Tri-Lingual Sign Language Translator

Shivangi Sharma
sharshiv
sharshiv@iu.edu

Abstract

Sign language is mostly utilized by hearing-impaired and mute people to communicate within and outside of their social groups. The goal of sign-language recognition(SLR)[1], is to identify acquired hand motions and to continue until related hand gestures are translated. The fact that some or all of the ordinary people do not speak this language and that every country has its sign language creates a barrier to communication. I have created a model that can understand the gestures and translate them to cater to a wider audience. By developing Deep Neural Network[2] architectures, the model will learn to detect letters and once the model successfully recognizes[3] the gesture, a relevant letter is generated. This paper will be a two-fold endeavor with comparative study as the research part and Tri-Lingual sign language translator, as the application part.

1 Introduction

Sign language is a medium of communication for the population that has difficulty communicating due to hearing, or speaking challenges. Hand gestures are used in sign language to communicate ideas and thoughts visually. There are between 138 and 300 different sign languages being used all over the world. More than 5% of the world's population consists of people with partial or fully disabling hearing or speaking loss. However, due to the profession of a sign language interpreter being the 55th most widely and onerously licensed one, it creates a scarcity of experts which further makes it difficult to teach and learn sign language. Communication in sign language is complex and covers hand gestures, body language, and facial expressions. In this paper, I will be concentrating on hand gestures based on ASL[4], GSL[5], and ISL[6].

Figure 1: Sign Language Hand Gestures

1.1 Motivation

The isolation that the impaired population experiences are what spurs this project's ambition. The impaired population has higher rates of loneliness and depression[7], particularly significant hurdles that negatively impact life quality are caused by the communication gap between the impaired population and the rest of the population. This paper can also aid systems that translate[8] hand gestures into text that is commonly utilized in public places including hospitals, airports, etc.

1.2 Problem Statement

Sign language has particular motions for each letter of the English alphabet and based on them, two categories of sign languages exist Static Gesture[9] and Dynamic Gesture[10]. The dynamic gesture is utilized for specific concepts whereas, the static gesture is used to symbolize the alphabet and numbers. Additionally, dynamic gesture encompasses phrases, clauses, etc. The difference between static and dynamic gestures lies

in the movements of the hand, the head, or both. Despite extensive research efforts over the past few decades, designing a sign language translator remains a difficult task and when done for multiple languages, the difficulty level increases extensively. Additionally, even identical signs can appear very differently depending on the things like angle from which they are viewed, the signer, etc. The primary goal of this research is to employ deep learning, machine learning, and transfer learning to identify the best-performing model based on the defined metrics and utilize that to create a static sign language translator. This research will become a stepping stone for the future study of a dynamic tri-lingual sign language translator[11].

1.3 Objectives

the approach focuses on promoting automatic sign language translation to overcome communication and learning barriers. This research aims to recognize hand gestures that include 26 English alphabet letters(A-Z).

2 Implementation

2.1 Dataset

I have integrated American Sign Language, Indian Sign Language, and German Sign Language hand gestures. The training dataset comprises 1,15,000 photos with a resolution of 200 x 200 pixels. There are 29 classes, with 3 classes each for SPACE, DELETE, and NOTHING. 26 of the classes correspond to the letters A through Z. For application purposes, SPACE, NOTHING, and DELETE were not used and only 26 classes for letters were defined.

2.2 Data Preprocessing

The procedures I used for image preprocessing[12] are as follows:
1. Read images
2. Make all of the photos the same size/form
3. Eliminate noise
4. By dividing the picture array by 255, all image pixel arrays are transformed to values between 0 and 255

A sample of the same can be seen in Figure 2.

2.3 Artificial Neural Networks

Deep Learning's functional component, neural networks[13], is renowned for simulating the behavior of the human brain to resolve challenging data-driven issues. To create the required output, the input data is processed through several layers of artificial neurons that are stacked one on top of the other.

Figure 2: Sample Image before and after Preprocessing

The main components of the artificial neural network are:

Input Layer: It brings the initial data into the system for further processing by subsequent layers of artificial neurons[14].

Hidden Layer: In neural networks, the function that adds weights to the inputs and guides them through an activation function as the output is positioned in a hidden layer between the algorithm's input and output.

Output Layer: The output layer uses data from earlier hidden layers and the model's learnings to make a final prediction.

2.4 Convolutional Neural Network (CNN)

A Convolutional Neural Network (CNN / ConvNet)[15] is a Deep Learning algorithm that can take in an input image, allocate the various elements in the image some learnable weights and biases, and be able to distinguish between them.

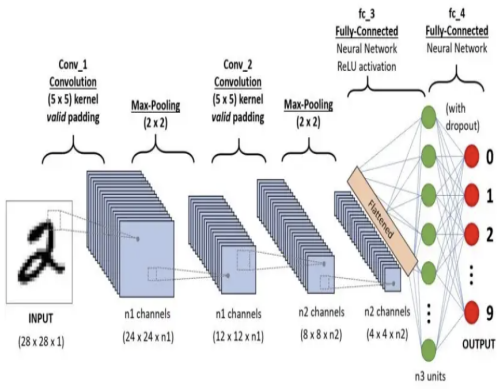

Figure 3: Convolutional Neural Network(CNN)

There are four primary sorts of layers:
1. Convolutional layer
2. Pooling layer
3. Filter
4. Fully connected (FC) layer

2.4.1 Convolutional Layer

It is the center structure block of CNN[13] which becomes more complicated in each layer, distinguishing more prominent segments of the picture. Additionally, there is a feature detector, also referred to as a kernel or filter, which will move through the image's receptive fields and determine whether the feature is present. This is the convolution procedure. Following the application of the filter to a portion of the image, the dot product between the input pixels and the filter is determined. After that, an output array receives this dot product. The filter then advances by a stride and repeats the operation until the kernel has covered the entire image. A convolved feature or an activation map are the terms used to describe the output from the series of dot products produced by the filter and the input. A Rectified Linear Unit (ReLU), which adds nonlinearity[16] to the model, is applied by a CNN after each convolution process.

2.4.2 Pooling Layer

Pooling Layers referred to as downsampling, carry out dimensionality reduction and lower the number of parameters in the input. The pooling layer loses a lot of information, but it also offers CNN several advantages. They lessen complexity, increase effectiveness, and lower the risk of overfitting. There are two types of pooling
1. Max Pooling
2. Average Pooling

Max Pooling: As the name implies, max pooling retains the most noticeable elements in the feature map. This technique is useful for extracting the prominent or very significant aspects from an image.

Average Pooling: As the name implies, the average pooling retains the average values of the pixel.

2.4.3 Filter

The flattening stage as shown in Figure 4, converts the pooled feature map produced during the pooling process into a one-dimensional vector.

2.4.4 Fully Connected Layer

Each node in the output layer of the fully-connected layer is connected directly to a node

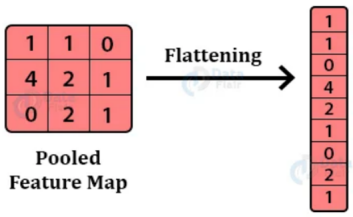

Figure 4: Flattening Process

in the layer above it. Based on the features retrieved from the previous layers and their various filters, this layer conducts the classification operation. Fully Connected layers often utilize a softmax activation function to classify inputs suitably while Convolutional and pooling layers frequently use ReLu[17] functions.

2.5 Architectures

2.5.1 VGG16 (Visual Geometry Group)

VGG16 is an object identification and classification algorithm that has a 92.7% accuracy rate when classifying 1000 photos into 1000 different categories. It is a well-liked technique for classifying images and is simple to employ with transfer learning.

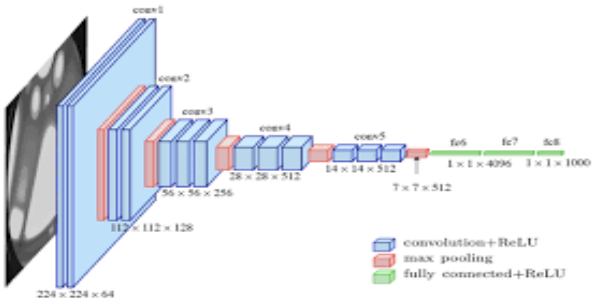

Figure 5: Implementation of VGG16

VGG16 Architecture:
1. The 16 in VGG16[18] refers to 16 layers that have weights. In VGG16 there are thirteen convolutional layers, five Max Pooling layers, and three Dense layers which sum up to 21 layers, but it has only sixteen weight layers.
2. VGG16 takes input tensor size as 224, 244 with 3 RGB channel
3. Conv-1 Layer has 64 filters, Conv-2 has 128 filters, Conv-3 has 256 filters, filters, Conv 4 and Conv 5 have 512 filters.
4. Three Fully Connected (FC) layers follow a stack of convolutional layers: the first two have 4096 channels each, and the third performs

1000-way ILSVRC classification and thus contains 1000 channels (one for each class). The final layer is the soft-max layer.

2.5.2 AlexNet

AlexNet was the first CNN that used GPU to boost performance.
AlexNet Architecture:
AlexNet architecture consists of 5 convolutional layers, 3 max-pooling layers, 2 normalization layers, 2 fully connected layers, and 1 SoftMax layer.

2.5.3 DenseNet

By altering the typical CNN architecture and streamlining the connectivity structure across layers, DenseNets[19] alleviate the "Vanishing Gradient Problem". A densely Connected Convolutional Network is the name given to an architecture called DenseNet because each layer is directly connected to every other layer.

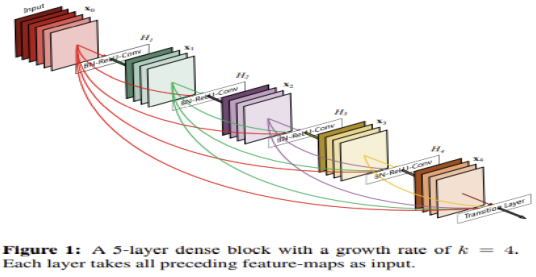

Figure 1: A 5-layer dense block with a growth rate of $k = 4$. Each layer takes all preceding feature-maps as input.

Figure 6: Block Diagram of DenseNet

DenseNet Architecture:
The four primary components of the DenseNet architecture are:
1. Connectivity
2. DenseBlocks
3. Growth Rate
4. Bottleneck layers

3 Experimental Results

Based on the comparative study, below is the graph that depicts the results of various models that I have trained the data on.
Also, below are some examples of the predicted images after training the model on the dataset.

Also, as the application part, I have created a Multilingual sign language translator, and below are some of the sample images that are translated to other sign language gestures, given an image of one sign language gesture of the dataset.

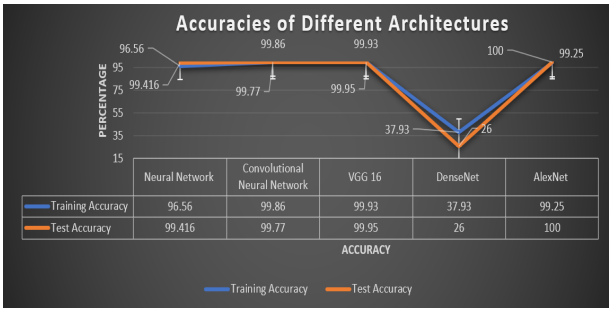

Figure 7: Table and Graph Representation of Obtained Results

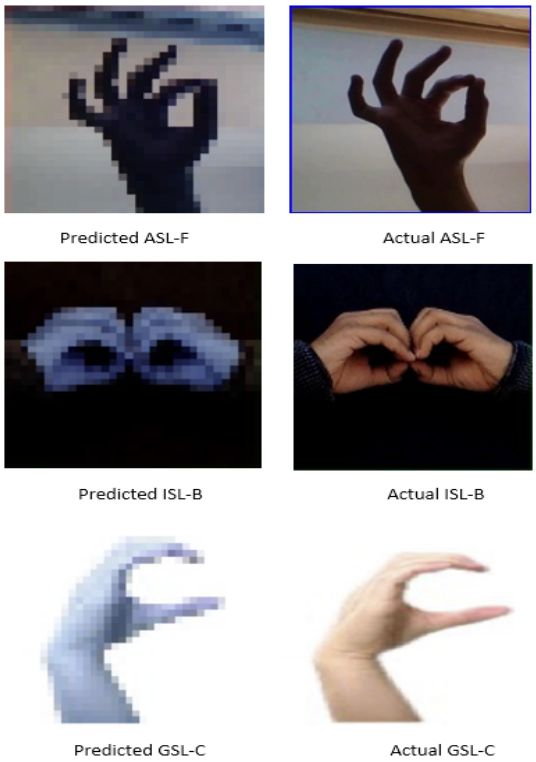

Figure 8: Actual and Predicted Images

4 Conclusion

In conclusion, I was successful in creating a system that can comprehend sign language and convert it to matching text. There are still a lot of gaps in this system, such as the fact that it can only recognize hand motions for the letters A to Z, leaving out body gestures and other dynamic gestures.
It can be further enhanced and optimized in the future.

References

[1] Lean Karlo S Tolentino, RO Serfa Juan, August C Thio-ac, Maria Abigail B Pamahoy, Joni Rose R Forteza, and Xavier Jet O Garcia. Static sign language recognition using deep learning. *Int. J. Mach. Learn. Comput*, 9(6):821–827, 2019.

[2] Muhammad AL-Qurishi, Thariq Khalid, and Riad Souissi. Deep learning for sign language recognition: Current techniques, benchmarks, and open issues. *IEEE Access*, PP:1–1, 09 2021.

[3] Aeshita Mathur, Deepanshu Singh, and Rita Chhikara. Recognition of american sign language using deep learning. In *2021 International Conference on Industrial Electronics Research and Applications (ICIERA)*, pages 1–5, 2021.

[4] Kshitij Bantupalli and Ying Xie. American sign language recognition using deep learning and computer vision. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 4896–4899, 2018.

[5] Shruti Mohanty, Supriya Prasad, Tanvi Sinha, and B. Niranjana Krupa. German sign language translation using 3d hand pose estimation and deep learning. In *2020 IEEE REGION 10 CONFERENCE (TENCON)*, pages 773–778, 2020.

[6] Pratik Likhar, Neel Kamal Bhagat, and Rathna G N. Deep learning methods for indian sign language recognition. In *2020 IEEE 10th International Conference on Consumer Electronics (ICCE-Berlin)*, pages 1–6, 2020.

[7] Mohammad Rostami, Bahman Bahmani, Vahid Bakhtyari, and Guita Movallali. Depression and deaf adolescents: a review. *Iranian Rehabilitation Journal*, 12(1):43–53, 2014.

[8] Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, Hermann Ney, and Richard Bowden. Neural sign language translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[9] Ankita Wadhawan and Parteek Kumar. Deep learning-based sign language recognition system for static signs. *Neural computing and applications*, 32(12):7957–7968, 2020.

[10] Ilias Papastratis, Christos Chatzikonstantinou, Dimitrios Konstantinidis, Kosmas Dimitropoulos, and Petros Daras. Artificial intelligence technologies for sign language. *Sensors*, 21(17):5843, 2021.

[11] R Martin McGuire, Jose Hernandez-Rebollar, Thad Starner, Valerie Henderson, Helene Brashear, and Danielle S Ross. Towards a one-way american sign language translator. In *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.*, pages 620–625. IEEE, 2004.

[12] Salvador García, Sergio Ramírez-Gallego, Julián Luengo, José Manuel Benítez, and Francisco Herrera. Big data preprocessing: methods and prospects. *Big Data Analytics*, 1(1):1–22, 2016.

[13] G. Anantha Rao, K. Syamala, P. V. V. Kishore, and A. S. C. S. Sastry. Deep convolutional neural networks for sign language recognition. In *2018 Conference on Signal Processing And Communication Engineering Systems (SPACES)*, pages 194–197, 2018.

[14] Andrej Krenker, Janez Bešter, and Andrej Kos. Introduction to the artificial neural networks. *Artificial Neural Networks: Methodological Advances and Biomedical Applications. InTech*, pages 1–18, 2011.

[15] Lionel Pigou, Sander Dieleman, Pieter-Jan Kindermans, and Benjamin Schrauwen. Sign language recognition using convolutional neural networks. In Lourdes Agapito, Michael M. Bronstein, and Carsten Rother, editors, *Computer Vision - ECCV 2014 Workshops*, pages 572–578, Cham, 2015. Springer International Publishing.

[16] Ruey S Tsay. Nonlinearity tests for time series. *Biometrika*, 73(2):461–466, 1986.

[17] Yu-Dong Zhang, Chichun Pan, Junding Sun, and Chaosheng Tang. Multiple sclerosis identification by convolutional neural network with dropout and parametric relu. *Journal of computational science*, 28:1–10, 2018.

[18] Tanseem N. Abu-Jamie and Samy S. Abu-Naser. Classification of sign-language using vgg16. *International Journal of Academic Engineering Research (IJAER)*, 6(6):36–46, 2022.

[19] Rangel Daroya, Daryl Peralta, and Prospero Naval. Alphabet sign language image classification using deep learning. In *TENCON 2018 - 2018 IEEE Region 10 Conference*, pages 0646–0650, 2018.