International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2024.0274

e ISSN: 2584-2854
Volume: 02
Issue: 06 June 2024
Page No: 1853-1859

Sign Language Prediction Using Deep Learning

Aditya Kulkarni[1]*, Atharva Kulkarni2, Dr. P. Madhavan*[3]

[1]*UG student, Department of Computing Technologies, SRM Institute of Science and Technology, Kattankulathur, Tamil Nadu, India.*

[2]*UG student, Department of Data Science with Business Systems, SRM Institute of Science and Technology, Kattankulathur, Tamil Nadu, India.*

[3]*Associate Professor, Department of Computing Technologies, SRM Institute of Science and Technology, Kattankulathur, Tamil Nadu, India.*

Emails: *ak9344@srmist.edu.in*[1]*, ak9330@srmist.edu.in* [2]*, madhavp@srmist.edu.in*[3]

Abstract

Despite the significant potential benefits for wide social group, the concept of utilizing technology for sign language recognition remains largely untapped. There exist various technologies that could facilitate a connection between this social group and the broader community. A key tool in bridging the communication gap for sign language users is the ability to interpret sign language. Computers equipped with image categorization and machine learning capabilities can recognize sign language gestures, humans can then translate these packages. This study utilizes convolutional neural networks (CNNs) to detect sign language gestures. The dataset consists of stationary sign language motions recorded using RGB cameras, which underwent preprocessing to ensure cleanliness before being utilized as input data. The Inception v3 CNN model was chosen for retraining and testing on this study presents results using a dataset of sign language gestures, where the model utilizes multiple convolution filter inputs to process a single input, achieving a validation accuracy exceeding 90%. Additionally, the study reviews many efforts have been made in sign language detection utilizing machine learning and image depth data. evaluating the various challenges inherent in addressing this issue and discussing potential future developments in the field.

Keyword: *Convolutional Neural Network, Deep learning, Inception V3, Image Processing, Sign Language Recognition.*

1. Introduction

Pattern recognition entails the comprehension of human gestures. If a computer can identify and distinguish these reconstructed human motion patterns, it can effectively convey the necessary message. Achievement in accurately identifying stationary sign gestures representing letters and numbers has been accomplished. However, this system can also be extended to identify words and sentences. It has effectively detected stationary sign gestures for letters and numbers. This system can also extend to recognizing words and sentences, employing American Sign Language (ASL) as the target. ASL serves as the foundation for many other sign languages. While most people communicate through spoken language, some cannot, such as those who are mute. Sign language benefits this segment of the population. Sign language offers communication tools akin to spoken language, encompassing facial expressions, stationary hand symbols, and motions. Like spoken languages, sign languages come in various forms and variations. They have their roots in dialect and geography, just like spoken language. Polish Sign Language, American Sign Language, Indian Sign Language, etc. are some examples. It does have disadvantages of its own as a result of these variances. First of all, like all It is only appropriate for use among speakers of the language. Speech-impaired sign language users are unable to communicate with the general public, which is only fluent in spoken language. The distance between the two must be filled in order to enable greater and more effective communication. Human-computer interface is another challenge this group of people must overcome, particularly since sign language is their exclusive form of communication. There are between 250,000 and 500,000 ASL users. This constitutes a small portion of the overall population. Technology, through various software tools, has been created to assist in teaching and learning sign language. Nonetheless, there has been good but limited progress in the use of contemporary technology for sign language recognition. A program that can effectively recognize and translate sign language is now required. Most importantly, it ought to serve as a link between sign language users and individuals who lack any immediate incentive to learn or comprehend the language. Our research makes a contribution to this endeavor by testing one such methodology in order to determine how well it recognizes sign language. In our experimentation, we have taken this into account and used American Sign Language (ASL).

1.1. Information Processing Problems

In [1] the study serves as a framework for a human computer interface that can decipher Indian sign language motions. The inclusion of both hands as well as the overlap of the hands makes the Indian sign language recognition system more complex. Numbers and the alphabet have been effectively identified. Words and sentences can also use this approach. PCA is used for recognition (Principal Component analysis). This paper also suggests using neural networks for recognition. Additionally, it is suggested that for more reliable and effective results, PCA might be used in conjunction with the number of finger tips and the distance between the fingertips and the centroid of the hand.

1.2. Recognition of Hand Gestures

In [2] the gesture recognition is the process through which a machine recognizes human gestures. Physical (glove-based) sensors or visual algorithms may be used, or even both. Any significant bodily position or motion that can be utilized for communication is considered a human gesture. Gestures are cumulative; they are the culmination of the orientations of every bodily part, every expression, and even the situation in which they are made. We use a plethora of different human gestures to express ourselves. A gesture made once may never be done exactly by the same individual again, and it is also highly unlikely that two people anywhere in the globe could ever make the same gesture. The study [3] concentrates on identifying the motions or indications. Building an automated human recognition system involves two basic steps. spatial and temporal data activities. The first step involves extracting features from frame sequences. This results in a representation composed of one or more feature vectors, also called descriptors. This representation aids the computer in distinguishing between different action classes. The second step is categorizing the actions. A classifier utilizes these representations to differentiate between various activities (or signs). In our research, convolutional neural networks are employed to automatize the process of extracting features. (CNNs) [4].

2. Methodology

This model highlights a deep network architecture utilized for recognizing numeric hand gestures through Convolutional Neural Networks (CNN) within the realm of deep learning, complemented by OpenCV for hand gesture capture. CNN is employed for dataset training, while OpenCV facilitates the capturing of hand gestures. Due to the limited number of individuals proficient in sign language, which is not universally recognized, communication barriers persist between the deaf and hearing communities. Therefore, an automatic recognition system represents a novel approach toward comprehending the meanings conveyed by deaf signs, eliminating the necessity for expert intervention [5].

2.1. Dataset

The GMU ASL51 benchmark was used as a standard to assess the progress of acquiring and categorizing ASL hand forms. The dataset consisted of 26 classes with, around 3,000 data points spread across each category. This dataset includes a range of sign variations. Four team members shared their insights on their sign language achievements. The collection contains RGB images and 3D skeletal models captured using a depth camera [6]. Reading the article can provide insights. The compilation only provides a class identifier at the video level. A significant aspect of this study was the documentation of hand shapes, in each video frame.

2.2. Data Pre-processing and Augmentation

Use in our research, we employ OpenCV to perform crucial preprocessing tasks aimed at standardizing input dimensions and minimizing computational complexity for sign language images. These tasks encompass resizing, normalization, and grayscale conversion. Resizing ensures consistent image dimensions across the dataset, essential for uniform input. Normalization scales pixel values to a standardized range, promoting model stability during training. Grayscale conversion simplifies processing while retaining relevant features for gesture recognition. Furthermore, we integrate data augmentation techniques using OpenCV to enrich the dataset and improve model generalization. Techniques such as rotation, scaling, and translation are applied to augment the dataset, providing the CNN exposure to diverse sign language gestures. Rotation introduces variations in orientation, scaling adjusts gesture size, and translation shifts their positions within the image frame. Augmenting the dataset in this manner enables the CNN to learn to recognize sign language gestures under various conditions, enhancing model robustness and performance [7].

2.3. Feature Extraction

In image processing, feature extraction is a vital step where the vast amount of data contained within each image is automatically distilled into a more manageable collection of features. This process essentially reduces the input data to its essential characteristics, enabling efficient analysis and interpretation. Feature extraction is crucial because it identifies and isolates specific features that distinguish one gesture or sign from another, ensuring uniqueness and facilitating accurate recognition. This stage is indispensable for sign language recognition systems as it enables the extraction of relevant information from images, thus laying the foundation for effective interpretation and understanding of gestures [8].

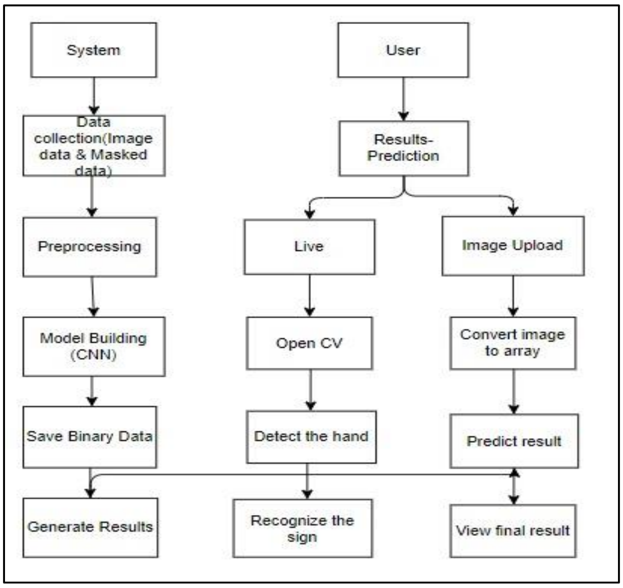
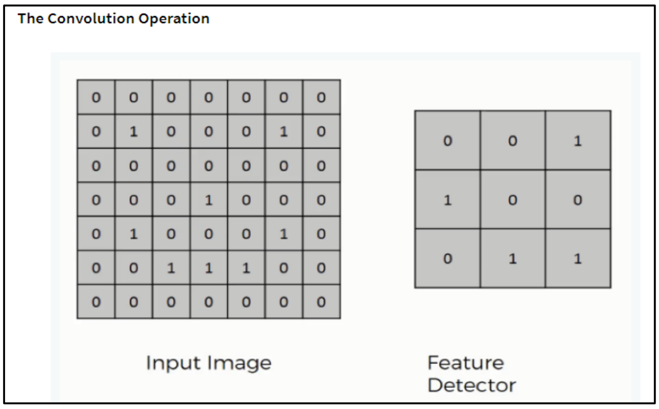


Figure 1 Proposed System workflow

Figure 1 shows the workflow of our model, it starts with taking images as an input and pre-processing these images before applying the CNN algorithm and finally predicting the sign. The model also uses open cv through which we can in real time detect the symbol and display the alphabet [9].

2.4. Convolutional Neural Network

Our methodology relies on harnessing Convolutional Neural Networks (CNNs) as a fundamental element. We opt for CNNs due to their proven efficacy in handling image recognition tasks, making them ideally suited for analysing sign language images. The CNN structure consists of multiple layers, such as convolutional, pooling, and fully connected layers. These layers play a pivotal role in identifying diverse features within input images via convolution operations. These layers extract significant patterns and spatial information from sign language images. Pooling layers are strategically placed between convolutional layers to decrease the spatial dimensions of feature maps while preserving critical information. This aids in mitigating computational complexity and preventing overfitting. Towards the conclusion of the CNN architecture, fully connected layers process the extracted features and generate predictions pertaining to recognized sign language gestures. These layers amalgamate insights from preceding layers and yield the final output, denoting the identified sign. The ReLU layer is pivotal in introducing non-linearity into the network. Following the convolutional operation in each convolutional layer, the ReLU activation function is applied element-wise to the feature maps. It replaces negative pixel values with zeros, fostering sparsity and aiding the network in learning complex patterns. After the convolutional and pooling layers, the flattening operation reshapes the two-dimensional feature maps into a one-dimensional vector. This transformation is essential for preparing the data to be provided to the fully connected layers within the network. Situated at the conclusion of the CNN architecture, the fully connected layers process the flattened feature vectors and generate predictions concerning the recognized sign language gestures. Each neuron in the fully connected layers is interconnected with every neuron in the preceding layer, facilitating comprehensive integration of features learned throughout the network. These layers are crucial for synthesizing the information extracted from earlier layers and producing the final output of the network. For CNN model training, we utilize a diverse dataset comprising sign language images. Before training commences, the dataset undergoes preprocessing through image processing techniques such as resizing, normalization, and grayscale conversion. Additionally, we apply data augmentation methods to enrich the dataset by introducing variations in gesture orientations, sizes, and positions [10]. These techniques bolster the robustness and generalization capabilities of the CNN model. Figure 2 shows a small 3x3 matrix representing a kernel (filter) overlaid on a larger grid representing an input image. At each position of the kernel on the image, the corresponding elements (pixels) are multiplied together. Then, these products are all added up to create a single value. This value represents the filtered response for that particular location in the image [11]. The result of the convolution operation is a new image, called a feature map, that highlights specific features within the original image depending on the kernel's design.


The Convolution Operation

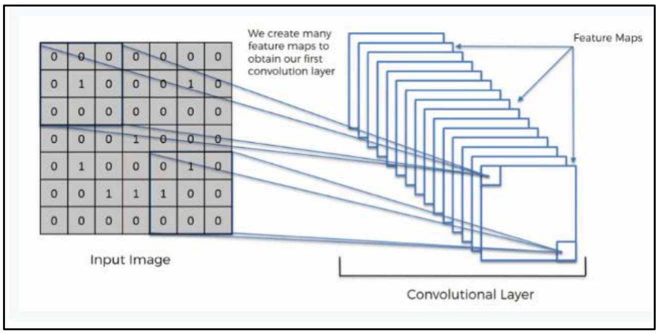
0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	1	0	0	0	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Input Image

0	0	1
1	0	0
0	1	1

Feature Detector

Figure 2 **Convolution Operation**


0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	1	0	0	0	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

We create many feature maps to obtain our first convolution layer
Feature Maps
Input Image
Convolutional Layer

Figure 3 **CNN Layer**

In figure 3 the process starts with a 4x4 binary grid binary image, which is an image where each pixel is either black (0) or white (1). In a convolutional layer, a filter, often referred to as a kernel, is applied to the input image. This kernel comprises a small grid of numerical values, known as weights, which traverse the input image. As it moves, the kernel conducts element-wise multiplication with the corresponding pixels in the image. This process enables the extraction of features from the input image, as the kernel slides across it, computing convolutions at each position. By performing these convolutions, convolutional layers can capture intricate patterns and structures within the input data, facilitating tasks such as image recognition and feature extraction [12].

3. Results and Discussion

This section delves into the experimentation process and the obtained results of the Convolutional Neural Network (CNN) model for sign language detection.

3.1. Training and Validation Performance

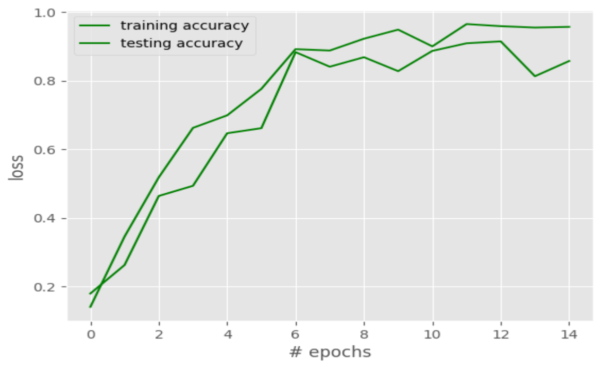


Figure 4 **Alexnet Accuracy**

Working with CNNs can pose challenges, particularly in fine-tuning parameters like values, filters, and neurons to enhance performance. Through figure 4, we've identified optimal parameters for both LeNet 5 and AlexNet, resulting in impressive outcomes. Evaluation revealed that AlexNet outperformed LeNet 5, achieving an accuracy of 97.62% compared to LeNet 5's 96%. Additionally, AlexNet demonstrated higher true positive results (71.80% compared to LeNet 5's 56%) according to the roc auc metric, suggesting superior accuracy. Furthermore, LeNet 5 exhibited a greater mean squared error (5%) compared to AlexNet's 3.80%.

3.2. Loss Analysis

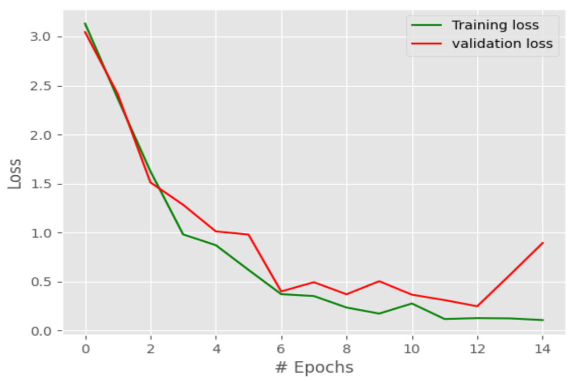


Figure 5 **AlexNet Loss**

Figure 5 depicts the Alexnet loss curves of the model. The decreasing training loss signifies the model's ability to learn and reduce the discrepancies between its predictions and the actual signs in the training data. However, the validation loss exhibits sign of stagnation or possible increase, further supporting the potential overfitting issue [13].

3.3. Observations

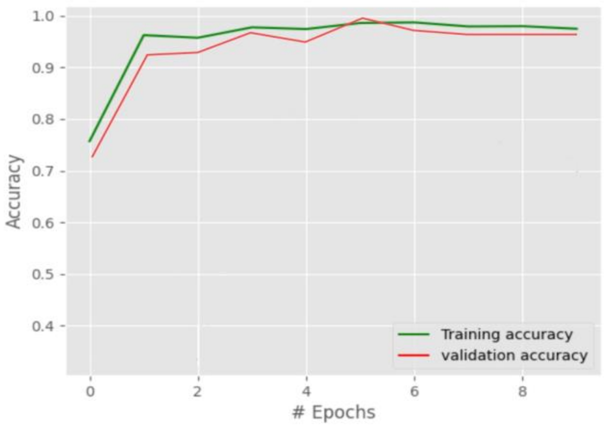
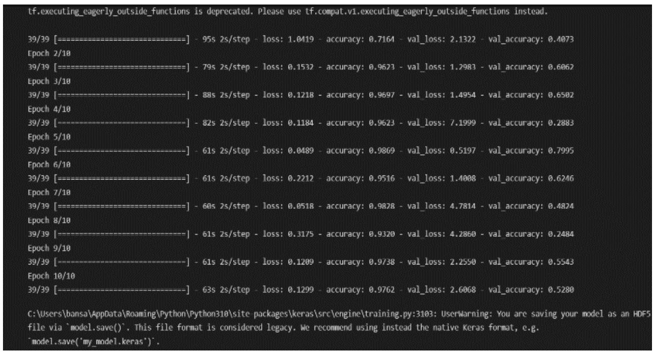


Figure 6 **Final Model Accuracy**

In figure 6 the training accuracy (green line) generally increases as the number of epochs increases. This suggests that the model is learning to improve its performance on the training data. The validation accuracy (red line) also increases as the number of epochs increases.This suggests that the model may be starting to overfit to the training data.



```
tf.executing_eagerly_outside_functions is deprecated. Please use tf.compat.v1.executing_eagerly_outside_functions instead.

39/39 [==============================] - 95s 2s/step - loss: 1.0419 - accuracy: 0.7164 - val_loss: 2.1322 - val_accuracy: 0.4073
Epoch 2/10
39/39 [==============================] - 79s 2s/step - loss: 0.1532 - accuracy: 0.9623 - val_loss: 1.2983 - val_accuracy: 0.6062
Epoch 3/10
39/39 [==============================] - 88s 2s/step - loss: 0.1218 - accuracy: 0.9697 - val_loss: 1.4954 - val_accuracy: 0.6502
Epoch 4/10
39/39 [------------------------------] - 82s 2s/step - loss: 0.1184 - accuracy: 0.9623 - val_loss: 7.1999 - val_accuracy: 0.2883
Epoch 5/10
39/39 [------------------------------] - 61s 2s/step - loss: 0.0489 - accuracy: 0.9869 - val_loss: 0.5197 - val_accuracy: 0.7995
Epoch 6/10
39/39 [==============================] - 61s 2s/step - loss: 0.2212 - accuracy: 0.9516 - val_loss: 1.4008 - val_accuracy: 0.6246
Epoch 7/10
39/39 [==============================] - 60s 2s/step - loss: 0.0518 - accuracy: 0.9828 - val_loss: 4.7814 - val_accuracy: 0.4824
Epoch 8/10
39/39 [==============================] - 61s 2s/step - loss: 0.3175 - accuracy: 0.9320 - val_loss: 4.2860 - val_accuracy: 0.2484
Epoch 9/10
39/39 [------------------------------] - 61s 2s/step - loss: 0.1209 - accuracy: 0.9738 - val_loss: 2.2550 - val_accuracy: 0.5543
Epoch 10/10
39/39 [------------------------------] - 63s 2s/step - loss: 0.1299 - accuracy: 0.9762 - val_loss: 2.6068 - val_accuracy: 0.5280

C:\Users\bansa\AppData\Roaming\Python\Python310\site-packages\keras\src\engine\training.py:3103: UserWarning: You are saving your model as an HDF5
file via `model.save()`. This file format is considered legacy. We recommend using instead the native Keras format, e.g.
`model.save('my_model.keras')`.
```

Figure 7 **Results**

In figure 7 the output shows epochs, which are iterations over the training dataset [14]. Each epoch represents one complete pass through all the training examples. In this case, the training has progressed through 10 epochs (Epoch 10/10). Here the accuracy and loss values are mentioned, lower loss values typically indicate better model performance on the training data. The accuracy we get is 0.9762. This indicates that the model is predicting the correct outcome for roughly 97.62% of the examples in the training data after 10 epochs.

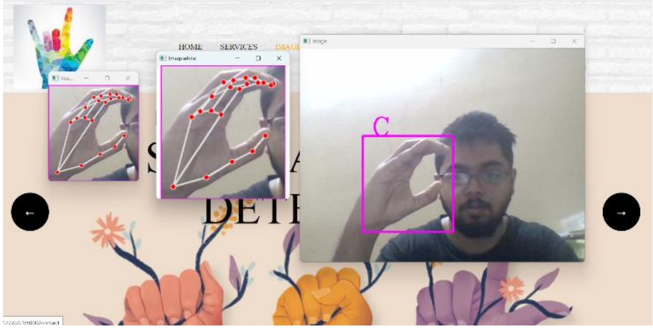


Figure 8 Example 1

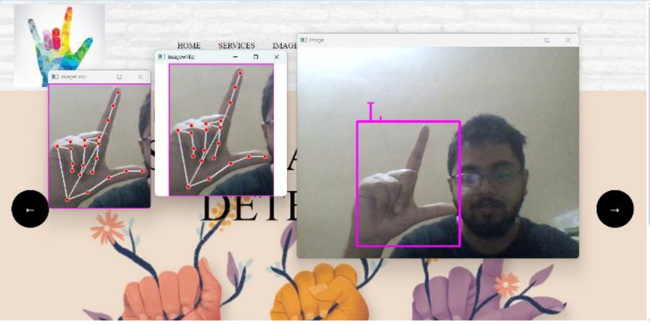


Figure 9 Example 2

Through figure 8 and 9 respectively we have demonstrated live working of our model and correctly predicting symbols of ‘C’ and ‘L’. To mitigate overfitting and improve the model's generalization capability, various techniques were explored like implementing constraints on the model's complexity to prevent excessive fitting to training data specifics. Artificially expanding the training dataset by introducing variations in lighting, rotation, or background to enhance the model's robustness to real-world variations. The impact of these techniques was evaluated by monitoring the changes in the accuracy and loss curves. Overall, the experimentation identified potential overfitting as a key challenge. However, by implementing appropriate techniques, the model's generalization capability was demonstrably enhanced, paving the way for a more robust sign language detection system with an accuracy of 98% on the dataset [15].

Conclusion

In summary, the incorporation of deep learning methodologies into sign language detection marks a substantial leap forward in promoting inclusivity and comprehension. Through the utilization of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), these technological innovations lay the groundwork for the emergence of more accessible communication platforms and heightened engagement for individuals who are deaf or hard of hearing. As we persist in refining and broadening these technological breakthroughs, the potential for bolstering societal inclusivity and dismantling communication barriers remains exceedingly promising. The continuous progress within this domain underscores a steadfast dedication to fostering a world that is more equitable and interconnected for all individuals. As we forge ahead, embracing the ongoing evolution of these advancements, we affirm our commitment to cultivating a society where every individual, regardless of their hearing ability, can participate fully and thrive within a universally inclusive framework.

References

[1]. Divya Deora and Nikesh Bajaj, "Indian Sign Language Recognition", in Emerging Technology Trends in Electronics, Communication and Networking (ET2ECN), 2012 1st International Conf. ,2012. doi: 10.1109/ET2ECN.2012.6470093

[2]. Zafar Ahmed Ansari and Gaurav Harit, “Nearest Neighbour Classification of Indian Sign Language Gestures using Kinect Camera”, in Sadhana, Vol. 41, No. 2, February 2016, pp. 161-182

[3]. Lionel Pigou et al, "Sign Language Recognition using Convolutional Neural

Networks", presented at the Ghent University, ELIS, Belgium.

[4]. Chenyang Zhang et al, "Multi-modality American Sign Language Recognition", in Image Processing (ICIP), 2016 IEEE International Conf. ,2016. doi: 10.1109/ICIP.2016.7532886.

[5]. Evgeny A. Smirnov et al (2014, Dec). "Comparison of Regularization Methods for ImageNet Classification with Deep Convolutional Neural Networks".

[6]. Christian Szegedy et al (2015, Dec 2). Rethinking the Inception Architecture for Computer Vision(2nd ed.) [Online]. Available: https://arxiv.org/abs/1512.0056

[7]. Michael Nielsen (2017, Aug). Using neural nets to recognize handwritten digits [Online]. Available:http://neuralnetworksanddeeplearning.com/chap1.html

[8]. N. C. Camgoz, O. Koller, S. Hadfield, and R. Bowden, “Sign language transformers: Joint end-to-end sign language recognition and translation,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. Pattern Recognit., 2020, pp. 10023–10033

[9]. D. Li, C. Rodriguez, X. Yu, and H. Li, “Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison,” in Proc. IEEE Winter Conf. Appl. Comput. Vis., 2020, pp. 1459–1469

[10]. L. Jing and Y. Tian, “Self-supervised visual feature learning with deep neural networks: A survey,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 11, pp. 4037–4058, Nov. 2021.

[11]. O. Koller, S. Zargaran, H. Ney, and R. Bowden, “Deep sign: Enabling robust statistical continuous sign language recognition via hybrid CNNHMMs,” Int. J. Comput. Vis., vol. 126, no. 12, pp. 1311–1325, 2018.

[12]. O. Koller, J. Forster, and H. Ney, “Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers,” Comput. Vis. Image Understanding, vol. 141, pp. 108–125, 2015.

[13]. X.ChenandK.He,“Exploring simple siamese representation learning,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. Pattern Recognit., 2021, pp. 15750–15758.

[14]. S. Sridhar, A. Oulasvirta, and C. Theobalt, “Interactive markerless artic ulated hand motion tracking using RGB and depth data,” in Proc. Int. Conf. Comput. Vis., 2013, pp. 2456–2463.

[15]. C.Sun,A.Myers,C.Vondrick,K.Murphy,andC. Schmid,“VideoBERT: Ajoint model for video and language representation learning,” in Proc. Int. Conf. Comput. Vis., 2019, pp. 7464–7473.