result, as shown in Figure 5. The validation data is the unseen data for the training set, and it predicts the sign and prints the class label based on the model's prediction. By using this, good insight is provided during the training and testing.

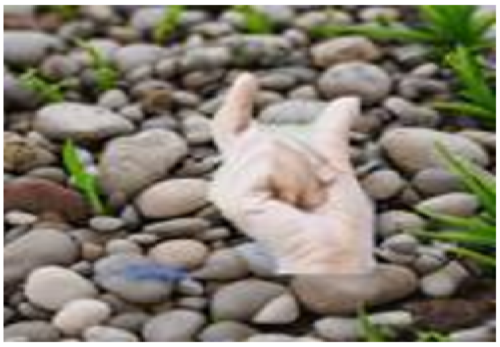

Figure 5. Testing image with actual label as 'U' and predicted label as 'U'

We evaluated the model on the validation set that produced the next results: Figure 6: training and validation accuracy, Figure 7: training and validation loss, Figure 8 and Table 1 illustrating training and validation performance and increasing accuracy and loss with the course of time. Furthermore, Figure 9 is real-time images captured to make predictions. Here again, the model was correct in recognizing signs for "V" and "O," even though there were two separate gestures for "O" . The bottom of each image shows the prediction made.

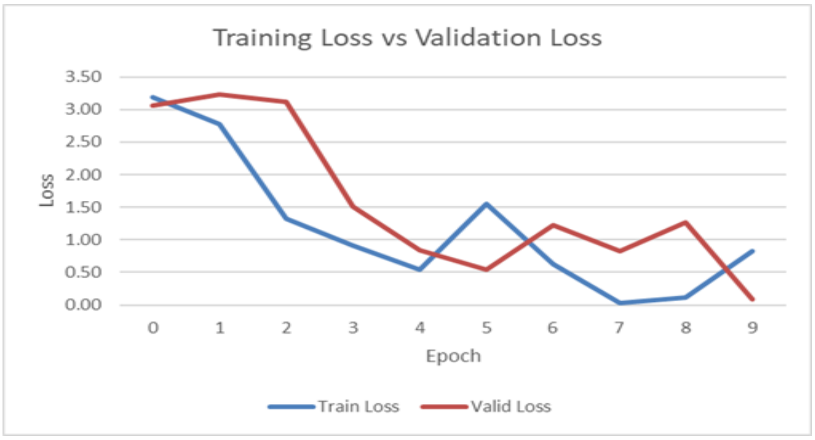

Figure 6. Training loss and validation loss

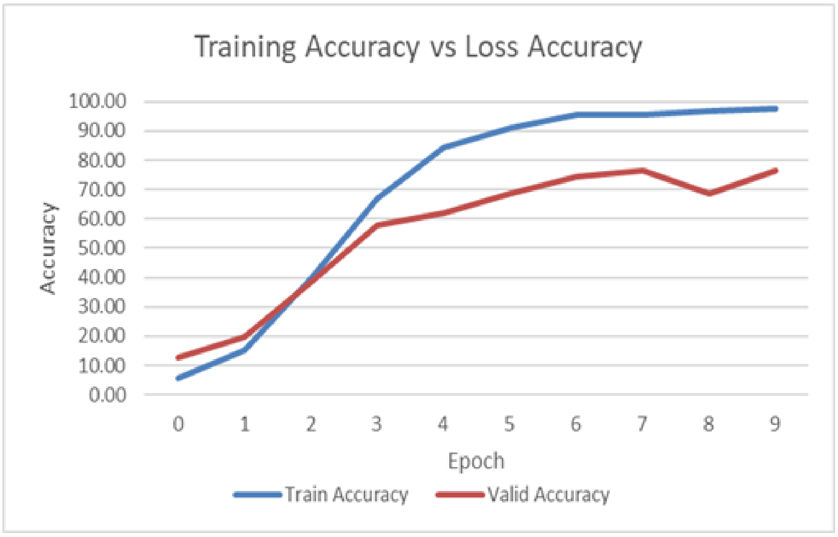

Figure 7. Training accuracy and validation accuracy

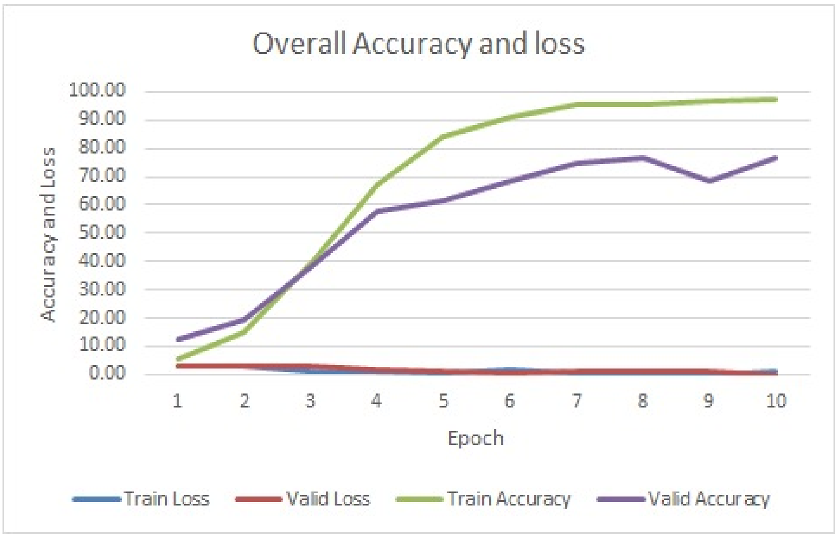

Figure 8. Overall training and loss

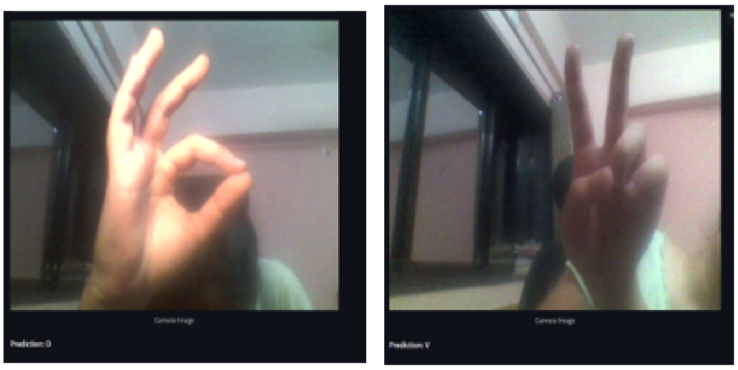

Figure 9. Real-Time predicted outputs of the model (letter O and V)

5. CONCLUSION

In conclusion, this sign language recognition system effectively translates sign images into text or speech, bridging the communication gap between individuals with disabilities and the broader community. In this paper, a training model utilizing the pre-trained VGG16 CNN is developed. The model achieves an accuracy of approximately 97%. Future work can focus on expanding the dataset to enhance the model's accuracy further. Additionally, incorporating advanced image processing techniques can improve the system's performance, leading to more precise and reliable sign language recognition. The proposed method demonstrates the potential of leveraging transfer learning and deep learning for accessible and cost-effective communication solutions.

REFERENCES

[1] N. Musthafa and C. G. Raji, "Real time Indian sign language recognition system," *Materials Today: Proceedings*, vol. 58, pp. 504-508, 2022, doi: 10.1016/j.matpr.2022.03.011.

[2] Y. Obi, K. S. Claudio, V. M. Budiman, S. Achmad, and A. Kurniawan, "Sign language recognition system for communicating to people with disabilities," *Procedia Computer Science*, vol. 216, pp. 13-20, 2023, doi: 10.1016/j.procs.2022.12.106.

[3] H. Alsolai, L. Alsolai, F. N. Al-Wesabi, M. Othman, M. Rizwanullah, and A. A. Abdelmageed, "Automated sign language detection and classification using reptile search algorithm with hybrid deep learning," *Heliyon*, vol. 10, no. 1, p. e23252, 2024, doi: 10.1016/j.heliyon.2023.e23252.

[4] K. Wangchuk, P. Riyamongkol, and R. Waranusast, "Real-time Bhutanese Sign Language digits recognition system using Convolutional Neural Network," *ICT Express*, vol. 7, no. 2, pp. 215-220, 2021, doi: 10.1016/j.icte.2020.08.002.

[5] Y. Liu *et al.*, "A wearable system for sign language recognition enabled by a convolutional neural network," *Nano Energy*, vol. 116, p. 108767, 2023, doi: 10.1016/j.nanoen.2023.108767.

[6] S. N. Koyineni, G. K. Sai, K. Anvesh, and T. Anjali, "Silent Expressions Unveiled: Deep Learning for British and American Sign Language Detection," *Procedia Computer Science*, vol. 233, pp. 269-278, 2024, doi: 10.1016/j.procs.2024.03.216.

[7] 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, doi: 10.1016/j.procs.2023.01.117.

[8] T. Liu, T. Tao, Y. Zhao, M. Li, and J. Zhu, "A signer-independent sign language recognition method for the single-frequency dataset," *Neurocomputing*, vol. 582, p. 127479, 2024, doi: 10.1016/j.neucom.2024.127479.

[9] X. Lu, H. Wang, J. Zhang, Y. Zhang, J. Zhong, and G. Zhuang, "Research on J wave detection based on transfer learning and VGG16," *Biomedical Signal Processing and Control*, vol. 95, p. 106420, 2024, doi: 10.1016/j.bspc.2024.106420.

[10] Y. Chen, Y. Chen, S. Fu, W. Yin, K. Liu, and S. Qian, "VGG16-based intelligent image analysis in the pathological diagnosis of IgA nephropathy," *Journal of Radiation Research and Applied Sciences*, vol. 16, no. 3, 2023, doi: 10.1016/j.jrras.2023.100626.

[11] D. K. Singh, "3D-CNN based Dynamic Gesture Recognition for Indian Sign Language Modeling," *Procedia Computer Science*, vol. 189, pp. 76-83, 2021, doi: 10.1016/j.procs.2021.05.071.

[12] M. A. Rahaman, K. U. Oyshe, P. K. Chowdhury, T. Debnath, A. Rahman, and M. S. I. Khan, "Computer vision-based six layered ConvNeural network to recognize sign language for both numeral and alphabet signs," *Biomimetic Intelligence and Robotics*, vol. 4, no. 1, p. 100141, 2024, doi: 10.1016/j.birob.2023.100141.

[13] Suharjito, R. Anderson, F. Wiryana, M. C. Ariesta, and G. P. Kusuma, "Sign Language Recognition Application Systems for Deaf-Mute People: A Review Based on Input-Process-Output," *Procedia Computer Science*, vol. 116, pp. 441-448, 2017, doi: 10.1016/j.procs.2017.10.028.

[14] Y. C. Bilge, N. Ikizler-Cinbis, and R. G. Cinbis, "Cross-lingual few-shot sign language recognition," *Pattern Recognition*, vol. 151, p. 110374, 2024, doi: 10.1016/j.patcog.2024.110374.

[15] P. Mohan, T. Sabarwal, and T. Preethiya," Indian Sign Language Character Recognition System," *2023 4th International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Coimbatore, India, 2023, pp. 1029-1034, doi: 10.1109/ICESC57686.2023.10193309.

[16] Y. I. Rokade and P. M. Jadav, "Indian Sign Language Recognition System," *International Journal of Engineering and Technology*, vol. 9, no. 3, pp. 189-196, 2017, 10.21817/ijet/2017/v9i3/170903S030.

[17] K. Shenoy, T. Dastane, V. Rao, and D. Vyavaharkar, "Real-time Indian Sign Language (ISL) Recognition," *2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Bengaluru Bengaluru, India, 2018, pp. 1-9, doi: 10.1109/ICCCNT.2018.8493808.

[18] R. Patil, V. Patil, A. Bahuguna, and G. Datkhile, "Indian Sign Language Recognition using Convolutional Neural Network," *ITM web of conferences*, vol. 40, 2021, doi: 10.1051/itmconf/20214003004

[19] H. K. Vashisth, T. Tarafder, R. Aziz, and M. Arora, "Hand Gesture Recognition in Indian Sign Language Using Deep Learning," *Engineering Proceedings*, vol. 59, no. 1, p. 96, 2023, doi: 10.3390/engproc2023059096

[20] A. Dodia and S. Kumar, "A Comparison of YOLO Based Vehicle Detection Algorithms," *2023 International Conference on Artificial Intelligence and Applications (ICAIA) Alliance Technology Conference (ATCON-1)*, Bangalore, India, 2023, pp. 1-6, doi:10.1109/ICAIA57370.2023.10169773.

[21] N. Sarma, A. K. Talukdar, and K. K. Sarma, "Real-Time Indian Sign Language Recognition System using YOLOv3 Model," *2021 Sixth International Conference on Image Information Processing (ICIIP)*, Shimla, India, 2021, pp. 445-449, doi: 10.1109/ICIIP53038.2021.9702611.

[22] B. Saini, D. Venkatesh, N. Chaudhari, T. Shelake, S. Gite, and B. Pradhan, "A comparative analysis of Indian sign language recognition using deep learning models," *Forum for Linguistic Studies*, vol. 5, no. 1, pp. 197-222, 2023, doi: 10.18063/fls.v5i1.1617.

[23] S. Kumar, R. Rani, and U. Chaudhari, "Real time sign language detection: Empowering the disabled community," *MethodsX*, vol. 13, p. 102901, 2024, doi: 10.1016/j.mex.2024.102901.

[24] S. Kumar, R. Rani, and S. K. Pippal, "Design of Appliance Control Prototype with Voice Command Using IoT for Physically Disabled People," *2023 2nd International Conference on Futuristic Technologies (INCOFT), Belagavi, Karnataka, India*, 2023, pp. 1-6, doi: 10.1109/INCOFT60753.2023.10425774.

[25] A. Singh, A. Mishra, S. Chitgopkar, T. Mahajan, and S. Kumar, "Homex: An Intelligent Home Automation and Security System," *2022 OPJU International Technology Conference on Emerging Technologies for Sustainable Development (OTCON), Raigarh, Chhattisgarh, India*, 2023, pp. 1-6, doi: 10.1109/OTCON56053.2023.10114053.

BIOGRAPHIES OF AUTHORS

Sumit Kumar received a bachelor's degree in Electronics and Telecommunication from Kurukshetra University, Kurukshetra, India in 2005, the master's degree from Guru Jambheshwar University of Science and Technology, Haryana, India in 2008, and a Ph.D. degree from Jamia Millia Islamia, Delhi, India in 2017. He works as a Professor at the Electronics and Telecommunication Department of Symbiosis Institute of Technology, Symbiosis International (Deemed University), Pune, Maharashtra, India. His research areas are machine learning and deep learning. He can be contacted at email: er.sumitkumar21@gmail.com.

Ruchi Rani received a bachelor's degree in Computer Science Engineering from Kurukshetra University, Kurukshetra, India in 2008, the master's degree from Maharshi Dayanand University, Haryana, India in 2012. She is currently pursuing a Ph.D. degree from the Department of Computer Science Engineering, Indian Institute of Information Technology, Kottayam, Kerala, and working as an Assistant professor at the Department of Computer Engineering and Technology, School of Computer Engineering and Technology, Dr. Vishwanath Karad MIT World Peace University, Pune 411038, Maharashtra, India. Her research interests include machine learning and deep learning. She can be contacted at email: ruchiasija20@gmail.com.

Sanjeev Kumar Pippal received B.Tech. degree from MJP Rohilkhand University, M.Tech., and Ph.D. from MNNIT Allahabad. His area of interest is cloud computing, distributed computing, and blockchain. He is certified in machine learning and deep learning. He has published over 30 research papers in SCI/Scopus International Journals and Conferences. He has filed four patents and published 02 patents. He can be contacted at email: sanpippalin@gmail.com.

Ulka Chaudhari is pursuing her Bachelor of Technology in Computer Science Engineering at the Department of Computer Engineering and Technology, School of Computer Engineering and Technology, Dr. Vishwanath Karad MIT World Peace University, Pune, Maharashtra, India. Her research interests include machine learning and deep learning. She can be contacted at email: ulka.m.chaudhari@gmail.com.