Bangla Sign Language Recognition and Sentence Building Using Deep Learning

by

Safayet Anowar Shurid
16101030
Khandaker Habibul Amin
16101093
Md. Shahnawaz Mirbahar
16101025
Dolan Karmaker
16101058

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering
Brac University
April 2020

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.

2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.

3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.

4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Shurid

Safayet Anowar Shurid
16101030

Nabil

Khandaker Habibul Amin
16101093

Shahnawaz.

Md. Shahnawaz Mirbahar
16101025

Dolan

Dolan Karmaker
16101058

Approval

The thesis/project titled “Sign Language Recognition Using Deep Learning” submitted by

1. Safayet Anowar Shurid (16101030)
2. Khandaker Habibul Amin (16101093)
3. Md. Shahnawaz Mirbahar (16101025)
4. Dolan Karmaker (16101058)

Of Spring, 2020 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on April 7, 2020.

Examining Committee:

Supervisor:

(Member)

Dr. Md. Ashraful Alam
Assistant Professor
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Rabiul

Dr. Md. Golam Rabiul Alam
Associate Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Prof. Mahbub Majumdar
Chairperson
Dept. of Computer Science & Engineering
Brac Universitv

Dr. Mahbubul Alam Majumdar
Professor and Chairperson(CSE), Interim Dean, School of Sciences
Department of Computer Science and Engineering
Brac University

Abstract

Modern age being the era of Information technology, it would not have come this far without the piled up data or information. Whereas communication is the basis of collecting or gathering data or information, almost 5% of the world's population is not blessed with the ability of verbal communication [1]. For deaf and dumb people lacking the ability of verbal communication, sign language is the solution. Sign language varies from the verbal language in every form and rule. This creates a gap between people conversing in verbal language and those communicating in sign language. Verbal languages are easy to interpret for having a common rule-following but sign language differs from region to region. This hampers the communication between normal people and those interacting in sign languages. Human to human interpretation is tough because of the enriched word wise signs and vocabs. To eradicate this issue, we are proposing a machine-based approach for training and detecting the Bangla Sign Language. Our aim is to train the system with enough samples containing different signs used in Bangla Sign Language. In this research, we are using the Convolutional Neural Network (CNN) for training each individual sign. In addition to working as a medium of communication between the deaf and mute with the remaining society, this approach would also serve as a tool for the hearing deprived to learn and use the sign language properly. Moreover, this would also come to assistance for anyone willing to learn or develop sign language or wishes to work with those with special needs of using sign language.

Keywords: Communication; Sign Language; Bangla Sign Language; deaf and mute; Convolutional Neural Network;

Acknowledgement

At first, all commendation to the Great Allah for whom our thesis have been finished with no significant interference.

Then we want to thank our beloved supervisor Dr. Md. Ashraful Alam sir who inspired us as well as gave us proper guideline which helped us to complete our thesis. His kind advices were really helpful. Also thanks to Dr. Md. Ashraful Alam sir for providing us Intel RealSense Depth Camera to create our dataset.

And finally we also thank to our friends and families who helped to create Bangla Sign Language dataset.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Objective

People with the inability of hearing or speaking cannot converse or interact with others via natural or verbal languages. For the purpose of communication they are solely dependent on sign language. In different region there exists sign language of different sort. Sign language which is way more versatile than the natural verbal languages is rich in contexts. Unlike any other verbal language, sign language cannot be bound or formulated under any specific grammatical rules. Just like the name goes the language operates with signs, each specifying a particular word or emotion. Each sign language is labeled with unique signs of its own based on the region and main language that it is built on. There can always be the same signs meaning different words in two different verbal languages.

In sign language one uses gesture and body language combined with facial expression to convey a certain meaning. While letters or alphabets can be formed with some static signs or hand shapes, a complete sentence formation requires continuous movements of the hand, gesture and facial expression. Sign language can become as complex as wanted. To define a single emotion or feeling one might require to combine body posture, lip movements, facial expression, hand movements, head movement and so on. Which is why proper identification of a complete sign language with all the words and vocabularies is troublesome. It is also not evident for each of the disable person to provide with a human interpreter for communication.

Other than communication, the people with inability of listening and speaking also require to learn sign language for their learning and formal education. Just like verbal languages there exists hundreds or thousands of sign languages but not all of them are recognized. For almost every verbal language, region wise a certain sign language is given recognition or legal acceptance which is further followed for training or teaching the disabled countrywide in schools. Here in Bangladesh, the one sign language that is legally accepted and followed countrywide to teach the speech and hearing impaired is developed by the Centre for Disability in Development (CDD).

Language barriers always trouble people even living in different borders. Due to people having lack of knowledge about the other language it is always a must to have an interpreter to keep the communication smooth and flowing. While the scenario is this harsh with both using verbal language, the concept is quite clear how it affects the speech and hearing-impaired people. One of the major reasons which hamper and limit their regular participation in society and other activities is this. People not having a proper idea or knowledge about the sign language greatly restricts communication with impaired people. Which holds back many awarding opportunities and stops them from attending their field of interests. Many of the great minds of this world were hearing or speech impaired. So it can be assumed people who have the inability to speak and hear could do much more if provided with proper assistance and opportunities. The question that remains here is how to assist someone when we lack the ability to communicate with them. Somehow the bigger picture here shows our inability to connect with them even though we are blessed with the ability. This drove us towards finding a solution that would limit the gap instead of the communication and build a bridge instead of a boundary. This very reason worked as the motivation of our current research on Bangla Sign Language and the thought process of developing a tool to assist in recognizing and interpreting the language properly.

There have been various research works done on the stated problem. Different methodologies and techniques were implemented for sign language recognition. In this paper, we are discussing our machine-based method where we are using CNN to train the system and to take samples of images and create the dataset we are using Intel RealSense. In this paper, we are discussing our work point and methodology for tracking the exact sign through training and building a model which would assist the hearing and speech impaired people with a complete interpretation of the original verbal sentence.

1.2 Report Overview

The objective of our research is to find an automated system where deaf and dumb people can easily express their thought in the natural language. So the motivation for us is to build a system by recognizing and classifying various hand signs and assigning the meaningful sentence for that sign so that the abled people can easily communicate with them. For this we studied how sign language works and we found out that each and every language has their own set of unique way to show signs. So as a result we tried to specify various Bangla sign languages. We used Intel RealSense Depth Camera D435 to capture information's, because it is perfect for taking depth information. After getting the information we used Open Source Computer Vision software library to provide a common infrastructure for applications of both computer vision and use machine perception. We also used Keras to experiment faster with deep learning models. Then we used a special kind of multilayered Neural Network also known as Convolutional Neural Network to differentiate among given inputs. After that we passed the images in the input layer of a CNN. After that we found out how hidden layer terminates all the functionality to execute the algorithm of our model. Finally in the output layer we saw that after being reduced in dimension it produces a single value form the entire input image. So we can say that Our whole system is consisting of three subsystems which are Dataset Processing, CNN model designing, training and testing and Generating real time voice output.

Firstly we created dataset with meaningful Bangla sign language words. As giving all the sign language is complex we took some datasets from Kaggle and GitHub. Then we started manipulating the data. We manipulated the data by turning a colored image into a threshold binary image which has only 1 channel. After this we augmented the data using random brightness, horizontal flip, vertical flip, random channel shift, random zoom and random shearing. Then we labeled our datasets sequentially according to the numbers, vowels and consonants. Finally with our augmented datasets we used back propagation algorithm by applying multiple filters and eventually with this model we found out the results and looked if the result is being predicted accurately by our model.

Chapter 2

Literature Review

2.1 Sign Language

The communication medium between deaf and mute people with other normal people is sign language. Gesture based communication is a language that utilize the visual-manual methodology to pass on significance. Basically, sign language needs hand gestures as well as facial expression to deliver the messages. Deaf and mute people feel the importance of sign language because it is their only communication medium with other people.

People who are using sign language express words. They use two to three words or more words to express the feelings. Those words must be in a sequence to make a sentence. Sometimes deaf people also use some facial expression that helps other people to understand the meaning. Sign language is not easy because there are also rules to follow like any language. Same word can express different meanings if there is a change in sequence between other words. Sign language has a common misconception. The misconception is sign languages are not original language. Sign languages depends on the spoken language. When we talk in our language those are converted into sign language [2]. These are the main misconception. Though these misconceptions are totally wrong. Now sign language stablished as a real language.

There is another misconception exist in people mind who does not know sign language. The misconception is that sign language is universal, which is wrong. Every language has its own sign language. Each one of them is different from the other languages. The alphabets are using in sign language those are having different shapes from other language alphabets.

2.2 Bangla Sign Language

The Ethnologue Languages of the World, lists that there are 142 sign languages are in use, however this number is hard to accurately pin down due to new sign languages frequently being created at schools in village communities with high levels of congenital deafness [3]. So we can say that sign language is not universal. As a result, in different communities and countries there are different types of sign languages based on their culture and interaction with each other. There is a common thought among the people that sign language is based on the spoken language of that culture, which is absolutely incorrect. Sign language is invented by the local deaf and mute people. As a result in Bangladesh, we also see a different pattern of sign language used by the local deaf and mute people. So the sign of something in English sign language is totally different in Bangla sign language.

As well as other sign languages there are also characters and digits in Bangla sign language. There are total 6 vowels and 30 consonants in Bangla sign language which can be interpreted into any signs. But most commonly rather than pointing out alphabets they usually use a group of word which makes a meaningful sentence so that the recipient can understand what he or she is trying to express.

In Bangladesh there are more than 30 lakh hearing impaired people. But unfortunately the opportunities for learning sign language are inadequate and disappointing. In Dhaka, Society of Deaf Sign Language Users (SDSL) is the lone organization attempting to teach sign language using an organized framework [4]. Although the Bangla sign language was recognized as a language of Bangladesh by the declaration of the prime minister in February 1, 2009 unfortunately there has been no proper steps taken to institutionalize the language. As a result many of them are deprived from education, employment opportunities and many other services.

2.3 Related work

Throughout the world, researchers have done some good works in sign language conversion field. Most of them have done converting sign language to text format which contains only words. Though very few have done sign language to text format which is a complete sentence. There are lots of work has been done by the researcher on American Sign Language (ASL). Although there have been many works done in this field of sign language recognition, very limited work is done on Bangla Sign Language. Converting Bangla Sign Language to text or into a proper sentence is very rare. Previously done works are reviewed here for having a glimpse over and differing ours from those.

In paper [5], the authors built a model for sign language recognition using the linguistic sub-units. They proposed two models for the built system while the Hidden Markov Models provide an accuracy of 54%, the sequential pattern boosting improved the result to 76%. The reason here is sequential pattern boosting works better with noise minimization. The system was based on the appearance data along with the inferred ones from 2D and 3D tracking which was later merged by a sign level classifier. The dataset was built by using Kinect sensor for accurate depth tracking which provided with the 3D tracking. This method of using sub-units was in practice since 1993. The first ones to adopt this method were Kim and Waldron (1993). Cooper and her team trained their system with multi signers which makes the system better tested due to the availability of various types of data. The accuracy produced by the system is much more reliable as it was not tested on a single signer.

Another paper that describes work on recognition and translation of sign languages, uses AI based solutions [6]. The paper mainly focuses on the works that were done or are being done in the field of expressive or rendering signs. While reading a sign is more troublesome for machines as it requires previous training and knowledge about all the necessary information, rendering a sign is comparatively easier as the machine requires to project a sign based on the current learning that it has. While to read a complete sign it would require to first isolate the sign, then recognize it and later match it with the learning or training that it has received or gained in the earlier period. In the field of robotics, a robotic hand that can spell words with the help of manual alphabets was built to assist the hearing or vision impaired but it could not properly replicate the letters that required wrist movement. In 1994, Ralph a fourth-generation computer controlled robotic hand was developed which marked excellency in the field of fingerspelling. CyberGlove which was developed on the basis of capturing signs through virtual reality, different sensors are used to measure the finger flex angles. Most researchers are currently experimenting on the development of systems in which they are using “mechanical skeletons” where in the signer’s joint the sensors were directly placed. For a more complex system to collect detailed data on body movement, body vest, gloves and headgear are used. CopyCat which is a gesture-based computer game for the hearing impared kids is built based on computer vision. Other than Hidden Markov Models, Neural Networking is the most promising method in the sign reading and rendering field. Different branches of Neural Networking are used to develop and integrate different systems for the purpose of sign rendering and reading.

There is a research paper which is about American Language detection with only thirty words. They used hand tracking system with Hidden Markov Model (HMM) [7]. They input the sign to their model from RWTH-BOSTON-50 database. After input the sign they achieved the error rate of 10.90

Another few researcher work on Indian Sign Language (ISL). They also used the static signs to develop their model [8]. Their model recognize the Indian sign language using deep learning algorithm. The algorithm they have used is convolutional neural networks (CNN). The dataset contains 35,000 sign images. These are the 100 static sign images from different people from angle. They evaluated their model on approximately 50 CNN models. The highest training accuracy they got 99.72% on colored image. They also got 99.90% on grayscale images. On testing accuracy they evaluate their model on precision, recall and F-score.

According to the paper for vision based hand gesture spotting and recognition using CRF and SVM, they used gesture spotting and recognition technique to find out hand gesture from the continuous movement of the hand. The hand motion is recognized by the Conditional Random Fields in conjunction with Support Vector machine. Firstly, they segmented the hand portion from the motion. For that they used YCbCr color space and 3D depth map. Here 3D depth map is used to remove the complex background form the frame. They also used 3D spatio-temporal features for hand volume of dynamic affine-invariants like elliptic Fourier and Zernike moments are extracted, in addition to three motion features [9]. Lastly they used discriminative Conditional Random Fields Model to uniquely identify the hand gesture that is extracted from the test. And the hand gesture is meaningful is monitored by the Support Vector Machine, which basically monitors the start and end point of a meaningful gesture. According to their test the proposed model can identify gesture from continuous hand motion with an accuracy of 92.50

According to the another paper which is Sign Language Spotting with a Threshold model Based on Conditional Random fields they felt difficulty with the instances where there is variant in both motion and appearance, also with the signs that appears with continuous gesture stream. So the model they proposed using threshold models in a conditional random field. This model works in such a way that it performs and adaptive threshold to distinguish between the signs and vocabulary and non-sign patterns. A short-sign detector, a hand appearance-based sign verification method, and a sub sign reasoning method are included to further improve sign language spotting accuracy [10]. According to their experiments their system can spot signs from a continuous flow of data with the accuracy of 87.0 percent and can recognize data from isolated data with an accuracy of 93.5 percent. They also experimented with the CRFs without using a threshold model where they got 73.5 percent accuracy with the continuous flow of data and 85.4 percent with the isolated data. However, their system can also identify 15.0 percent sign error rate from continuous data and 6.4 percent sign error rate from isolated data where this percentage goes to 76.2 percent for continuous data and 14.5 percent for isolated data which is for Conditional Random fields.

2.4 Intel RealSense Depth Camera D435

Intel RealSense Depth Camera D435 is a product of Intel RealSense D400 series. This series was introduced in January 2018. This camera is perfect for taking stereo depth information. Intel RealSense Depth Camera D435 camera has Intel RealSense Vision Processor D4 which has high depth resolution maximum 1280x720 at 30 frames per second (fps) [11]. D435 camera has special feature like capturing motion object as well as it also able to reduce the blind spots. It has USB 3.0. The specification about D435 is given below in the table.

Property	Values
Use Environment	Indoor/Outdoor
Depth Technology	Active IR stereo
Components Included	RealSense Vision Processor D4
Depth Stream Output Resolution	Up to 90 frame per second
Max Range	10 meters
RGB Sensor Resolution & Frame Rate	1920x1080 at 30 frame per second
Camera Dimension	90mm x 25mm x 25mm
Connectors	USB 3.0 Type-C

Table 2.1: Intel RealSense Specifications

Table data taken from [11].

2.5 OpenCV

OpenCV or Open Source Computer Vision is an open source computer vision and machine learning software library. Built to provide a common infrastructure for application of both computer vision and use of machine perception. The main focus of OpenCV being image processing, video capture and analysis, it plays a crucial role in real-time computer vision applications like the face or object detection [12]. The basis of our research is to detect and recognize the signs made or produced by the user skeleton or signer. As the system needs to perform in a real-time detection scenario, which would model and replicate the human vision via computer vision, OpenCV is the perfect match.

OpenCV operates on the basis of computer vision. Computer vision is just a replication model of human vision combining both hardware and software. The process is to reconstruct, interrupt and understand a 3D scene from its 2D image. The usage of computer vision is seen in almost all fields. Image processing, pattern recognition, and photogrammetry are common fields that overlap with computer vision [13]. Unlike image processing which has input and output data both as images, computer vision takes input in the form of image but provides output as a description or interpretation of the input image. This is the formulation required in our research. OpenCV uses computer vision besides being an open-source software library which is easy to configure and fits perfectly with the requirement of our research and so has been used in this model.

OpenCV was built with keeping the focus on real-time applications for computational efficiency. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android [14]. Python is simpler compared with other programming languages due to its simpler programming syntaxes and codse readability, so python has been used for our work.

2.6 Keras

Keras is an API for high level neural networks. It is written in the language Python and it can be continuously run on top of TensorFlow,CNTK and Theano. It helps to experimenting faster with deep learning models. It helps to show result from the idea as faster it can get. So basically keras is a deep learning library that can help us with more faster and much simple way of prototyping while supporting convolutional networks and recurrent networks, as well as the combinations of both of the networks. It can run without any flaw both on CPU and GPU. It is compatible with Python version 2.7 to 3.6.

Keras is very user friendly. As machine language is tougher to understand prototyping solely in machine language is tough. So keras is very helpful here because this API is designed to be more human friendly. It priorities the user experience by providing consistent and simple APIs which eventually reduces the cognitive load. However by using this we can easily combine the neural layers, cost function, optimizers, initialization schemes, activation functions, regularization schemes and all other standalone modules in our new model. So, we can say that this helps us with modularity also. As more days goes by new modules and extensions can be added to the system. As, Keras is user friendly it makes sure that these extensions can be added easily to the module. Moreover as this API is written in Python the models created can be compact and debugging becomes very easy. This makes Keras very powerful yet free to use open source Python library. It is very helpful to evaluate deep learning models.

It is helpful for defining and training our neural network models easily as it combines highly efficient numerical computation libraries such as Theano and TnesorFlow. So in the end one thing we can say that training deep neural network model is very complex and frameworks build for machine learning are much harder to understand but Keras is the API designed for human understanding and user friendly experience while being one of the leading high level neural networks APIs.

2.7 Convolutional Neural Network

The current vision or goal of almost all modern technologies or research out there is to minimize the gap between machines and humans. From Natural Language Processing to Computer Vision, from robotic arms to voice synthesizing almost all the fields that have been on the leap and spreading exponentially aim to serve one single purpose. Computer vision is one of such fields that thrive to bridge this gap between machines and humans. The prime and prior motive of this field is to assist the machines to view the world as we, humans do. Along with observation it also aims to provide a machine the capability of segmenting, analyzing, interpreting and using the knowledge from that the same way a human would do. With time the construction and betterment of computer vision merged with Deep Learning have converged to the Convolutional Neural Network.

Convolutional Neural Network which is a special kind of multilayered Neural Network. It is a Deep Learning algorithm that can differentiate among given inputs by classifying them based on their features. The feature extraction part works on the basis of given images and the assigned importance. Here assigned importance refers to the fact of providing learnable weights and biases. As CNN is mostly used in supervised learning, the images that the algorithm will try to differentiate are always predefined or labeled. Based on the previously labeled or defined images which work as the training or learning dataset, CNN tries and finds the weight and biases. Later on the algorithm uses the same weights to use on the input images to find a readable value and compare with the earlier learning. Based on the outcome or value it decides the label of the input image.

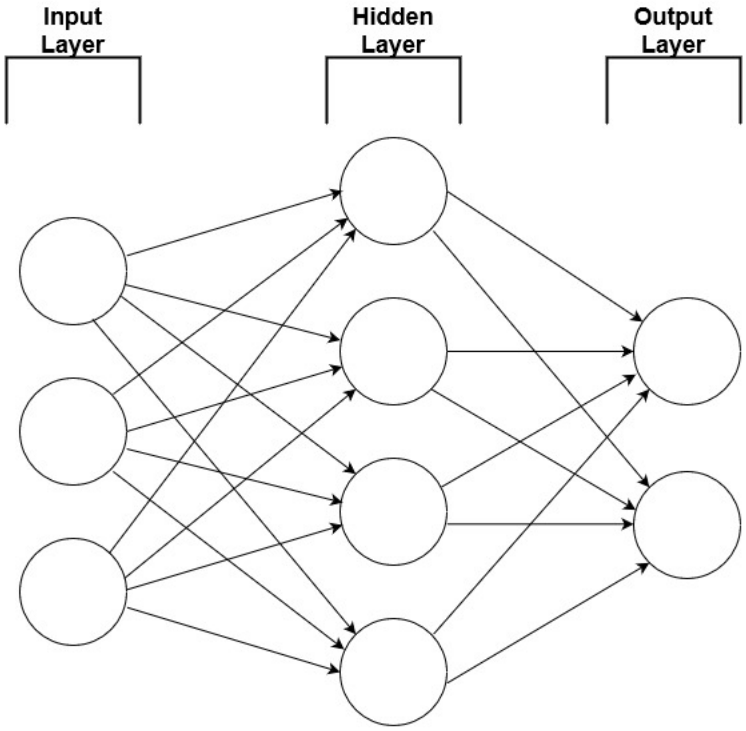


Figure 2.1: Neural Network Architecture

Convolutional Neural Network is one of the most popular variants of the Neural Network. As the name suggests the concept of Neural Network came from the idea of how the neurons (functional unit of the nervous system) function in a human brain. A typical Neural Network model can be segmented into three different parts: input layer, hidden layer and output layer. The input is provided in the input layer. Based on the type of Neural Network used the number of the hidden layers varies. If the number of layers in a Neural Network is more than three it's known as Deep Neural Network [15]. Each of these layers consists of nodes or neurons. The main computational work is done on the nodes or neurons. Neurons in each layer take input to perform computation and pass it to the next layer of neurons. In each layer the neurons take input and sum it up with the product of weights and pass it to the activation function. The activation function determines whether the output of that neuron should be passed on the next layer or not. It's similar to the function of neurons in the human brain where the neuron fires if it gets enough stimuli. Finally, after all the computation, the output layer gives the output.

While solving an image-related problem, CNN comes first inline. CNN is inspired by the idea of the human visual cortex. Where the neurons in the visual cortex are responsible for the small regions in the receptive field. In 1962 Wiesel and Hubel researched sensory processing and found that in the presence of only some certain patterns some neurons responded [16]. This research acted as an important factor behind the notion of the convolutional neural network.

One of the upper hand of CNN is that it identifies important features without the intervention of humans. To do so the architecture of CNN plays the most important role.

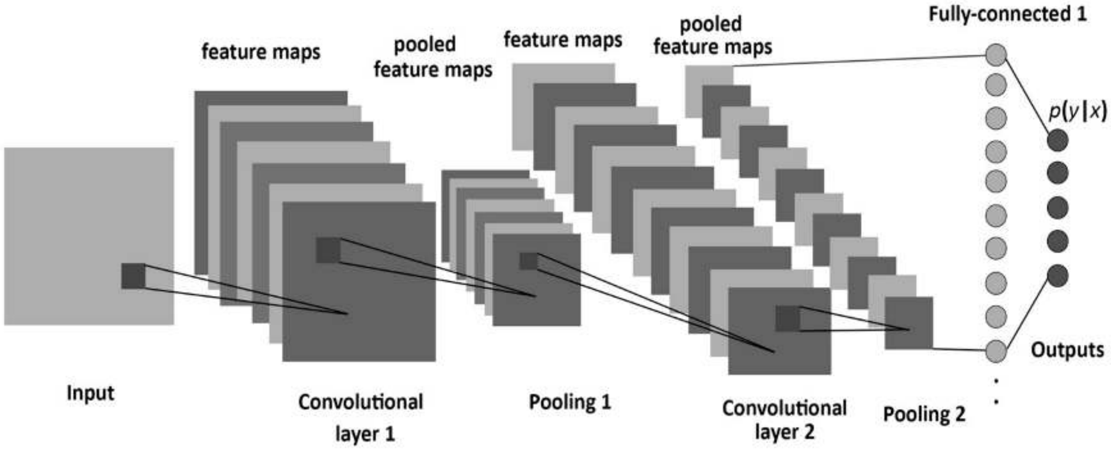


Figure 2.2: CNN layered architecture [17]

The layers are described in depth in the subparts each addressing their own.

2.7.1 Input Layer

In the input layer of a CNN image data is passed. An image is nothing but a three-dimensional matrix, where the data for the image's height, width and depth is provided. An image can belong to a number of color spaces. Greyscale, RGB, CMYK, HSV and so on. The image is separated in respective color channels according to the color space it belongs to. For a RGB image the image is separated into 3 channels. Red, Green and Blue. Each channel is just a square grid of numbered matrices containing values of the respective shade it contains. An image contains detailed information that gets computationally intensive once this reaches dimension. The task of CNN is to simplify the readability by applying filters on it. Once the image is separated into color channels according to its color space, filters are applied on it in the hidden layer and the information is kept from being lost and at the same time reduced in size, to make the processing easier. After the input layer CNN passes the image onto the hidden layer for applying filters and making it simpler for machine readability.

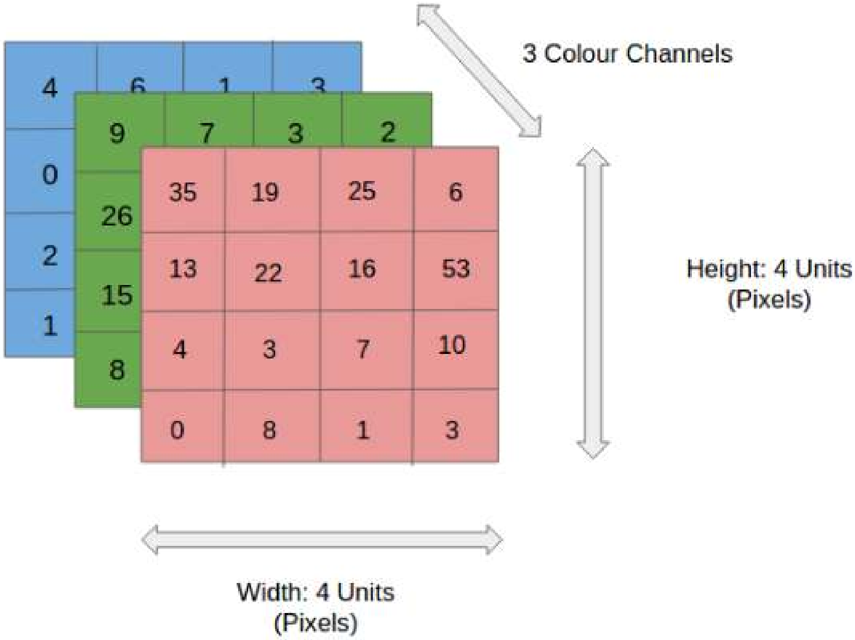


Figure 2.3: RGB image separation in color planes [18]

2.7.2 Hidden Layer

The hidden layer that results in the output image from the input image. The hidden layer basically terminates all the functionality to execute the job or algorithm. In CNN the hidden layer mainly consists of Convolutional, fully connected, pooling layer and more.

Convolutional Layer

The convolutional layer is the main building block of CNN. In this layer a mathematical computation is done between the input image and the filter or kernel producing an output image. The filter is slid over the input image by stride and continues. In each step of convolution, the number of steps taken is known as a stride. ReLU activation function is applied to the output of convolution to convert all the negative values to zero. In the convolutional layer after the computation, the edges are detected.

Pooling Layer

After the convolutional layer comes the pooling layer, in this layer the dimension of the input image is reduced. In this layer the image is downsampled while the depth is unchanged. Due to the pooling operation, the time for training reduces significantly. It also helps to overcome the problem of overfitting. The most common kinds of pooling are average pooling and max pooling. Average pooling suggests that the average value from the selected portion of the image is returned. It downsamples the dimension of the given image. Max pooling returns the maximum value from a selected portion of the image. The max-pooling operation reduces the noise while downsampling the dimension of the image, so it's better than average pooling. In order to perform max-pooling, we need mainly two things, one is a grid and the other one is stride. Grid is actually the size of the pool used in pooling and stride is the number of steps it slides. After the max-pooling operation is done the height and width are reduced but the depth remains the same. Hence the important information is kept intact while reducing the size.

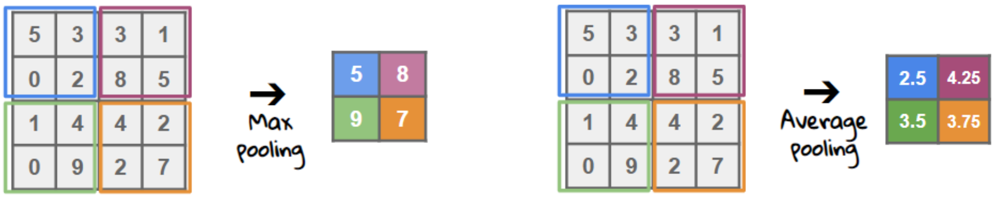


Figure 2.4: Max Pooling and Average Pooling [19]

Fully Connected Layer

After we get the output from the pooling layer it's converted into a one-dimensional array by the input layer of the fully connected layer, this process is known as flattening. After the flattening is done output is passed into the fully connected layers. The number of fully connected layers varies from problem to problem. In the fully connected layers the nodes of one layer are fully connected with the layer of the next one. The high-level features are learned in this layer. The main purpose of this layer is to correlate the features and classify the images into different categories.

2.7.3 Output Layer

Output Layer is the final decision making layer. After being reduced in dimension the output layer produces a single value from the entire input image. The generated value is then compared with the earlier stored values from the training dataset. The generated value must match with either of the trained value or a close value. The value residing closest is taken as a decision or label for the input image.

Chapter 3

Methodology and Implementation

3.1 System Architecture

Our whole system mainly consists of three sub-systems which are

- Dataset processing
- CNN Model designing, training and testing
- Real time data capture and Generating voice output

For this research, our first priority was to ensure a rich dataset and so we started collecting all the available datasets related to sign language. We also created our own dataset taking samples from different people performing multiple signs. We processed and labeled those data in such a way that it was optimized for our CNN model. We created a Sequential CNN model with multiple hidden layers and split the data into training and testing. After performing several epochs, our model was performing well and ready for testing. It had a high accuracy in most cases but was mixing up some very similar signs which is a common error in CNN. After readying our model we did some real time testing by taking input from a camera and predict accordingly. Since we are trying to build full sentences, an algorithm was formed where a user can perform multiple signs in front of the camera sequentially. Finally, a voice output is produced though Google Translator API and thus we convert the sign language to a spoken language. A diagram of the system is given below –

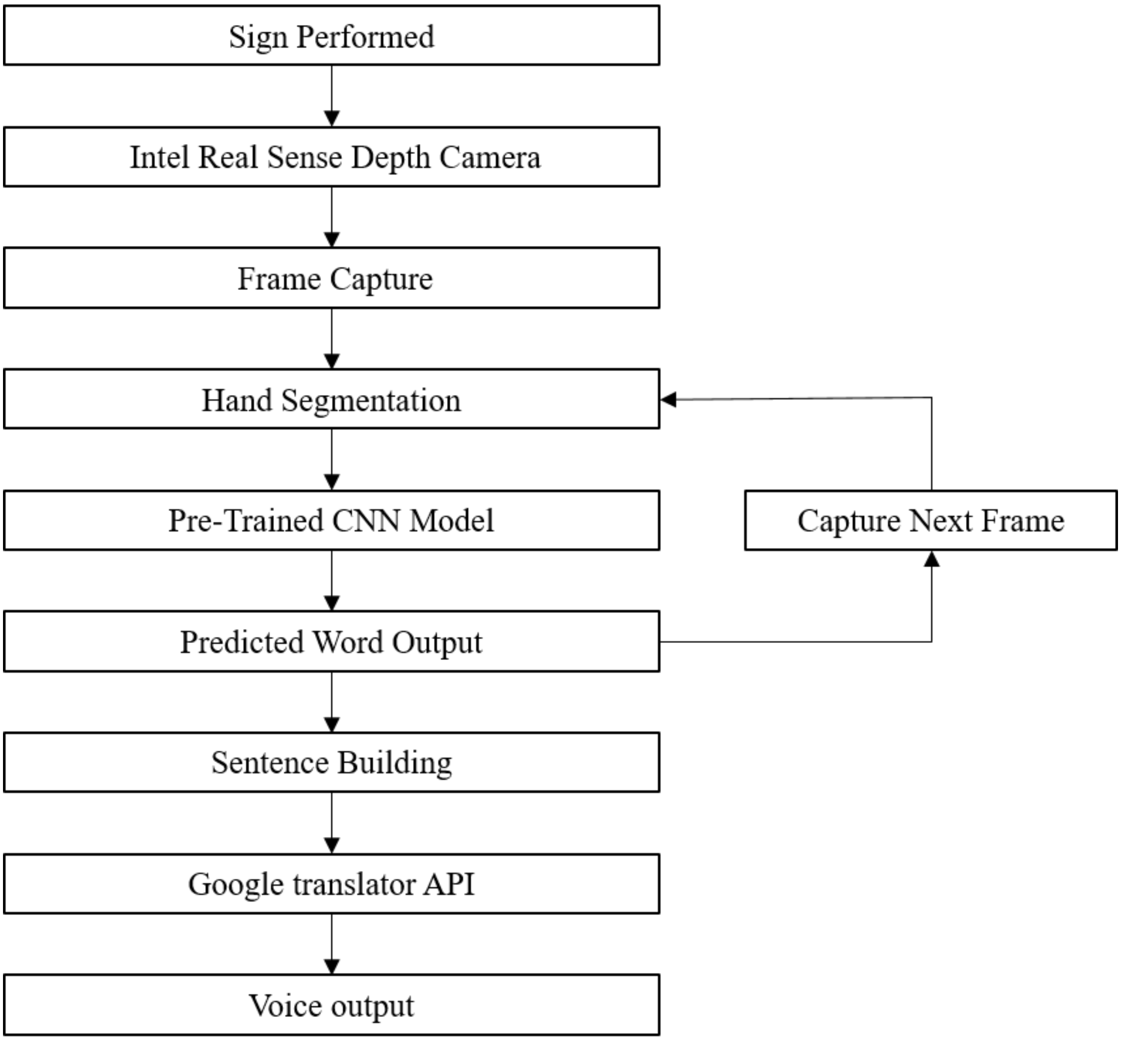


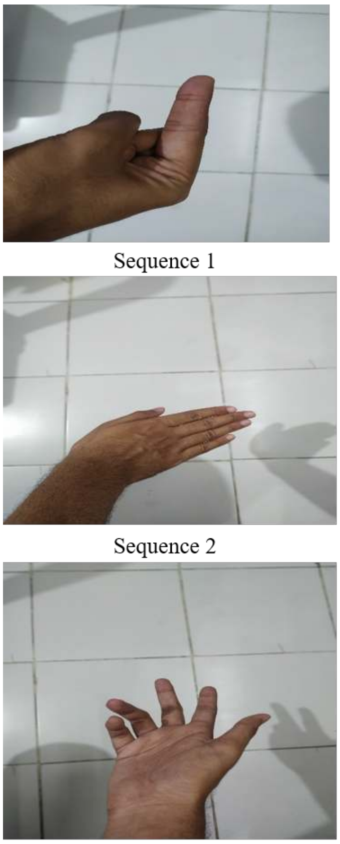
Figure 3.1: System Architecture

3.2 Dataset

Now we want quantitative way of splitting the data-set by using a particular attribute. We can use a measure called Information Gain, which calculates the reduction in entropy that would result in split-ting the data on an attribute, A. Information Gain is actually a procedure to select the particular attribute to be a decision node of a decision tree.

3.2.1 Data Collection

We have created our own dataset with meaningful Bangla sign language words. We also collected some datasets from Kaggle and GitHub [20][21][22]. We use those datasets to train and test our model. Here is an example of our dataset which is given below,



Sequence 1

Sequence 2

Sequence 3

Figure 3.2: Dataset Preparation

Here Sequence 1 contains Bangla sign language word which means "Apni". In

English which is similar to “You”. Then Sequence 2 contains “Kemon”, which is similar to “How”. At last Sequence 3 has “Achen”. These three figure all together makes a complete sentence. The sentence represents “Apni Kemon Achen ?”. In English that means “How are you ?”. Like these three words we have created more words and add them in our dataset.

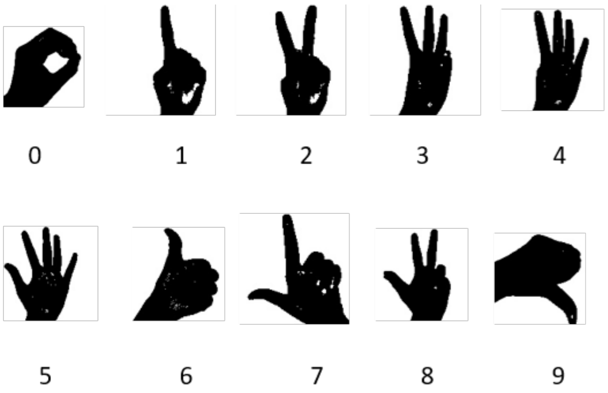


Figure 3.3: Labeling Data As Per The Sign

We have collected our raw Bangla hand sign data from our friends. They helped us to create our own dataset. Like previous figure they helped us to make more bangla sentences with Bangla sign language words. Figure 3.3 has data from datsets. We choose Bangla numbers from there. Those datasets are enriched with Bangla numbers from 0 to 9. For that reason we did not create any Bangla sign numbers from 0 to 9. There are also Bangla alphabets. We just use those data from their to train and test our model. Moreover getting the accuracy higher. Those dataset contains image from different angle. They took the pictures of those from different angle. We just took the one image for our data and we automatic get the images of that picture from different angle. The method we used here is data augmentation. For this reason we spent few times to build our dataset. The datasets, which we collected from the Kaggle and GitHub, those datasets have only alphabets and numbers. That is why we create Bangla sign language words dataset.

3.2.2 Data Manipulation and Labeling

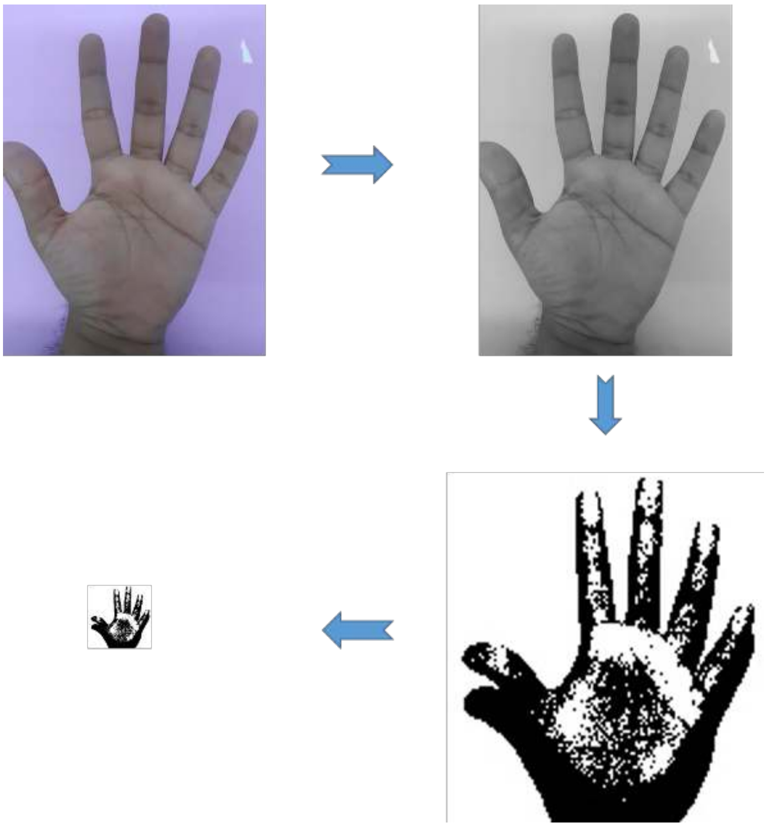


Figure 3.4: Data Manipulation

Manipulation

We have to work with a lot of raw data's, these data's were unorganized at first. So we had to take these unorganized data and change the type of information for machine to understand. And for this manipulation to work the data must be in uniform structure to work correctly. But to be usable by the humans the data needs to be translated and manipulated in such a way that it becomes much refined and easy to work with. When we need to work with a huge amount of data we must have to do manipulation of the data else the dataset will become more complex and hard to work with, because when the amount of data increases the use of hardware as well as storage increases. So to make our machine work efficiently we must manipulate the data and organize it properly.

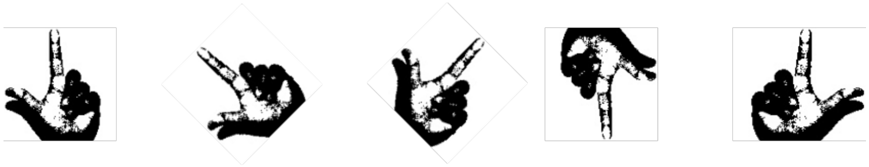


Figure 3.5: Data Augmentation

Firstly we took the colored image for our data manipulation, but as the colored picture has 3 channels which are known as RGB, so for a simple 256*256*3 pixel colored image the pixel calculation output becomes very high which is very tough to process in neural network. As this processing will require a lot of time and also it will put a huge pressure on the hardware also. So we converted the colored image to a grayscale image to reduce the channel. Grayscale image has only 1 scale which has the value of either black and white. This value ranges from 0 to 255 where 0 is pure black and 255 is pure white. If the value is between any of this number it will be the mix of black and white shading.

Though this process reduces the calculation but still it is very complex to process. So we used Binary Threshold Image to reduce the complexity more. We took a mid-value of 127 for our binary image. So that, the values that are below 127 in grayscale will be considered as black pixels and the values that are above 127 will be considered as white pixels. In this way we have converted our colored image into only two shades of color which is known as Binary Image. This image is easy to manipulate as the size is very low for this image. In this binary image the size becomes 28*28*1. So it becomes very easy to manipulate and work on. After manipulating the data we need to augment the data. As we know that a huge amount of dataset is very important for the better performance of our model. We can enhance this performance by augmenting the data we already have. The way augmentation enhances the quality and performance of our model is enhancing the diversity of data already available for us. So it will be very significant because working with a huge amount of data is very critical and for better performance we need to keep adding the data. This is very time consuming and complex for training the machine. So by augmenting the data we do not need to add new data whereas we can train with the existing data we have.

There are actually many processes and techniques by which we can augment the data. The techniques that we used in our model are random brightness, horizontal flip, vertical flip, random channel shift, random zoom and random shearing. There are many purposes of using these techniques for training the data. The main purpose of this is to train our dataset with the newer plausible examples. Firstly random brightness fixes the brightness of our picture by a random factor which is very important because we need to recognize our image at different brightness. The horizontal and vertical shift methods are used to look at the image at different angles, because our machine might capture the image at different point of view. So we need to rotate the data in such a way that if flipped vertically or horizontally the model can recognize the sign efficiently. We used random channel shift method to shift the channeling factors of our image in such a way that for the same image but with different channeling factors the image gets recognized efficiently by the model. Now shearing is used if the bounding border of our image transforms.

So basically we took different approach to the hand sign to recognize it effectively and efficiently so that when newer data comes we can easily identify it and work with the augmented data rather than inputting the new data every time.

Labeling

Data labeling is an important part of data preprocessing. This is important in our case because in our model we are using supervised learning where in both input and output data we are labeling the data for classification to create a basis for our model for future data processing [23].We classified the hand signs separately. For this we separated the numbers 1 to 10 and created folders for these signs. For example, we labeled folder 1 for all the signs that implies the number 1.By following this order we created folders 1 to 10 to classify the numbers from 1 to 10.By taking the dataset from internet we also classified the vowels and consonants of Bangla language and recognized them. We also used our own dataset images to recognize the hand signs. Primarily everything we have done to recognize these data's is done using machine learning methods. We also labeled the Bangla numbers from 0 to 9 folders. From the 10th folder we started recognizing the vowels and consonants sequentially. In our own dataset the sentences we have added gets roughly 30 words per sentence and we got 90 signs in total. By this the number of our dataset gets to 4500.

3.3 Model Design

There are different kinds pre trained models available already but every model has its flaws and so we decided to make our own custom CNN model. We chose a sequential model which had multiple hidden layers one stacked after another for feature extraction. The Keras Python library makes making profound learning models quick and simple. The consecutive API permits us to make models layer-by-layer for most issues.

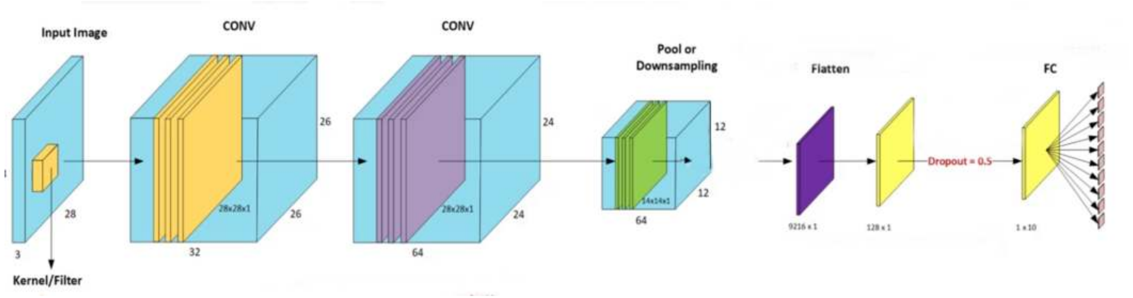


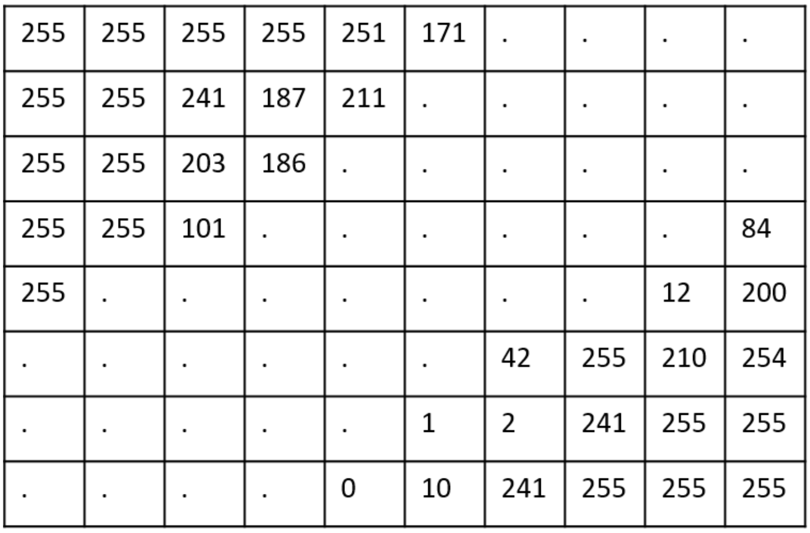
Figure 3.6: Custom CNN Model

Our input images are 28x28x1 as previously mentioned that we have only one colour channel here consisting of only black and white and all the images. All the image inputs are resized to 28x28 and converted from RGB to binary images. In our first hidden convolutional layer, we have used 32 filters and a kernel of 3x3 with the activation function 'relu'.

3.3.1 Kernel

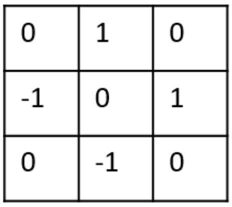
Filters are the sum of learnable weights which are calculated through backpropagation algorithm. By applying multiple filters, we get multiple features of an image. The first convolutional layers which are at the beginning of a network is generally knows as low level filters and the later filters are high level filters. Generating patters by updating weights gradually, low level filters can detect things like edge ,color ,density , sharpness and extract features accordingly while high level filters learn larger spatial patterns.

Figure 3.7: Input Image



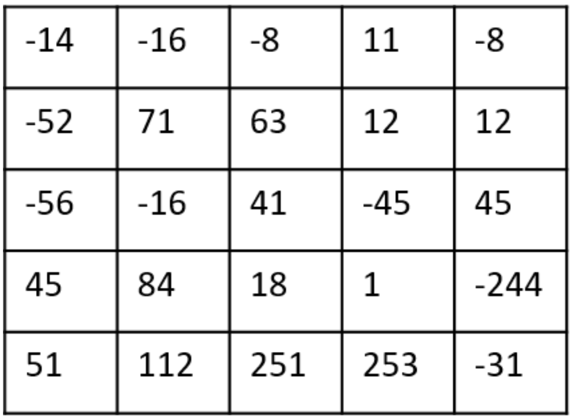
255	255	255	255	251	171	.	.	.	.
255	255	241	187	211	.	.	.	.	.
255	255	203	186	.	.	.	.	.	.
255	255	101	.	.	.	.	.	.	84
255	.	.	.	.	.	.	.	12	200
.	.	.	.	.	.	42	255	210	254
.	.	.	.	.	1	2	241	255	255
.	.	.	.	0	10	241	255	255	255

Figure 3.8: 28x28x1 Image Matrix



0	1	0
-1	0	1
0	-1	0

Figure 3.9: 3x3 Filter Map



-14	-16	-8	11	-8
-52	71	63	12	12
-56	-16	41	-45	45
45	84	18	1	-244
51	112	251	253	-31

Figure 3.10: Output Matrix

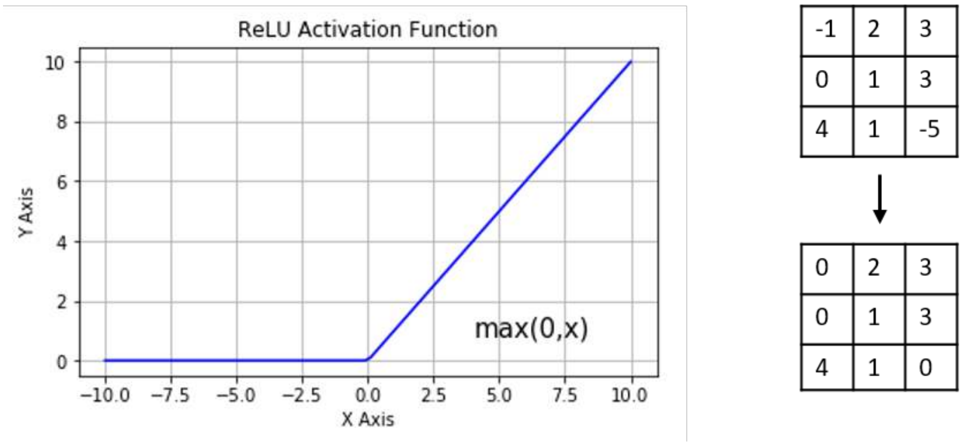
These filter maps generates features depending on the labeled images by multiplying each pixel's value with the corresponding filter values and sum the values.

3.3.2 Stride and Padding

Stride denotes how many steps we are moving in each steps in convolution. It is the number of pixels that shifts over the input matrix. When the stride is 1 then we move the filters to 1 pixel at a time and when it is 2, we move 2 pixels. Stride can reduce the size of the input matrix however, we always do not want that since it can harm useful features extraction therefore there is also a technique called zero-padding which allows us to extract features without decreasing the size. . We can observe that the size of output is smaller than input. To maintain the dimension of output as in input, we use padding. Padding is a process of adding zeros to the input matrix [24] symmetrically. It is a good practice to use 1 or 2 Stride values. Hence we set the stride value to 2. By default it is one

3.3.3 Activation Function

In a neural network, the activation function is responsible for transforming the summed weighted input from the node into the activation of the node or output for that input. Activation function decides, whether a neuron should be activated or not by calculating weighted sum and further adding bias with it. The purpose of the activation function is to introduce non-linearity into the output of a neuron. There are different kinds of activation functions like Binary Step, Softmax, Sigmoid , Relu , Leaky Relu , Swish and so on. We have used Relu activation function in our convolutional layers and Softmax at the outer layer.



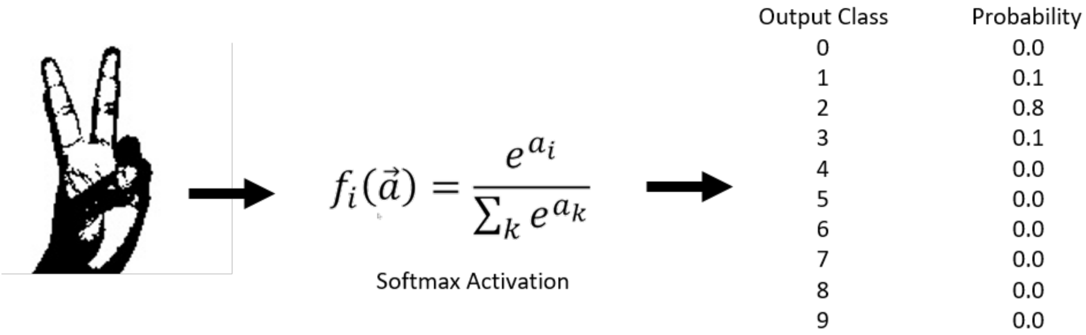
-1	2	3
0	1	3
4	1	-5

0	2	3
0	1	3
4	1	0

Figure 3.11: Relu Activation Function

Relu : The ReLU work is another non-straight activation function that has picked up fame in the deep learning space. ReLU represents Rectified Linear Unit. The principle favorable position of utilizing the ReLU fucntion over other activation function is that it does not initiate all the neurons at the equivalent time. This implies that the neurons may be deactivated if the yield of the linear change is under 0. For the negative input values, the outcome is zero that implies the neuron does not get initiated. Since just a specific number of neurons are enacted, the ReLU function is undeniably more computationally productive when contrasted with the sigmoid and tanh work [25].

Softmax: Dissimilar to some other activation function, softmax is generally applied to the last layer of neural system. It is utilized to create a non-normalized output of a system and basically gives us a probability distribution. Softamx activation function is the best activation function for CNN models since it visualized the predicted output for all the potential classes and helpes us to distinguish the misplaced outputs. Every component in the output relies upon the whole arrangement of components of the information. Since hand signs can turn out to be extremely comparative now and again , that is the reason an activation function like softmax is an unquestionable requirement for our neural system.



$$f_i(\vec{a}) = \frac{e^{a_i}}{\sum_k e^{a_k}}$$

Output Class	Probability
0	0.0
1	0.1
2	0.8
3	0.1
4	0.0
5	0.0
6	0.0
7	0.0
8	0.0
9	0.0

Figure 3.12: Softmax Activation

The above figure 3.12 shows that the softmax function turns input into probabilities. In the equation, ‘a’ refers to each element in input vector. It takes the exponents of each input and normalizes each number by the sum of exponents. Therefore, the sum of the output vector is always one. From the diagram, the input data is a sign of the number two and the output prediction justifies that it is a picture of class 2 since it has 0.8 probability. So we can assume that this sign belongs to the hand sign which is labeled as 2.

3.3.4 Loss Function

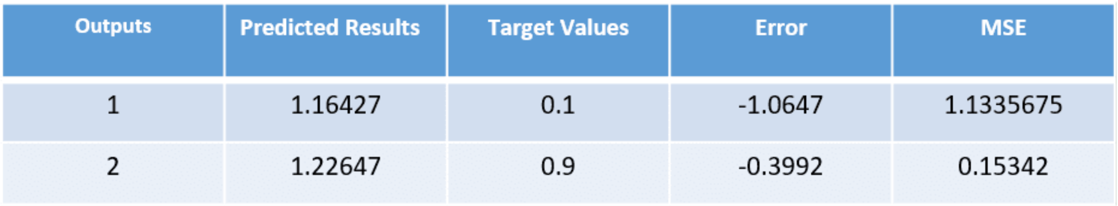
For a neural network we cannot compute the ideal weights; there are too many unknowns. Instead, the learning problem is cast as a search or optimization question, and an algorithm is used to explore the space of possible weight sets that the model can use to make good or good enough predictions. A neural network model is usually trained using the stochastic gradient descent optimization algorithm and weights are modified using the error algorithm backpropagation. Hence the function which minimized the error is known as loss function. A loss function is extremely simple: it's an assessment method of how well the algorithms models dataset. If estimates are totally wrong, a higher number will be given out by the loss function. When they are pretty decent, a smaller number will be generated. We used the two major types of loss functions to train our neural network model such as cross-entropy and mean squared error.

Mean Squared Error (MSE): It gives the square difference between the true and the predicted values. As the result is squared, so we always get a positive value.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(f_i - y_i)^2$$

Figure 3.13: MSE Formula

In the formula above, N is the total number of data, fi is the predicted value which is returned by the model and yi is the actual value. Since, we need to fix the random number generation of weights, MSE calculates the error and helps the model to adjust weights accordingly.



Outputs	Predicted Results	Target Values	Error	MSE
1	1.16427	0.1	-1.0647	1.1335675
2	1.22647	0.9	-0.3992	0.15342

Figure 3.14: MSE Table Data

Cross-Entropy: Cross-entropy or log loss measures the efficiency of a classification model with a probability value between 0 and 1. Loss of cross-entropy increases as the predicted likelihood varies from the true label. Cross entropy is definitely a must needed loss function for visualizing a probability output.

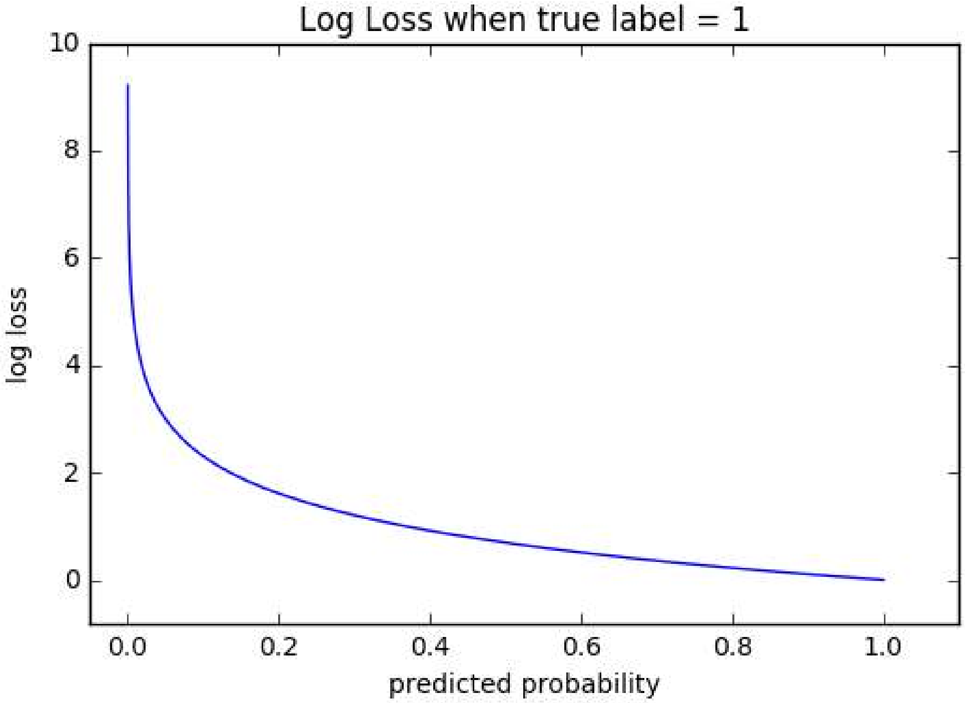


Figure 3.15: Cross Entropy

The graph, figure 3.15 shows us the range of loss values given the real observation. As the expected output reaches 1, log loss decreases gradually. However, as the expected output reduces, log losses increase exponentially. Log loss penalizes all forms of mistake, but particularly those predictions that are both positive and incorrect.

3.4 Training and Testing

In machine learning, the whole dataset is usually divided into two parts which is training dataset and testing dataset. The training set includes known value and the model learns from this data in order to test the remaining data later on. As for our model, we split the total data to a ratio of 4:1 where 80% of the data in training dataset and the remaining 20% data in testing and validation dataset such as x_train, y_train, x_test ,y_test . The x_train and x_test contains the actual data which is a vector matrix and y_train and y_test contains the class labels for the corresponding data. This ratio enables us to train our model with efficiently and also reduces the chance of overfitting and under fitting which is a common issue in neural net.

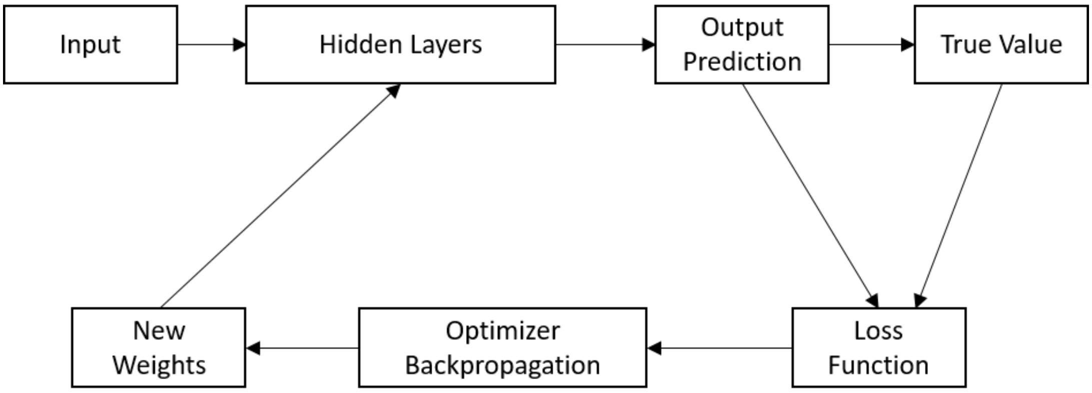


Figure 3.16: Training Process

From the above figure 3.16, we can figure out the whole training process. Through multiple epochs the random weights get updated with the help of loss function which is MSE in our case. Then it goes through the backpropagation process once again and hence the weights get updated every epochs. Depending on the GPU performance we set a batch size of 64 and continued the iteration until an epoch was completed.

3.4.1 Overfitting & Under fitting

Overfitting is a common problem in neural net which means generalizing the output on the training data. It means that our model has trained so well that its prediction has been perfectly fit to our training data set. It mainly occurs if the model becomes complex or the data portion in the training dataset is very low. The model can predict output on the training dataset very accurately but fails to give prediction on the test data or misclassifies the input data.

In the above figure, the black line is the decision boundary and as we can see in the third diagram that the decision boundary fit the model in such a way that it can easily misclassify the data between red dots and green dots. On the other hand,

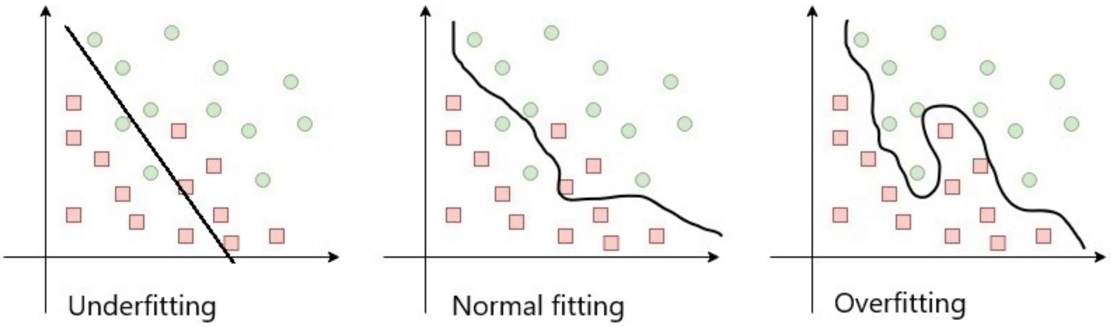


Figure 3.17: Graph of Overfitting, Normal Fitting and Underfitting Model

under fitting occurs when the model becomes way too simple. It means that the model unable to detect the patterns in the training data. It gives us a straight linear line and cannot distinguish between the two outputs. To overcome the problem of overfitting and under fitting, we can use less weights to get smaller/less deep Neural Network which is called regularization. There are few types of regularization and we used cross validation and drop out.

3.4.2 Cross Validation & Dropout

Cross validation or also known as k-fold cross validation is a method of training where we split our dataset into k folds instead of a typical training and test split. For example, if we use 5 folds then we train on 4 and use the 5th final fold as our test. We then train on the other 4 folds and test on another. After that we use the average weights across coming out of each cycle. Cross Validation reduces overfitting but slows the training process. On the other hand, Dropout refers to dropping nodes both hidden and visible in a neural network with the aim of reducing overfitting. In training certain parts of the neural network are ignored during some forward and backward propagations. Dropout is an approach to regularization in neural networks which helps reducing interdependent learning amongst the neurons. Thus the neural network learns more robust or meaningful features. In Dropout we set a parameter that sets the probability of which nodes are kept or for those are dropped. Dropout almost doubles the time to converge in training. But it is definitely worth the time as it removes the possibility that a model will be over fitted.

Moreover, the bigger the size of data the lesser the chance of the model to be over fit. That is why we used data augmentation while processing the data as it increased the total amount of data by a large margin and also to provide better accuracy on the model.

3.5 Real Time Data Capture

As previously mentioned, we captured real time data and used that data in our pre trained CNN model to predict an output. First of all, we took a powerful webcam, in our case it was Intel real sense and used it to capture data from a person performing different hand signs.

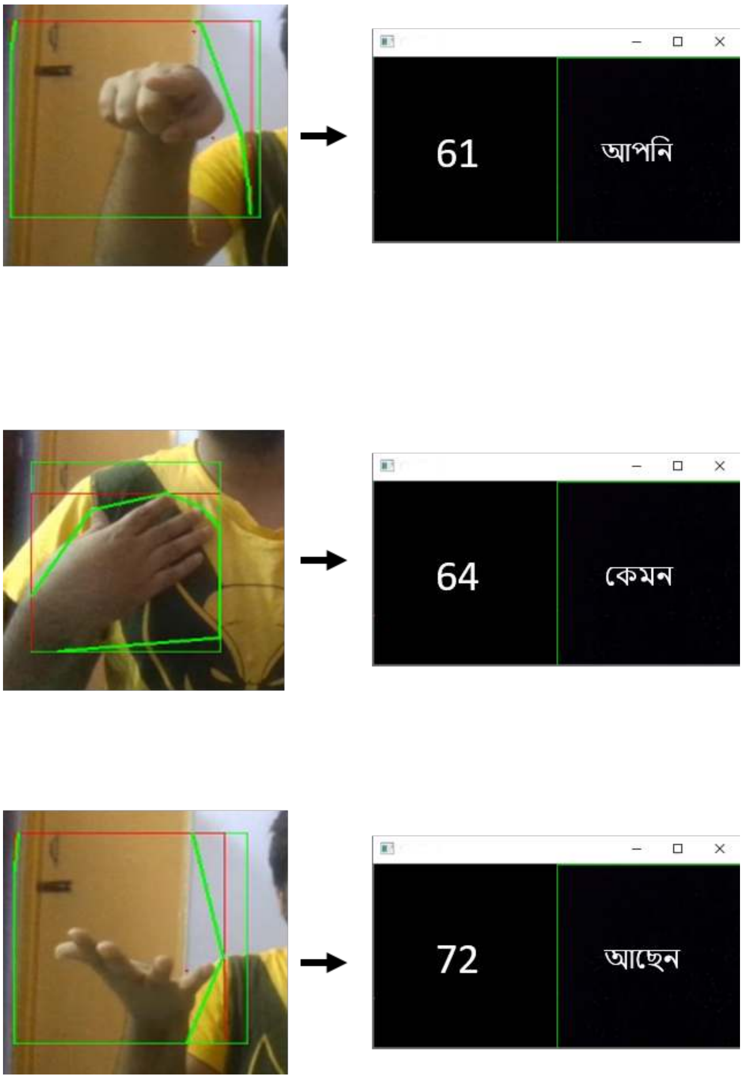


Figure 3.18: Real Time Capture

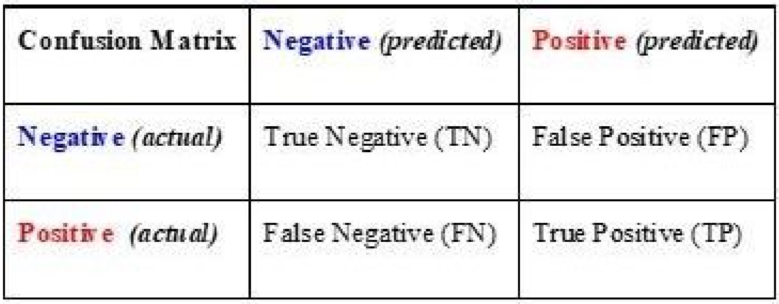
From the above figure 3.18, we can see that real time data is being captured. Firstly, a hand sign is performed in a specific region the bounding box. Then the hand data is segmented. A single frame is captured from the video and processed for the prediction. It is turned into a grayscale image and then to a binary image where it only contains black and white pixels. Moreover, it is resized into 28*28*1 matrix and passed to the model for prediction. Since we are trying to build a sentence here, there is an algorithm developed which allows us to pause the capture and then resume again to capture a new frame. Through this technique, the previously captured frame is saved in a variable and the next captured frame is added to the sequence. Later on, all the frames are places one after another and hence a sentence is formed. Due to such processing, our model can accurately predict the signs. Through this method the counting of contours are not needed and hand segmentation is done without the need of background removal and hand color detection. Although techniques like contour finding and drawing a convex hull could improve the detection of the hand signs but it was unnecessary since we are already processing the image. We have 90 classes in total and each class represents a single sign. Every class label is mapped with a Bangla word and the word can also be displayed in the program. For voice output generation, we have used the library google trans. Google Trans is free and unlimited python library which implements Google Translate API. With the help of this library we can translate the output according to our needs and also generate a voice output for the user. Through this process, a hand sign is finally converted in to a voice output and we can also build meaningful sentence by performing signs sequentially.

Chapter 4

Result and Analysis

4.1 Confusion Matrix

Entire system of machines that tries to replicate human senses works on prediction based on the earlier learning or training that it receives. As machines do not own any senses of their own they are provided with virtual processing of compare and predict formulation that allows them to perceive an object and define that based on earlier training. Not all systems or models can provide a precise or accurate prediction. To determine whether the system is reliable enough to function on it's own or needs to be trained more a test is performed to see the accuracy and precision of it's prediction. A simple and effective tool for determining this is confusion matrix. A general structure of confusion matrix for two classes is a 2x2 matrix. With column and row having actual and predicted values. A visual representation is as follows



Confusion Matrix	Negative *(predicted)*	Positive *(predicted)*
Negative *(actual)*	True Negative (TN)	False Positive (FP)
Positive *(actual)*	False Negative (FN)	True Positive (TP)

Figure 4.1: Confusion matrix for two classes is a 2x2 matrix

Here, the term True Negative (TN) implied that both the actual and predicted result is negative. Same as this, True Positive(TP) also implies that actual and predicted results both are positive. For False Positive (FP) the algorithm or model actually predicted positive but the actual result was negative. Lastly, False Negative (FN) is when the actual result is positive but the algorithm or system predicted a negative result.

From a confusion matrix, we can obtain some further advanced classification metrics. Parameters that can be calculated from a confusion matrix are Precision, F1 score, Accuracy, Recall or Sensitivity, Specificity etc.

Precision: It is a ratio of the correctly classified positive examples with the total number of positive predicted examples.

$$Precision = TP/(TP + FP) \tag{4.1}$$

Sensitivity/Recall: Also known as the true positive rate is the measure of positive examples that are labeled as positive by the algorithm or model.

$$Recall = TP/(TP + FN) \tag{4.2}$$

Specificity: Opposite to recall, specificity measures the rate of true negative. This is the measure of negative examples labeled as negative by the algorithm.

$$Specificity = TN/(TN + FP) \tag{4.3}$$

F1 Score: This is the weighted measure of recall and precision.

$$F1score = 2 * ((Precision * Recall)/(Precision + Recall)) \tag{4.4}$$

Accuracy: Proportion of the total number of predictions that are actually correct.

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \tag{4.5}$$

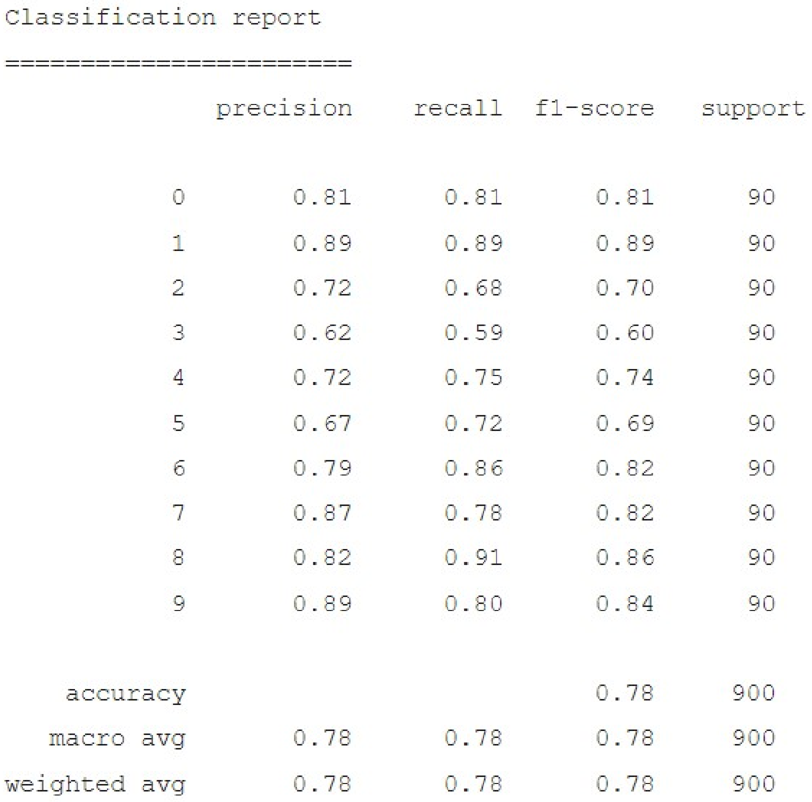
For any algorithm the aforementioned parameters are used to judge and define the strength of that algorithm. These are the major parameters to form a basis to evaluate the quality of an algorithm.

Micro Average: The averaging of the total true positives, false negatives and false positives samples is Micro Average.

Micro Average: Macro Average is the calculated metrics for each label and later finding their unweighted mean.

Weighted Average: Weighted Average is calculating the metrics for each label and finding their average weight by support.

To find the specifications of our constructed model, the classification report here would be reviewed and evaluated according to the generated value in the classification result from the model.



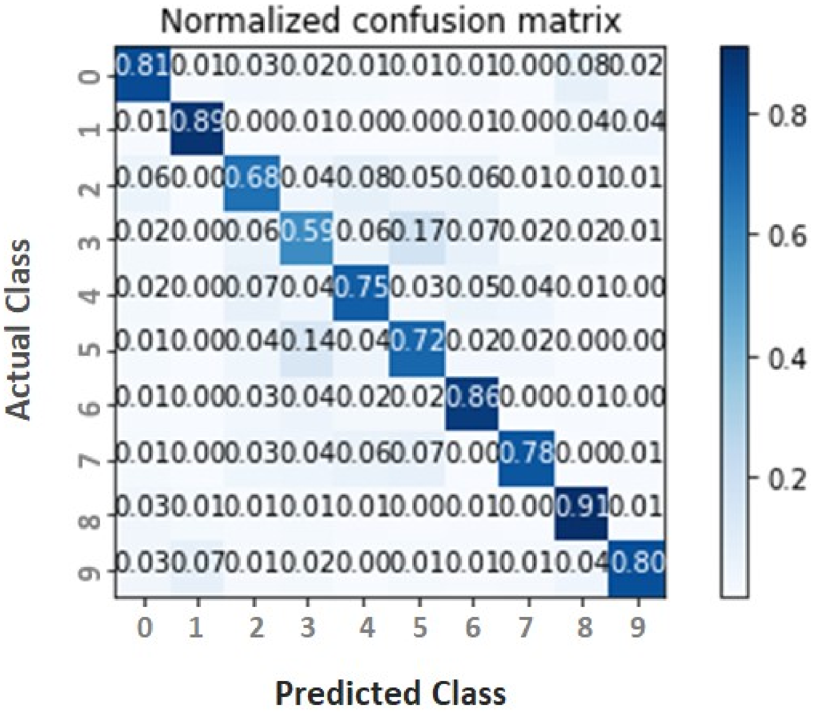
```
Classification report
======================
           precision    recall  f1-score   support

       0       0.81      0.81      0.81        90
       1       0.89      0.89      0.89        90
       2       0.72      0.68      0.70        90
       3       0.62      0.59      0.60        90
       4       0.72      0.75      0.74        90
       5       0.67      0.72      0.69        90
       6       0.79      0.86      0.82        90
       7       0.87      0.78      0.82        90
       8       0.82      0.91      0.86        90
       9       0.89      0.80      0.84        90

accuracy                           0.78       900
macro avg      0.78      0.78      0.78       900
weighted avg   0.78      0.78      0.78       900
```

Figure 4.2: Classification Report

From the aforementioned Classification Report, we can see that precision, recall and f1-score are all the same for word_1 which is 0.81. Again for Word_2 all three values are the same and that is 0.89. Word_3 has a precision of 0.72, recall of 0.68 and f1-score of 0.70. For word_4 the values drop. Here precision is 0.62, recall is 0.59 and f1-score is 0.60. Again at Word_5 the scores rise and precision, recall and f1-score are 0.72, 0.75 and 0.74 respectively. Word_6 has a medium score for precision having 0.67 and f1-score of 0.69 but an above average recall score of 0.72. From Word_7 to Word_10 we can see a gradual rise in the precision score which are 0.79, 0.87, 0.82 and 0.89 respectively. Again the f1-score also maintains similar scores 0.82, 0.82,0.86 and 0.84. The recall scores vary only at Word_9 which is 0.91 but for other Word_7 to Word_10 they have almost similar values as 0.78,0.86 and 0.80. From these scores we calculate the macro average and weighted average of 0.78.



	0	1	2	3	4	5	6	7	8	9
0	0.81	0.01	0.03	0.02	0.01	0.01	0.01	0.00	0.08	0.02
1	0.01	0.89	0.00	0.01	0.00	0.00	0.01	0.00	0.04	0.04
2	0.06	0.00	0.68	0.04	0.08	0.05	0.06	0.01	0.01	0.01
3	0.02	0.00	0.06	0.59	0.06	0.17	0.07	0.02	0.02	0.01
4	0.02	0.00	0.07	0.04	0.75	0.03	0.05	0.04	0.01	0.00
5	0.01	0.00	0.04	0.14	0.04	0.72	0.02	0.02	0.00	0.00
6	0.01	0.00	0.03	0.04	0.02	0.02	0.86	0.00	0.01	0.00
7	0.01	0.00	0.03	0.04	0.06	0.07	0.00	0.78	0.00	0.01
8	0.03	0.01	0.01	0.01	0.01	0.00	0.01	0.00	0.91	0.01
9	0.03	0.07	0.01	0.02	0.00	0.01	0.01	0.01	0.04	0.80

Figure 4.3: Confusion Matrix

In the above figure, a confusion matrix is attached for the proposed model. The earlier classification report was generated at the same phase. While we can see a highlighter beside marking the range from 0 to 1, we can point out the marking in the matrix and analyze the outcome for the test result. In X-axis the predicted labels are shown, while the Y-axis has the plotted values for true or actual labels. The diagonal section which joins the actual and prediction for the same class represents the correct predictions. The rate of accurate prediction here is 0.78 or 78%. The highest accuracy is for word_9 with a rate of 0.91 or 91% while lowest prediction rate is for Word_4 which is 0.59 or 59%. This produces an average accuracy of 0.78 or 78% which stands for a medium performing system. To have a better outcome, the feeding data needs to be more. The classes providing lower accuracy rate should have more data feed in the system.

4.2 Other pre-trained models

There are many pre-trained models already and some of the most recognized models are VG16, VG19, InceptionV3 and ResNet50 . These models are really efficient for predicting images as such models are trained with millions of data and has over thousand classes. The problem is, these models are trained with images of different types such as bird,animal,vehicel,objects etc. But as for our model, we needed to be specific for only hand signs. For example, we used the model VG19 to predict a hand sign from our dataset and it correctly predicted that it was hand but cannot determine the specific sign as it was not trained for that kind of output. If we were to train the VG19 model, we needed to modify the layers to our need but it was complex to do as VG16 and has 16 layers and VG19 has almost 19 layers which uses a 3x3 kernel. It was built such a way for prediction images of different objects and extract feature for different kinds of images. Moreover, it takes way too much to time to train on these models due to having so many layers. So the performance becomes really slow and training takes a huge amount of time. Though these models are feature enriched and considered to be the best models out there but for researches where similar types of images are trained, it is better to use a smaller network and modify it to the point. So using pre-trained models and modifying them was not a smart approach for us, hence we built our simple model which had layers according to our need.

We tried tweaking some values to the VG19 model and removed some of the layers from it and trained it with our data but the prediction was not good as our model as it gave a 70% accuracy to the data and we had almost 90% accurate for most of our data. So our model was better performing than the pre-trained models.

4.3 Limitation and Future work

Though, the model was created to efficiently recognize hand signs in real time. But on the process we faced some difficulties. The first problem we faced was it is very complex to recognize critical hand signs. Here, critical hand signs mean signs with a flow. In sign language sometimes to express something the hearing impaired people use a certain movements which is very complicated to recognize via the resources available. So, our system finds difficulties with this kind of hand signs. Another problem we faced is with training the dataset. As, we have limited GPU power it is not possible to train 90 classes of data at once. So we trained our data one by one which is very time consuming. However we also found difficulties to properly manage our bounding border. Bounding border is the window or frame where we try to recognize the hand sign. However in Bangla sign language movement of the hands is not the only way to express something. There are some other factors like movement of lips, facial expression, and movement of the eyes also matter to express a meaningful sentence. As our bounding border only focuses on the hand gestures it cannot detect properly the signs that involve the body movements and facial expressions.

The main thing we want to do in future is to enrich our dataset with more hand signs. For this we need to research more about the sign language of Bangladeshi people. As we know that sign language is not universal we need to enrich our dataset more so that our system can predict more efficiently. Another method we want to apply to our system which is YOLO method also known as you only look once technique which is a real time object detection method. This is much faster than the other RCNN methods available. It follows the architecture of FCNN, it passes the image once through the FCNN (Fully Convolutional Neural Network) and the output it gives is the final prediction. We also want to develop our systems bounding border in such a way that it can detect complicated sign efficiently and successfully.

4.4 Conclusion

This paper discusses the work done on Bangla Sign Language Recognition. Considering the diverse and complicated field of Sign Language, our work spans over only a subset and part of it. Besides interpreting the signs according to the standard and legally accepted terms, the system would convert the text to speech. The system is built on an experimental round in a preliminary phase. This only works on only a few sentence recognition segmenting it into words. Currently, the dataset is also of medium level. As we can see from the result analysis part, the accuracy of this model stands on 78% which is of average level. For a properly functioning model, the accuracy rate needs to rise up. For this reason, better training is required to have results. For boosting performance it requires a rich training set. To make it useful for everyone the system requires multi-signer data. As it is in the preliminary phase the system was trained by dual-signers. The system needs to function the same irrespective of signer or user. As our aim is to provide a perfectly functioning model that will be able to interpret signs properly and generate voice output for all, the system needs to be trained and tested on multiple signers. We hope to work at depth and make a fully functional model that would be able to interpret all the registered signs in Bangla Sign Language and would have a higher accuracy rate so that it can predict the signs perfectly. Also the model is currently limited to sentences, we look forward to making it a complete interpreter of any continuous speech.

Bibliography

[1] "Deafness and hearing loss". In: *World Health Organization* (). URL: https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss.

[2] David M. Perlmutter. "What is Sign Language?" In: *LSA Archived* (Apr. 2014).

[3] Niki and Int'l Ltd. "The Different Types of Sign Language". In: *Niki's Int'l Ltd* (Sept. 2017).

[4] Fayeka Zabeen Siddiqua. "The Silent Conversation". In: *The Daily Star* (Mar. 2017).

[5] Helen Cooper et al. "Sign Language Recognition Using Sub-unit". In: *The Journal of Machine Learning Reasearch* (July 2012).

[6] B. S. Parton. "Sign language recognition and translation: a multidisciplined approach from the field of artificial intelligence". In: *Journal of deaf studies and deaf education* (2006). URL: https://www.ncbi.nlm.nih.gov/pubmed/16192405.

[7] M. M. Zaki and Shaheen S. I. "Sign language recognition using a combination of new vision based features". In: *Pattern Recognition Letters* (Mar. 2011). URL: https://dl.acm.org/doi/10.1016/j.patrec.2010.11.013.

[8] Ankita Wadhawan and Parteek Kumar. "Deep learning-based sign language recognition system for static signs". In: *Neural Comput & Applic* (Jan. 2020). URL: https://doi.org/10.1007/s00521-019-04691-y.

[9] Ghaleb F. et al. "Vision-Based Hand Gesture Spotting and Recognition Using CRF and SVM". In: *Journal of Software Engineering and Applications* (2015). URL: https://doi.org/10.4236/jsea.2015.87032.

[10] Yang Hee-Deok, Sclaroff Stan, and Lee Seong-Whan. "Sign Language Spotting with a Threshold Model Based on Conditional Random Fields". In: *IEEE transactions on pattern analysis and machine intelligence* (2009). URL: https://doi.org/10.1109/TPAMI.2008.172.

[11] "Intel RealSense". In: *Wikipedia* (2019). URL: https://en.wikipedia.org/wiki/Intel_RealSense#Intel_RealSense_Depth_Camera_D435.

[12] "OpenCV-Overview". In: *Tutorialspoint* (). URL: https://www.tutorialspoint.com/opencv/opencv_overview.htm.

[13] "OpenCV-Overview". In: *GeeksforGeeks* (). URL: https://www.geeksforgeeks.org/opencv-overview/.

[14] "About". In: *OpenCV* (). URL: https://opencv.org/about/.

[15] "A Beginner's Guide to Neural Networks and Deep Learning". In: *Pathmind* (). URL: https://pathmind.com/wiki/neural-network.

[16] A. Deshpande. “A Beginner’s Guide To Understanding Convolutional Neural Networks”. In: *A Beginner’s Guide To Understanding Convolutional Neural Networks* (2019). URL: https://adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/.

[17] V. Nigam. “Understanding Neural Networks. From neuron to RNN, CNN, and Deep Learning”. In: *Medium* (Feb. 2020). URL: https://towardsdatascience.com/understanding-neural-networks-from-neuron-to-rnn-cnn-and-deep-learning-cd88e90e0a90.

[18] S. Saha. “A Comprehensive Guide to Convolutional Neural Networks-the ELI5 way”. In: *Medium* (Dec. 2018). URL: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[19] J. Jeong. “The Most Intuitive and Easiest Guide for CNN”. In: *Medium* (July 2019). URL: https://towardsdatascience.com/the-most-intuitive-and-easiest-guide-for-convolutional-neural-network-3607be47480.

[20] Sanzid Kawsar. “Bangla-Sign-Language”. In: *GitHub* (2018). URL: https://github.com/Sanzidikawsar/Bangla-Sign-Language.

[21] Abdul Muntakim Rafi. “Bangla Sign Language Dataset”. In: *Kaggle* (Mar. 2020). URL: https://www.kaggle.com/muntakimrafi/bengali-sign-language-dataset.

[22] Muhammad Khalid. “Sign Language for Alphabets”. In: *Kaggle* (Nov. 2019). URL: https://www.kaggle.com/muntakimrafi/bengali-sign-language-dataset.

[23] Rouse Margaret. “What Is Data Labeling ?” In: *WhatIs.com* (July 2019). URL: https://whatis.techtarget.com/definition/data-labeling.

[24] “Convolutional Neural Network”. In: *Medium* (Feb. 2019). URL: https://towardsdatascience.com/covolutional-neural-network-cb0883dd6529.

[25] Gupta Dishashre. “Fundamentals of Deep Learning - Activation Functions and Their Use”. In: *Analytics Vidhya* (Feb. 2020). URL: https://www.analyticsvidhya.com/blog/2020/01/fundamentals-deep-learning-activation-functions-when-to-use-them/.