Journal of Soft Computing Paradigm (ISSN: 2582-2640)
www.irojournals.com/jscp/

Classification of Electromyographic Hand Gesture Signal using Deep Learning

Maitreyi Rajaraman[1], Sarojini Premalatha J.[2]

Department of Computer Science, Sathyabama Institute of Science and Technology, Chennai, Tamilnadu, India

E-mail: [1]rrmaitreyi@gmail.com, [2]sarojinipremalatha.j.cse@sathyabama.ac.in

Abstract

The electromyography (EMG) signal measures the electrical activity of muscles and is often described as a function of amplitude, frequency, and phase over time. These signals are commonly employed in both clinical and biomedical applications. They are used to identify neuromuscular disorders and in other activities such as controlling robots and computers. This proposed study utilizes the CNN to analyse the hand gestures and extract valuable information from these gestures. Consecutive training and testing using images were conducted to evaluate the CNN's performance. The findings demonstrate the effectiveness of the proposed methodology in discerning significant features from complex movements.

Keywords: Convolutional Neural Network, Sign Language Recognition, Hand Gestures, Image Processing, EMG.

1. Introduction

Human communication includes many forms, such as spoken language and non-verbal cues like gestures. However, there is a notable gap in communication between those who can hear and those who have hearing impairments. This communication gap becomes more noticeable for people with hearing and speaking impairments, as the traditional methods are not very effective. This paves the way for an innovative solution to bridge the gap and help people with impairments. To address this issue, the research explores the hand gestures or the sign language used by hearing and speaking-impaired people. Beyond written or spoken words, language also consists of gestures. It can be very difficult for people who use sign language to

Extended version of article received from 4th INTERNATIONAL CONFERENCE on Artificial Intelligence,5G Communication and Network Technologies
DOI: https://doi.org/10.36548/jscp.2024.3.002
Received: 02.06.2024, received in revised form: 26.06.2024, accepted: 10.07.2024, published: 23.07.2024

communicate their ideas to people who are not familiar with them. This complexity is further increased by the vast cultural variations in sign language.

This study focuses on employing technology to recognize static hand motions, which are an important component of sign language. Convolutional neural networks (CNNs) are to be used to interpret these hand movements. CNNs are effective instruments for identifying patterns in complex images and comprehending them. This research attempts to create a more inclusive atmosphere where everyone can express themselves by bridging the communication gap between the deaf-mute community and others using CNNs.

2. Related Work

Many techniques have been used in the identification of sign language in an attempt to comprehend the meanings of gestures with the hands and symbols. Early methodologies, such as SIFT, LBP, and HOG-based static gesture recognition, laid the framework for understanding geometric hand features [3,8], These rules were relaxed in certain research to take into consideration differences, but more additional data was required to increase CNN accuracy. [5].

Additionally, the utilization of Binary Image methods integrating "Hu moments or Hu invariants" enhanced robustness in feature extraction and KNN-based classification. A more elaborate exploration involved analysing hyperparameters' influence on hand gestures through convolutional neural networks and EMG data [6]. Moreover, the advent of technologies like Microsoft Kinect-V2 introduced novel models for database creation. In the domain of CNN-based sign language recognition, approaches like ORB (Oriented FAST and Rotated BRIEF) integration and Fisher Vector’s representation [11] presented advanced feature extraction techniques, empowering SVMs for accurate classification. A groundbreaking achievement showcased in [12] yielded an exceptional 98.5% accuracy in detecting and recognizing American sign language, emphasizing the potential of SVMs and CNNs in enhancing sign language comprehension.

These collective efforts underscore a spectrum of methodologies, ranging from geometric feature extraction to sophisticated CNN-based recognition, significantly contributing to the advancement of sign language recognition technology. Such multifaceted approaches

hold promise in enhancing inclusivity and societal integration for the deaf-mute community[9,10].

The researchers have employed diverse techniques and algorithms to decipher signs across both video and image formats. These efforts have extended to encompass a wide array of gestures used by individuals who are deaf or mute. For instance, the research in [1] focuses on developing a real-time hand gesture recognition system using CNN by capturing the hand gesture images using the web camera.

The authors of study [2] explore the use of CNN in Indian sign language recognition whereas in [4] the study explores the use of multiple kernel learning for Pakistan sign language recognition. The author Shubham A sangale et al [7] applies the RNN, CNN, and SVM for the real-time hand gestures collected through the webcam.

The comparison shows that CNN performs better with an accuracy of 91% over the other methods. Based on the insight gained from the related work the CNN was used in the proposed study to explore the real-time images that were collected using the web camera. Additionally, the work also developed a user interface to test its performance.

3. Hand Gesture Recognition

The utilization of CNN for hand gesture computation proves more effective than traditional component-based recognition methods, as it avoids potential confusion caused by other body parts such as the elbow. This approach accommodates gestures involving hand movement, including gestures incorporating palms, wrists, and elbows within the frame. The subsequent sections elaborate on the various steps involved in developing this network.

A. Image Capture

Images used as input were captured using the computer's webcam and processed using the MATLAB code. The entire image captured from the webcam served as input, with a blue box displayed to guide users in performing the desired gesture, as depicted in Figure 1.

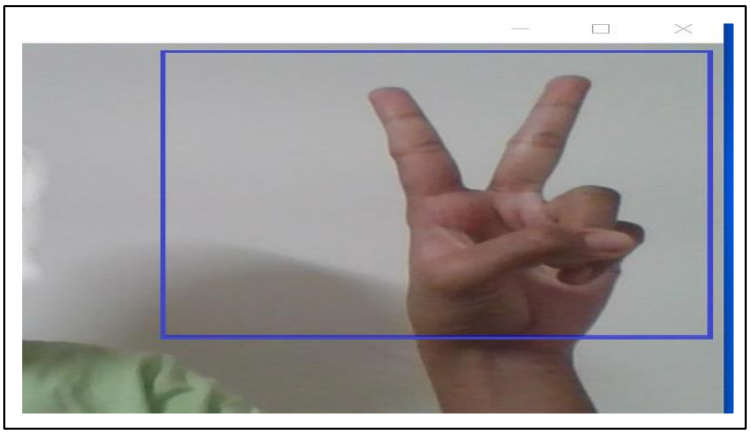

Figure 1. Input Image (Gesture within Blue Box)

B. Image Processing

Each webcam-captured image underwent meticulous handling to optimize its suitability for network training. Initially, background subtraction was conducted by deducting the captured gesture image from the program's first-captured background image. Following this, the image underwent grayscale transformation, reducing its dimensions from 80*50*3 to 80*50*1, consolidating the RGB layer into a single layer. Subsequently, the image was converted to a black and white format using a threshold value of 10.

C. Training Set

Each image used to train the network was unique and contributed to a dataset specific to different gestures (1, 2, 3, 4, 5). A total of 400 images were generated, with each gesture having a set of 80 distinct images. These images were systematically produced using a 'for loop' in the programming sequence. To assist the network in better understanding the gestures, each image was labeled according to its distance perception, categorized as either "closer" or "farther." This labeling strategy was important in refining the network's ability to interpret and categorize gestures accurately when users provided input. Figure. 2 showcases a selection of examples from the resulting image set, illustrating the varied gestures used for training and their categorized distances.

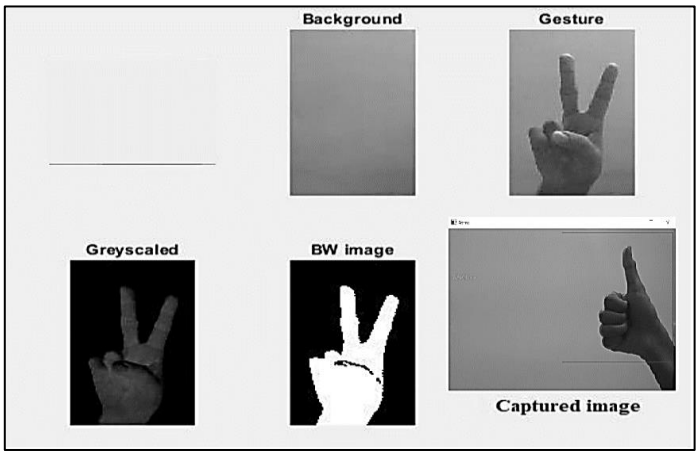

Figure 2. Image Pre-Processing

During the training phase of the Convolutional Neural Network (CNN), several hyperparameters were tuned to optimize the model's performance in recognizing and classifying hand gestures. These hyperparameters include:

Learning Rate: This parameter controls the step size during the optimization process, influencing how quickly or slowly the model learns from the training data. A suitable learning rate ensures that the model converges efficiently without overshooting or getting stuck in local minima.

Batch Size: The batch size determines the number of training samples processed in each iteration before updating the model's parameters. It balances the trade-off between computational efficiency and the quality of parameter updates, affecting the stability and convergence speed of the training process.

Number of Epochs: An epoch refers to one complete pass through the entire training dataset. The number of epochs defines how many times the model iterates over the entire dataset during training. Tuning this hyperparameter helps prevent underfitting or overfitting by finding the optimal balance between model complexity and generalization.

Optimizer: The choice of optimizer algorithm, such as Adam, RMSprop, or SGD, influences how the model's parameters are updated during training. Each optimizer has its own set of hyperparameters, such as momentum and decay rates, which can be adjusted to improve convergence and performance. The proposed study utilizes Adam optimizer.

Regularization: Regularization techniques, like dropout and L2 regularization, are used to prevent overfitting by penalizing overly complex models. Tuning the regularization strength helps control the model's capacity and improves its ability to generalize to unseen data. Figure. 3 shows the hand gesture identified using CNN.

The Table.1 below shows the details of the hyperparameter used

Table 1. Hyperparameters Used

Hyperparameter	Values
Learning Rate	1×10^{-3}
Batch Size	32
Number of Epochs	20
Optimizer	Adam
Dropout Rate	0.3
L2 Regularization	1×10^{-4}

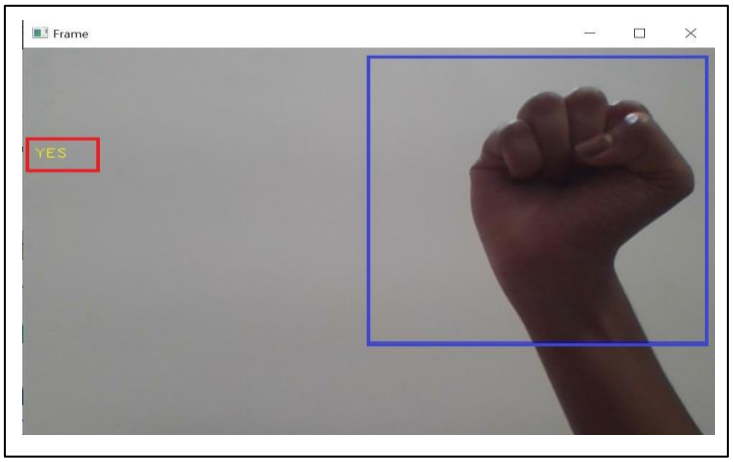

Figure 3. Hand Gesture Identified using CNN

4. Methodology

The Convolutional Neural Network (CNN): The choice of CNN architecture for this study relies on several essential factors, that are required for analysing the gestures effectively. The study focused on employing a CNN architecture with ability to capture the detailed and

intricate features of the images. Architectures like ResNet and VGG have demonstrated exceptional performance in image recognition tasks due to their deep structures and sophisticated feature extraction capabilities. Additionally, architectures with adaptable configurations, such as Inception and MobileNet, were considered for their efficiency in handling varying image resolutions, which is particularly relevant in capturing the subtleties of gestures across different contexts. Moreover, architectures incorporating recurrent or attention mechanisms, such as LSTM-CNN or Transformer-based models, were explored to account for temporal dependencies in sequential gestures, enhancing the model's ability to discern meaningful patterns over time. The proposed study uses the basic CNN model for image processing and feature extraction tasks. Through thorough experimentation and performance evaluation, the selected architecture showcased superior proficiency in accurately interpreting gestures, thus validating its suitability for the intended task.

4.1 Architecture

The image in Figure 4 visually outlines how Convolutional Neural Networks (CNNs) process information in three dimensions—width, height, and depth. In CNNs, neurons are structured to receive 3D inputs and produce 3D outputs. For RGB images, where each pixel has three color channels (red, green, blue), the depth is typically set to 3, while the height and width correspond to the image's spatial dimensions. This depiction illustrates CNN's proficiency in handling image data. The layers within a CNN—such as convolutional layers, pooling layers, and fully connected networks—combine to effectively extract features from images. Convolutional layers detect various patterns like edges and textures, pooling layers reduce spatial dimensions while preserving important information, and fully connected layers analyze these features for accurate classification or recognition [6].

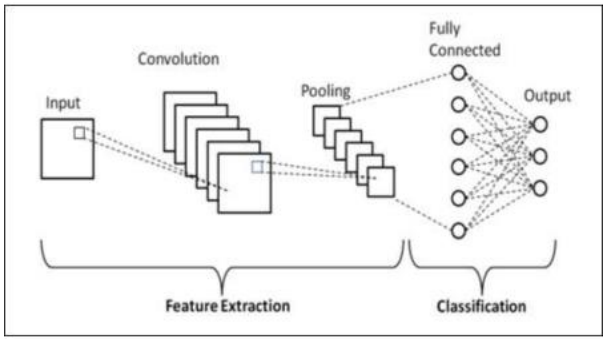

Figure 4. CNN Architecture

The Table.2 below shows the Model Summary of the CNN used

Table 2. Model Summary

Layer Type	Output Shape	Param#
Conv2D (Conv1)	(None, 128, 128, 32)	896
MaxPooling2D (MaxPool1)	(None, 64, 64, 32)	0
Conv2D (Conv2)	(None, 64, 64, 64)	18,496
MaxPooling2D (MaxPool2)	(None, 32, 32, 64)	0
Conv2D (Conv3)	(None, 32, 32, 128)	73,856
MaxPooling2D (MaxPool3)	(None, 16, 16, 128)	0
Flatten	(None, 32,768)	0
Dense (Dense1)	(None, 256)	8,388,864
Dropout	(None, 256)	0
Dense (Dense2)	(None, 128)	32,896
Dropout (Dropout1)	(None, 128)	0
Dense (Dense3)	(None, 96)	12,384
Dropout (Dropout2)	(None, 96)	0
Dense (Dense4)	(None, 64)	6208
Dense (Dense5)	(None, 5)	325
Total Parameters		
Total Parameters:	8,533,639	
Trainable Parameters:	8,533,639	
Non-Trainable Parameters:	0	

4.2 Tools Used

The user interface was designed and developed using Python 3.6+ with PyCharm as the IDE. The libraries utilized include Numpy, IO, OS, Keras, and Flask. The hardware

configuration comprises an Intel i3 processor, 160GB hard disk, and 8GB RAM, while the software configuration involves the Windows 10 operating system.

4.2.1 Creating Node.js Web App

An essential aspect of deploying a deep learning model is to share it with others through a user-friendly interface. Node.js, an open-source, cross-platform runtime, is particularly advantageous for real-time, data-intensive applications, making it an ideal fit for the research. The Node.js application, written in JavaScript, leverages asynchronous and event-driven programming for non-blocking execution and a responsive user experience. Its speed in code execution, scalability through a single-threaded model with an event loop, and absence of data buffering seamlessly integrate with our machine learning model. This user-friendly interface, developed using HTML, CSS, and JavaScript, enhances the deployment and accessibility of our model, allowing users to interact with and experience the trained model in action. HTML structured the web pages, CSS was used to style them, and JavaScript added interactivity. In the backend, Node.js handled server operations, Flask managed the machine learning model, and Keras built and trained the model

5. Result

The experiment focused on the network's ability to interpret and recognize hand gestures initiated by the user. The outcome of this analysis is visually represented in Figure. 7, showcasing the detected hand gesture captured at the top of the camera frame. Throughout the testing phase, an average recognition time of approximately 721 milliseconds was recorded. It's important to note that this recorded time is subject to the specific hardware configuration of the system executing the code. Factors such as processing power, memory, and other system specifications can impact the recognition speed.

This recognition time, is an essential metric, that must be considered in context. Future enhancements and optimizations might further refine the system's performance, potentially reducing the time required for accurate gesture recognition. The Figures 4-8 depict the user interface developed and the results obtained.

5.1 Home Page

The homepage displays a logo alongside a menu bar offering options for "Home," "About," "Upload," and "Live Stream." It's complemented by a cover page emphasizing "Sign Language Recognition" at the forefront.

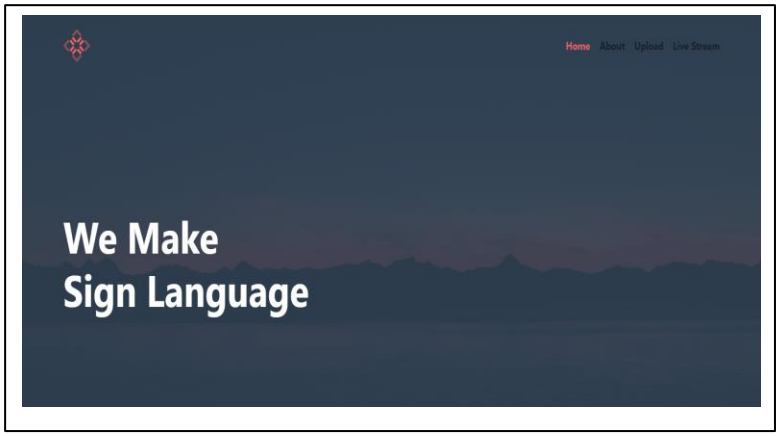

Figure 4. Home Page

5.2 About Page

The "About" page typically provides insights into the system. It details the purpose, goals, and background of the initiative. This section includes information about the developers, their motivations, the technology employed, and the objectives behind the sign language recognition system. It's a space to share the journey, milestones, and aspirations of the research.

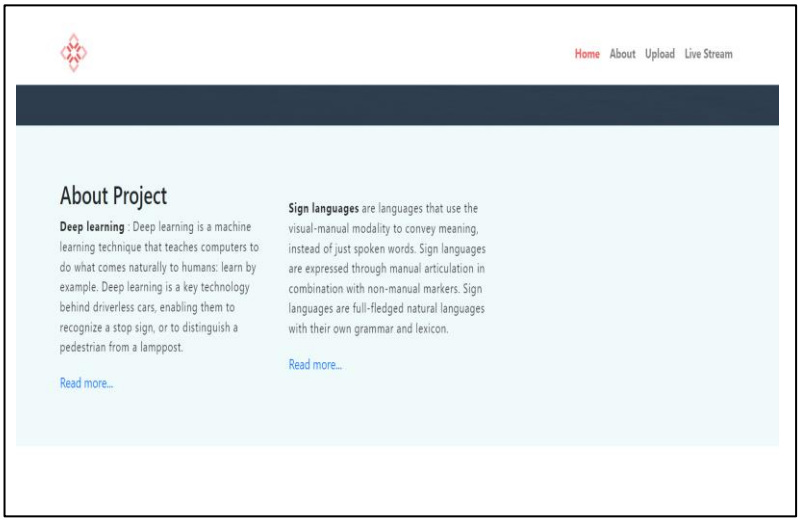

Figure 5. About Page

5.3 Image Uploading Page

The "Image Upload" page allows users to submit images for classification. Users can upload images containing hand gestures or sign language signs. These images are then processed by the system's deep learning model, which identifies and categorizes the gestures. The page includes features like a file upload interface, a submit button, and feedback on the classification results.

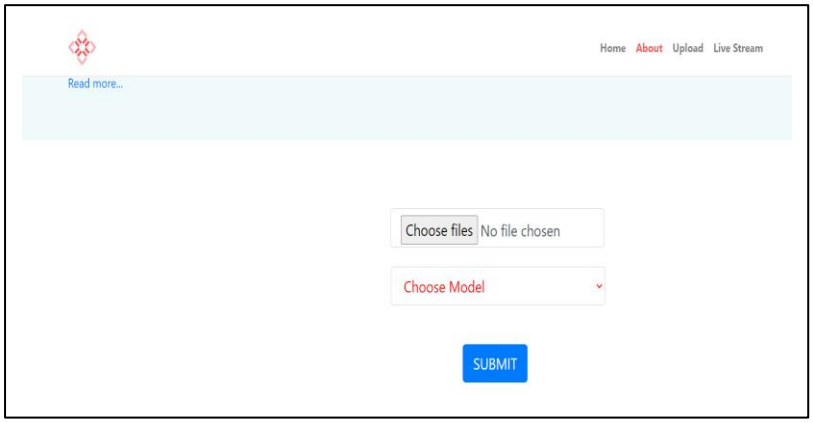

Figure 6. Image Uploading Page

5.4 Classified Output

Below is the output classified as sign M.

Figure 7. Output of Sign M

Below is the output classified as sign R.

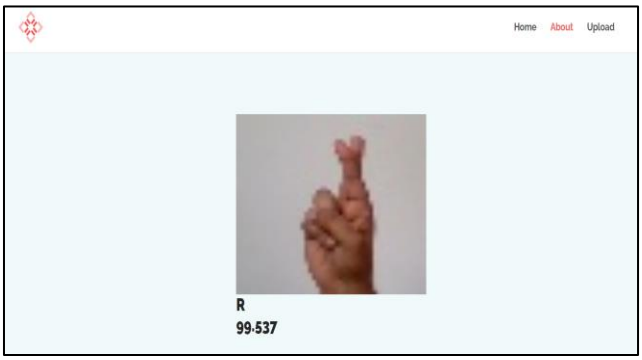

Figure 8. Output of Sign R

6. Conclusion and Future Work

In summary, this research journey into hand gesture recognition using Convolutional Neural Networks (CNN) has shown the effectiveness of data augmentation in enhancing deep learning accuracy. The CNN approach demonstrated commendable success in recognizing hand gestures, yet opportunities for enhancement exists. The incorporation of knowledge-driven methodologies like the Belief Rule Base (BRB) could notably enhance accuracy, especially in scenarios involving uncertainty.

Future work might encompass expanding the research of recognized gestures, including those complex backgrounds, and the ability to discern gestures made with both hands. While the current system adeptly identifies signs under various conditions, there's an ongoing scope to fine-tune recognition for specific gestures and common words. This system's potential to recognize 40 words, including alphabets, marks a significant stride toward creating a universally accessible platform for the deaf and mute community. Refinement efforts could centre around making the system more user-friendly and easily accessible for individuals with hearing or speech impairments.

References

[1] Narayana, Ch Lakshmi, and M. Venkata Praveena. "Real-Time Hand Gesture Recognition System using CNN." International Journal of Current Science (IJCSPUB) 13, no. 3 (2023): 724-730.

[2] “Alaria, Satish Kumar, Ashish Raj, Vivek Sharma, and Vijay Kumar. "Simulation and analysis of hand gesture recognition for indian sign language using CNN." International

Journal on Recent and Innovation Trends in Computing and Communication 10, no. 4 (2022): 10-14.

[3] “American Sign Language,” National Institute of Deafness and Other Communication Disorders, 14-Dec-2020. [Online]. https://www.nidcd.nih.gov/health/american-sign-language. [Accessed: 12-Feb-2021].

[4] Shah, Farman, Muhammad Saqlain Shah, Waseem Akram, Awais Manzoor, Rasha Orban Mahmoud, and Diaa Salama Abdelminaam. "Sign language recognition using multiple kernel learning: A case study of Pakistan sign language." Ieee Access 9 (2021): 67548-67558.

[5] A. Deshpande, A. Shriwas, V. Deshmukh and S. Kale, "Sign Language Recognition System using CNN," 2023 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE), Bengaluru, India, 2023, pp. 906-911,

[6] Shubham A. Sangale, R.M. Samant, Narayan B. Kirtane, Avinash A. Bhatane, Avinash A. Bhatane Hand Gesture Recognition using Machine Learning with Convolutional Neural Network (CNN) International Journal of Computer Applications (0975 – 8887) Volume 184– No.16, June 2022. 29-32

[7] Sharma, Saransh, and Samyak Jain. "A static hand gesture and face recognition system for blind people." In 2019 6th International Conference on Signal Processing and Integrated Networks (SPIN), Noida, India, IEEE, 2019. pp. 534-539.

[8] Brownlee, Jason. Deep learning for computer vision: image classification, object detection, and face recognition in python. Machine Learning Mastery, 2019.

[9] Prakash, Kolla Bhanu, Rama Krishna Eluri, Nalluri Brahma Naidu, Sri Hari Nallamala, Pragyaban Mishra, and P. Dharani. "Accurate hand gesture recognition using CNN and RNN approaches." International Journal of Advanced Trends in Computer Science and Engineering Volume 9, No.3, June2020. 3216 -3222.

[10] Babitha, Donepudi, T. Jayasankar, V. P. Sriram, S. Sudhakar, and K. B. Prakash. "Speech emotion recognition using state-of-art learning algorithms." International

Journal of Advanced Trends in Computer Science and Engineering 9, no. 2 (2020): 1340-1345.

[11] Bharadwaj, Yellapragada SS, P. Rajaram, V. P. Sriram, S. Sudhakar, and Kolla Bhanu Prakash. "Effective handwritten digit recognition using deep convolution neural network." International Journal of Advanced Trends in Computer Science and Engineering 9, no. 2 (2020): 1335-1339.

[12] M. R. Ahsan, M. I. Ibrahimy and O. O. Khalifa, "Electromygraphy (EMG) signal based hand gesture recognition using artificial neural network (ANN)," 2011 4th International Conference on Mechatronics (ICOM), Kuala Lumpur, Malaysia, 2011, pp. 1-6.