Real-time Static Gesture Detection Using Machine Learning

By

Sandipgiri Goswami

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science (MSc) in Computational Sciences

The Faculty of Graduate Studies
Laurentian University
Sudbury, Ontario, Canada

THESIS DEFENCE COMMITTEE/COMITÉ DE SOUTENANCE DE THÈSE
Laurentian Université/Université Laurentienne
Faculty of Graduate Studies/Faculté des études supérieures

Title of Thesis Titre de la thèse	Real time static gesture detection using machine learning	
Name of Candidate Nom du candidat	Goswami, Sandipgiri	
Degree Diplôme	Master of Science	
Department/Program Département/Programme	Computational Sciences	Date of Defence Date de la soutenance Aprill 24, 2019

APPROVED/APPROUVÉ

Thesis Examiners/Examinateurs de thèse:

Dr. Kalpdrum Passi
(Supervisor/Directeur de thèse)

Dr. Ratvinder Grewal
(Committee member/Membre du comité)

Dr. Meysar Zeinali
(Committee member/Membre du comité)

Dr. Pradeep Atray
(External Examiner/Examinateur externe)

Approved for the Faculty of Graduate Studies
Approuvé pour la Faculté des études supérieures
Dr. David Lesbarrères
Monsieur David Lesbarrères
Dean, Faculty of Graduate Studies
Doyen, Faculté des études supérieures

ACCESSIBILITY CLAUSE AND PERMISSION TO USE

Abstract

Sign gesture recognition is an important problem in human-computer interaction with significant societal influence. However, it is a very complicated task, since sign gestures are naturally deformable objects. Gesture recognition contains unsolved problems for the last two decades, such as low accuracy or low speed, and despite many proposed methods, no perfect result has been found to explain these unsolved problems. In this thesis, we suggest a machine learning approach to translating sign gesture language into text.

In this study, we have introduced a self-generated image data set for American Sign Language (ASL). This dataset is a collection of 36 characters containing alphabets A to Z and numeric digits 0 to 9. The proposed system can recognize static gestures. This system can learn and classify specific sign gesture of any person. We used a convolutional neural network (CNN) algorithm for the classification of the images to text. An accuracy of 99.00% was achieved on the alphabet gestures and 100% accuracy on digits.

Keywords: *Sign gestures, Image processing, Machine learning, Conventional neural network.*

Acknowledgments

First and foremost, I would like to express my deepest gratitude and appreciation to my supervisor, Dr. Passi, who gave me an opportunity, support, knowledge and dealt with my queries with prodigious patience throughout the study period. I believe no supervisor could be better than who God gifted to me.

I want to thank all my friends and my family members who made my research experience enjoyable. Thanks to all my classmates for their help and support on the testing phase. I could never have completed my master's research without the support of my family and my friends. I offer special thanks to my parents for support and patience and believing, and I am deeply grateful to my wife for everything.

Finally, and most significantly, I would like to acknowledge the persistent cooperation of my beloved family.

Table of Contents

Abstract…… iii
Acknowledgments…… iv
Table of contents…….. v
List of Figures…….. vii
List of Tables…… viii
List of Acronyms……. ix

1 Introduction
1.1 Sign Language…… 1
1.2 Mythologies and Misunderstandings about Sign Language……... 2
1.3 Objectives…… 3
1.4 Contributions…….. 3
1.5 An Overview of Methodology …….. 4
1.6 Outlines……. 5

2 Related Work
2.1 Recognition of American Sign Language…….. 6

3 Data set
3.1 American Sign Language…….. 11
3.2 Characteristics of American Sign Language…….. 12
3.3 Statistics about sign language use in Canada …… 13
3.4 Dataset and variables …… 13
3.5 Capturing Images for Dataset …… 15

4 Hand Gesture Detection
4.1 Introduction …… 16
4.2 Image Processing…… 17
4.3 Hand Detection Approaches ……. 19
4.3.1 Colour …… 19
4.3.2 Shape of hand……. 20
4.3.3 Learning detector from pixel values …….. 20
4.3.4 3D model-base detection …… 21
4.3.5 Motion …… 21
4.4 Approach for Hand Detection in this Research…….. 21
4.4.1 Skin Detection……. 23
4.4.2 Contour Comparisons……. 26

5 Machine Learning Approach
5.1 Artificial Intelligence ……. 27
5.2 Machine Learning …….. 28
5.2.1 Machine Learning Approaches…….. 28
5.2.2 Structure of Machine Learning Algorithm……. 29
5.2.3 Model Complexity……. 33

6 Neural Network
6.1 History…… 36
6.2 Structure of Neural Networks……. 38
6.2.1 Model of Neuron…… 39
6.2.2 Topology of the Network…….. 41
6.2.3 Cost Function…… 44
6.2.4 Optimization Procedure…… 44
6.3 Convolutional Neural Networks…… 46
6.3.1 Architecture…… 47
6.3.2 Training the CNN…… 50

6.4 Regularization of Neural Networks…… 53
7 Methods and Results
7.1 Software tools…… 55
7.2 Hardware and Software Configuration……. 57
7.3 Model Structure…… 59
7.3.1 About Keras Layers …… 59
7.4 Results ……... 61
8 Conclusion and Future work
8.1 Conclusion…… 70
8.2 Future work…….. 71
8.3 Limitations…… 72
References 73
Approval for conducting research involving human subjects…….. 78

List of Figures

Figure 1.1	Project overview of American sign language …….	4
Figure 2.1	A Data Glove design with Sensor ……..	9
Figure 2.2	A Glove device with Sensor ……...	9
Figure 3.1	American Sign language Manual Alphabet ……	11
Figure 3.2	American Sign language numbers ……..	12
Figure 3.3	Data set images ……	14
Figure 4.1	Diagram of the image processing pipeline……..	18
Figure 4.2	Hand posture detection steps ……..	22
Figure 4.3	HSV Colour Space …….	24
Figure 4.4	Image of detecting hand postures ……	26
Figure 5.1	Figure shows different levels of generalization of the model …….	34
Figure 5.2	Relationship between the model complexity and its ultimate accuracy is the relationship between training and testing error ……...	35
Figure 6.1	Diagram of the artificial neuron ……..	39
Figure 6.2	Activation Functions ……...	42
Figure 6.3	Fully connected Feed Forward Neural Network …….	43
Figure 6.4	Convolutional Neural Network (CNN) Architecture ……..	48
Figure 6.5	CNN network architecture for Numbers……..	49
Figure 6.6	Dropout: (a)Standard fully connected network. (b) Network with some neurons deactivated. (c) Activation of a neuron during the training phase. (d) Activation of a neuron during testing phase……	54
Figure 7.1	Epochs vs. validation accuracy for digits ……..	62
Figure 7.2	Epochs vs. validation accuracy for alphabets ……	63
Figure 7.3	Confusion matrix for 0 to 9 digits …….	63
Figure 7.4	Confusion matrix for A to Z alphabets ……..	64
Figure 7.5	ROC AUC graph for 0 to 9 digits ……..	68
Figure 7.6	ROC AUC graph for A to Z alphabets ……..	68
Figure 7.7	Shows real time testing with different users …….	69

List of Tables

Table 1.1	Sign Language in the Americas ……	1
Table 3.1	Statistics about Sign Language as a Mother Tongue ……	12
Table 3.2	Statistics about Knowledge of Sign Languages ……	12
Table 3.3	Dataset Description and Image property …….....	15
Table 7.1	Hardware Configuration …….....	58
Table 7.2	Software Configuration ……..	58
Table 7.3	Performance on Digits…….	65
Table 7.4	Shows the performance metrics for the .	67

List of Acronyms

- World Health Organization (WHO)
- American Sign Language (ASL)
- British Sign Language (BSL)
- Human computer interaction (HCI)
- Area of Interest (AOI).
- Hue, Saturation, Value (HSV)
- Red Blue and Green (RBG)
- Region of Interest (ROI)
- Support Vector Machines (SVM)
- K-Nearest Neighbours (KNN)
- Hidden Markov Model (HMM)
- Dynamic Time Warping (DTW)
- Dynamic Bayesian network (DBN)
- Artificial Neural networks (ANN)
- Microelectronic Mechanical System (MEMS)
- Artificial Intelligence (AI)
- Self-Organizing Maps (SOMs)
- Recurrent Neural Network (RNN).
- Graphics Processing Units (GPUs)
- Fully Connected Neural Network (FCNN)
- Application Programming Interface (API)

Chapter 1

Introduction

1.1 Sign Language

The World Health Organization (WHO) estimated that 250 million people in the world are deaf [1]. This group of people use symbolic language to communicate with other people. This symbolic language is called a sign language. Sign Language was developed for people with hearing-impairment so they could communicate with others. Sign language is not a unique language and there have been very improvements in the sign languages. There is proof that communication between hearing-impaired has been carried since the development of the sign languages [20]. Many counties have their own sign language, such as American Sign Language, French Sign Language, Indian Sign Language and Puerto Rican Sign Language, to name a few. Table 1.1 gives information about different sign languages used in the western continent. Gesture-based communication is dependent on region and has significant differences from other languages. It is essential to understand sign language when we communicate with hearing-impaired and their families. Lack of understanding results in significant challenges in understanding this community and may result in miscommunication.

Table 1.1: Sign Language in the Americas [20]

North America	Central America	South America
• American Sign Language •Inuit Sign Language • Quebec Sign Language •Puerto Rican Sign Language	•Costa Rican Sign Language •Guatemalan Sign Language •Honduras Sign Language •Mayan Sign Language •Mexican Sign Language •Nicaraguan Sign Language •Panamanian Sign Language •Salvadorian Sign Language •Tijuana Sign Language	•Argentine Sign Language •Bolivian Sign Language •Brazilian Sign Language •Chilean Sign Language •Colombian Sign Language •Ecuadorian Sign Language •Paraguayan Sign Language •Peruvian Sign Language

Sign Language is used to convey messages using hand movements, facial expressions and body language. It has been used by deaf people who cannot hear but can speak. Sometimes family members and relatives need to learn sign language to communicate with the hearing-impaired. By using a sign language a successful channel of communication can be established among the hearing-impaired community.

1.2 Mythologies and Misunderstandings about Sign Language

Many mythologies and misunderstandings have been interrelated with the concept of the Sign language. Normal people believe that Sign Language is simply a manual representation of the spoken word, which might not be true. There is somewhat a connection between a sign language and common parlance. Sign language has the difficulty of verbal communication, but it is self-determining from the alphabets. The best example is British Sign Language and American Sign Language, which are distinct languages, and the facts show that deaf people from the United States and Britain cannot communicate with each other using their distinct sign languages [72].

Another common misunderstanding about sign language is that it is globally understandable which is of course not true. As explained above, the Sign language that is used by the deaf in the United States is not the same as that in Britain. Different sign languages might be similar in some alphabets, but a deaf person from one country cannot communicate with a person from another country fluently [72].

Since sign language is a language with its distinct features, finger spelling or the use of guidebook alphabet cannot be used as an alternative to sign. It is utilized in marking the words with a non-existing sign or when the sign is not known. Also, a deaf person would take hours to convey a few minutes of messages through finger spelling.

1.3 Objectives

The main objective of this research is to help the deaf community to learn ASL alphabets and digits using a computer model. Students who are deaf or have a deaf parent or have a close relative who is deaf, can practice by themselves to learn a sign language alphabets and numbers. Real-time hand gesture detection is one of the most challenging problems in image processing and computer vision. The objective is to provide a fast and accurate detection of hand gestures in different lighting conditions and in cluttered background. Machine learning methods were used to detect gestures and translate them into text.

1.4 Contributions

The contributions in this thesis include the following:

- A dataset was created for the alphabets and numeric digits in the American Sign Language. An existing dataset could not be used in the experimentation as a large vocabulary database is required and laboratory data would be very restrictive in correct prediction.
- Convolutional Neural Network (CNN), a model of deep learning was developed to detect and interpret the sign language to text form.
- Skin detection and contour compression algorithms were used to detect hand gestures.
- A Gaussian filter was used to reduce noise in the images.
- Thresholding technique was used to get good results for higher computation speed when compared with other methods.
- Tests were conducted with non-controlled form, i.e. in different lighting conditions and in different backgrounds.

- A high accuracy of 100% was achieved to detect hand gestures for numeric digits and an accuracy of 99% was achieved to detect hand gestures for alphabets in American Sign Language.

1.5 An Overview of Methodology

In this thesis, Image classification and machine learning have been used for interpreting American Sign Language. For image classification, computer vision algorithms were used to capture images and to process data set for filtering as well as reducing noise from images. Finally, the data set is trained to recognize hand gestures using machine learning algorithm. A conventional neural network (CNN) for measuring the accuracy of training data sets. The results of the experimentation are given in Chapter 7. The general overview of the derived approach is combining the image classification and machine learning for American sign language recognition as shown in Figure 1.

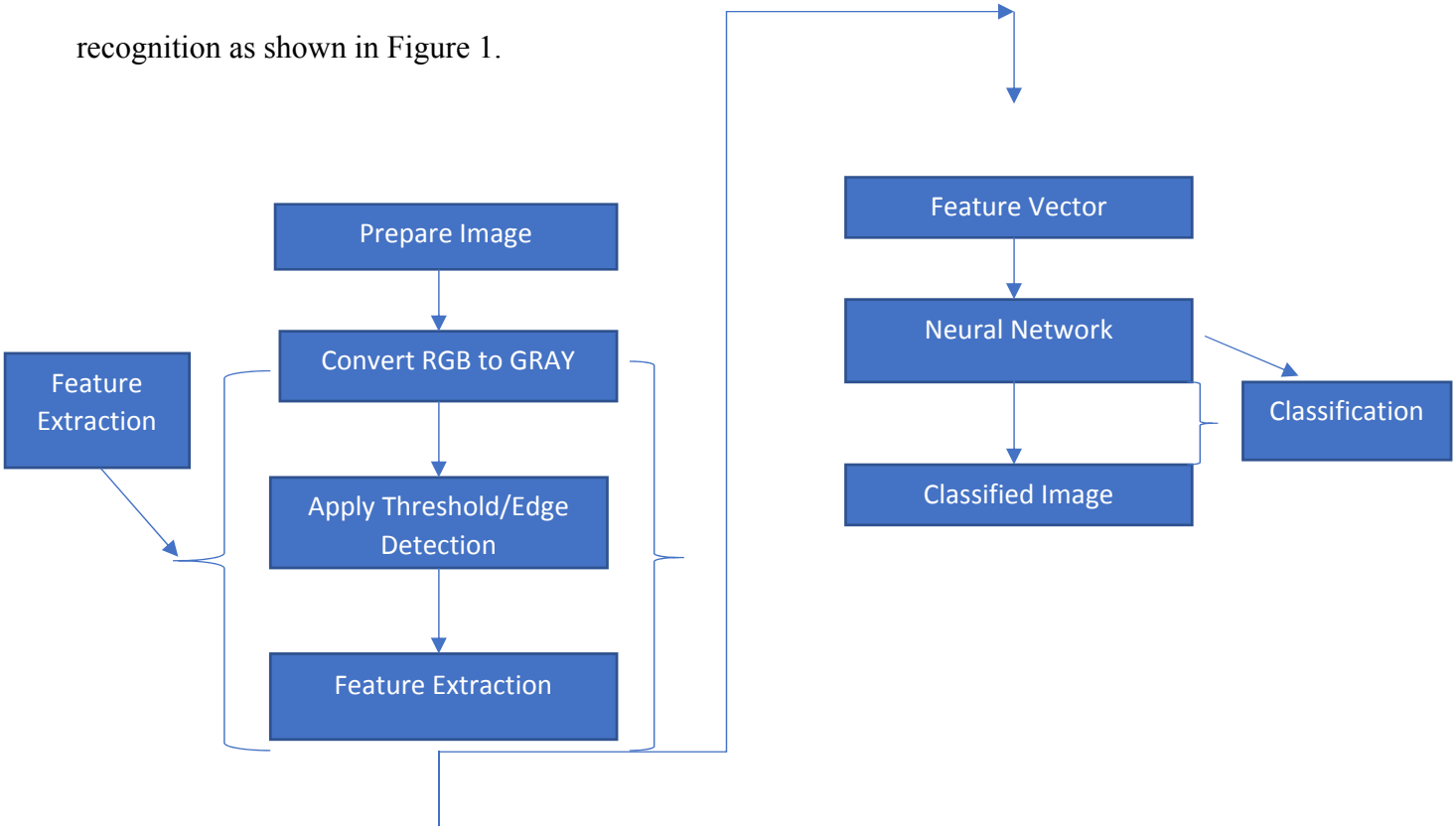

Figure 1.1: Project overview of American sign language

1.6 Outline

The thesis is organized as follows:

Chapter 2 discusses related work on different techniques and machine learning approaches used by other researchers.

Chapter 3 discusses the characteristics of the American Sign Language (ASL) and creation of the dataset for the ASL.

Chapter 4 discusses the hand gesture detection techniques and the proposed method using skin detection and contour comparison algorithm.

Chapter 5 discusses the machine learning approach used in the detection of hand gestures.

Chapter 6 describes Neural Network in general and Convolution Neural Network (CNN), a deep learning model for hand gesture detection.

Chapter 7 describes the software tools used in the experimentation, the configuration of the testing equipment, the implementation details and results.

Chapter 8 concludes the research work and discusses future work.

Chapter 2

Related Work

2.1 Recognition of American Sign Language

American Sign Language recognition using machine learning has been used by researchers in the past decades with different classifiers such as linear classifiers, neural networks and Bayesian networks [2-11].

As per researcher's point of view, a linear classifier is easy to work with because the linear classifiers are a relatively simple model, it requires sophisticated feature extraction and preprocessing methods to get good results [2, 3, 4]. Singha and Das [2] achieved an accuracy of 96% on Ten classes for images of gestures of one hand using Karhunen-Loeve Transforms. It translates and rotates the axes to build up a new framework based on the variance of the data. This technique is useful after using a skin colour detection, hand cropping and edge recognition on the images. They use a linear classifier to recognize number sign including a thumbs up, first and index finger pointing left and right, and numbers only. Sharma [4] has done research using Support Vector Machines (SVM) and k-Nearest Neighbors (KNN) to illustrate each colour channel after background noise deletion and noise subtraction [4]. Their research suggests using contours, which is very useful to represent hand contours. They achieved an accuracy of 62.3% using a Support Vector Machine on the segmented colour channel model.

Machine learning has been used for image recognition. Hidden Markov Model (HMM) and Dynamic Time Warping (DTW), two kinds of machine learning methods, are widely applied to achieve high accuracies [5, 6, 7]. These are mostly good at capturing time-based patterns, but they require characterized models that were defined before learning. Sterner and Pentland [5]

used a Hidden Markov Model and a 3-Dimensional glove that detects hand movement. Since the glove can attain 3-Dimensional detail from the hand regardless of spatial orientation, they achieved the best accuracy of 99.2% on the test set. Hidden Markov Model uses time series data to track hand actions and classify based on the position of the hand in new frames.

Suk [7] suggested a system for detecting hand gestures in a continuous video stream using a dynamic Bayesian network or DBN model. They try to classify moving hand gestures, such as creating a circle around the body or waving. They attain an accuracy of nearly 99%, but it is worth noting that all hand gestures are different from each other and are not American Sign Language. However, the motion-tracking feature would be applicable for classifying the dynamic letters of ASL: j and z.

Artificial Neural Networks (ANN) has been used to capture American Sign language transformation [8, 9, 10, 11]. Possibly, the essential advantage of artificial neural networks is that they represent the essential classification structures. However, ANN requires significantly more time and data to train. Up to the present time**,** most have been comparatively low. Mekala [8] classified video of the ASL alphabet into text using unconventional feature abstraction and a three-layer Neural Network. They extracted features using hand situation and movement. In the past, American Sign Language classification could recognize the presence and position of 6 “points of interest” in hand, each finger and the center of the palm. Mekala also used Fourier Transforms of the images to classify what section of the frame the hand was positioned at. Whereas they claim to correctly categorize 100% of images with this framework, there is no indication of whether this result was reached in the training, validation or test set.

Admasu and Raimond [9] classified Ethiopian Sign Language and achieved an 88.5% accuracy result using a feed Forward Neural Network. They used a substantial amount of image preprocessing, including image size standardization, image background deduction, contrast adjustment, and image segmentation**.** Gabor Filter and Principal Component Analysis method

was used to extract features. The most related work up to date is by L. Pigou's [11] research of ANN's to categorize 20 Italian gestures from the "ChaLearn 2014 Looking at People" gesture recognizing competition. They used a Microsoft Kinect on whole body images of people performing the gestures and reached a cross-validation accuracy of 91.7%. With the 3-Dimensional glove, the Kinect allows detection of depth features, which helps significantly in classifying American sign language.

Non-Vision based technology such as Glove-based hand shape recognition usually contains the person wearing gloves and a certain quantity of wires to connect this glove to a computer. These methods are a tough and non-natural way to communicate with the computer [15]. This device requires electricity or electromagnetic interference to get data about the hand, which is sufficient to describe a handshape gesture [16]. Scientists refer to data gloves in different ways, e.g. CyberGlove and Accele Glove.

Figure 2.1 shows the position of the sensors in a data glove proposed by Bedregal [17]. The timeline of frames can characterize any movement. Thus, a timeline of hand arrangement represents a hand movement using a data glove. An arbitrarily generated hand configuration was used to replicate the data transfer [17]. Each expression of the handshape is represented by a tuple of interval angles from each sensor. The detection was applied to Brazilian Sign Language (LIBRAS), using Fuzzy logic.

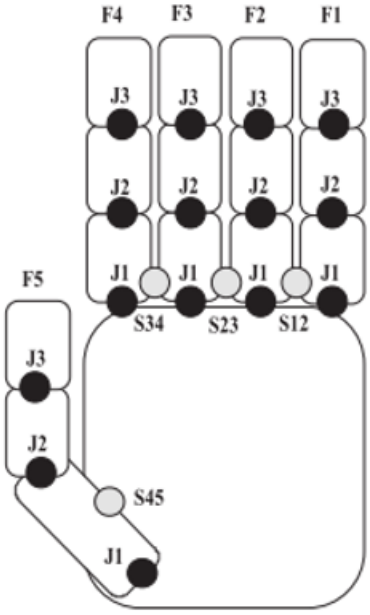

Figure: 2.1. A Data Glove design with Sensor.[17]

In this paper, they developed a similar hardware device called the Accele Glove. In their research, they used a microelectronic mechanical system (MEMS) to extract hand configuration. They have been functional in Vietnamese Sign Language for twenty-three gestures with Fuzzy logic. They achieved the results by handshape, with an overall 98% precision. The relative angles between the palm and finger are the data found from the sensing device. The glove covers six accelerometers and a BASIC Stamp microcontroller as in Figure 2.2 [17].

Figure 2.2 A Glove device with Sensor. [17]

Ashok and Gauri [18] have explained the state of the art about sign language recognition. They have also included major sign languages of the world and surveyed a number of papers published on sign language recognition for different countries. In this research paper, they provided details about the availability of a standard database and creating their own database. Standard dataset for static or dynamic does not exist for all countries [18].

Nguyen HD and Phi LT [19] have proposed a new system for a gesture-to-speech/text for the deaf community, applied to Vietnamese Sign Language. They have created a smart glove. They attached 10 flex sensors and 1 accelerometer on that glove. Flex sensors were used for detecting the curves of fingers, and the accelerometer was used in spotting a movement of a hand. The main advantage of this glove is that it does not depend on light conditions, which means it gives better accuracy in dark environments. As per the author, the glove is low price, low power consumption and has full mobility. Another advantage of these gloves is that they attached flex sensors which used a wireless interface to a microcontroller.

Chapter 3

Data Set

In this chapter, we first describe the American Sign Language and then describe the acquisition of the dataset.

3.1 American Sign Language

American Sign Language (Figure 3.1, 3.2) [13] is used to communicate between the deaf community. However, there are only 2.5 million ~ 5.0 million who use sign language which significantly limits the number of people they can easily communicate with [12].

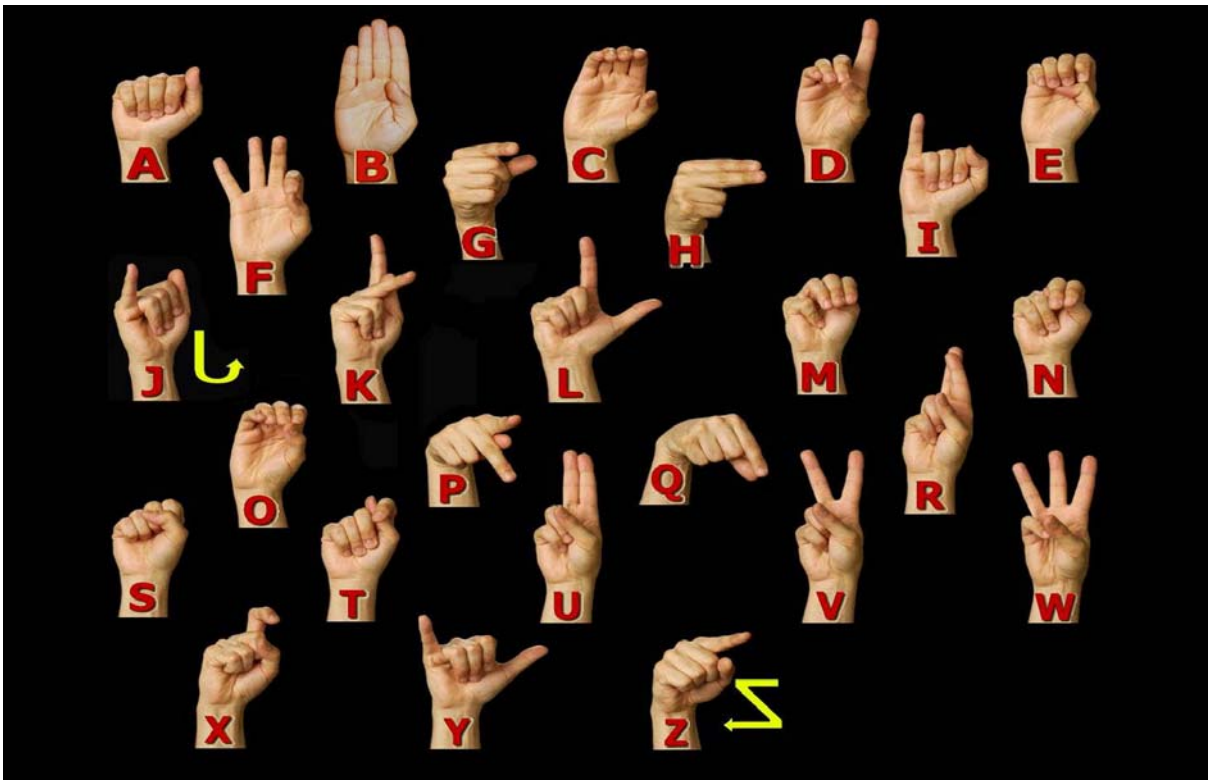

Figure 3.1 American Sign language Manual Alphabet [13].

American Sign Language was adapted from French Sign Language which was introduced by Thomas Hopkins Gallaudet in the United States [12]. ASL is like French Sign Language; individuals who know American Sign Language can effectively communicate in French Sign

Language [49]. As there are variations between English spoken in England, United States and Australia, there are differences in their sign languages [12].

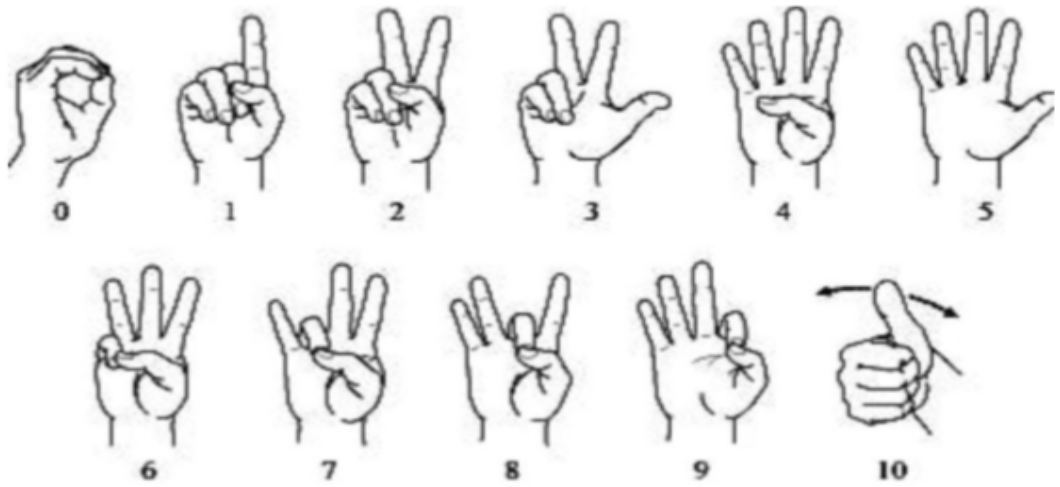

Figure 3.2. American Sign language numbers. [13]

3.2 Characteristics of American Sign Language

- American Sign language is an entire visual-gestural dialect with its very own language structure, vocabulary, and linguistic structure.
- Like other sign languages, it utilizes the hands, the body, and the face looks (counting mouth developments) to express the significance and the eyes to see the meaning.
- The hand-to-hand connection is especially critical in ASL since it has no composed frame. There are in any case, documentation frameworks that were utilized for recording signs on paper.
- ASL is separate from English and is unique from other sign languages. An example of the distinctiveness of sign languages from each other and the surrounding spoken language(s) is that, although English is the shared spoken language of the U.S., Canada, and Britain, signers of ASL do not understand signers of British Sign Language (BSL).

3.3 Statistics about sign language use in Canada

In Canada, Statistics Canada reports that, as indicated by the 2006 Census, 8,995 people revealed a gesture-based communication just like their primary language or one of their first languages, as given in Table 3.1.

Table 3.1: Statistics about Sign Language as a Mother Tongue [14].

American Sign Language	2,485
Quebec Sign Language	730
Sign languages, not included elsewhere	5,780

In addition, Statistics Canada reports that as per the 2006 Census, 43,090 people reported knowledge of a gesture-based communication, as given in Table 3.2.

Table 3.2: Statistics about Knowledge of Sign Languages [14].

American Sign Language	11,110
Quebec Sign Language	730
Sign languages, not included elsewhere	5,780

3.4 Dataset and variables

A dataset for American Sign Language was created as there is no standard dataset available for all countries/sub continents [18]. In the present scenario, the requirement for large vocabulary dataset is in demand [18]. Moreover, the existing dataset is incomplete and additional hand gestures had to be included. In future, this research will help other researchers to develop their own dataset based on their requirements. This dataset is a collection of 36 characters which contains A to Z alphabets and 0 to 9 numerical digits. Right hand was used to capture 1000 images for specific alphabets and numbers. Code was implemented to flip the images from the

right to the left-hand image. The height and width ratios vary significantly but average approximately 50x50 pixels. The dataset contains over 72,000 images in grayscale. Additionally, people can add their images to this dataset. Figure 3.3 shows images of A to Z alphabet. Table 3.3 describes the dataset properties.

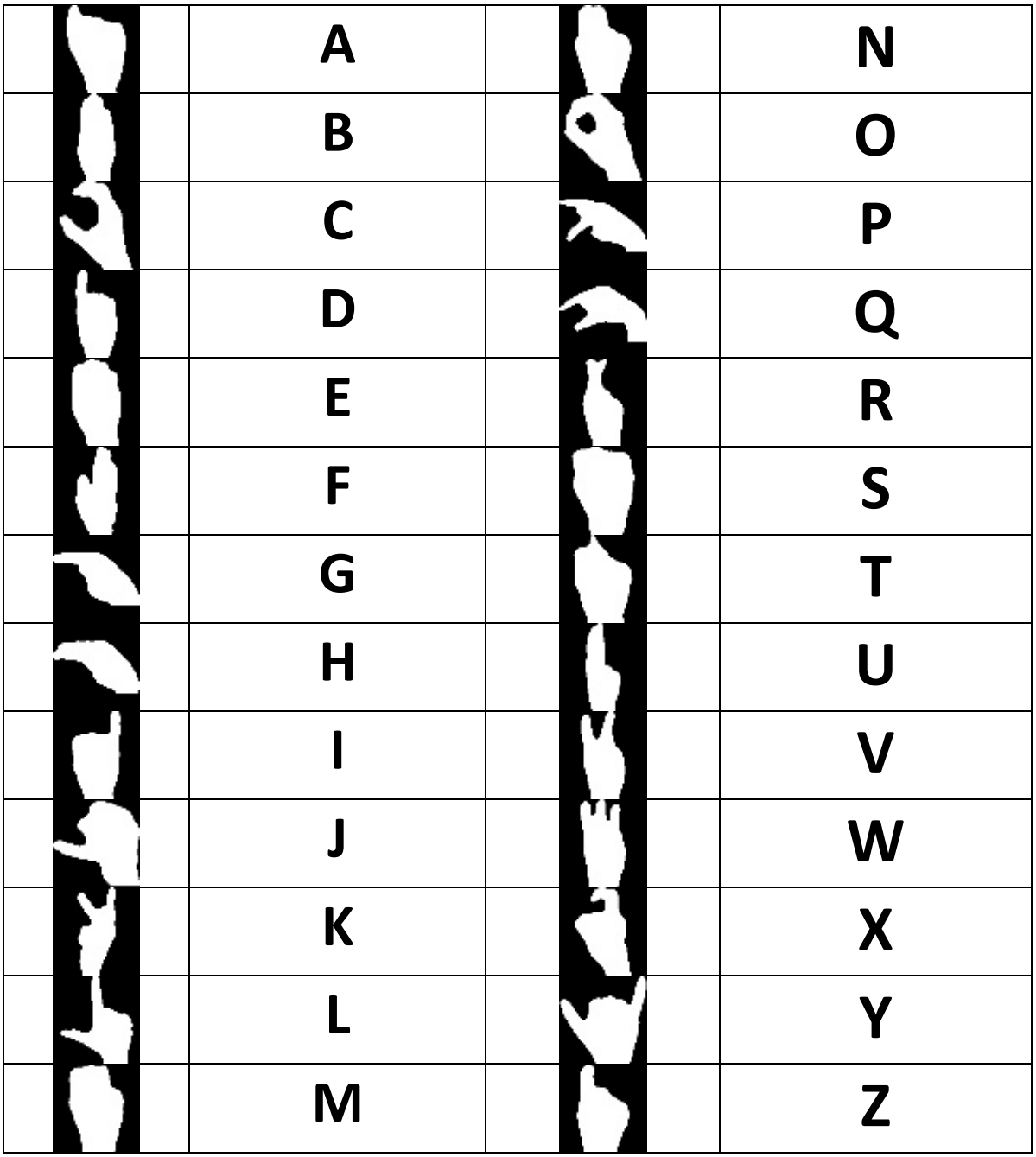

Figure 3.3: Data set images.

Table 3.3: Dataset Description and Image property

Property	Description
Alphabets	A to Z
Numbers	0 to 9
Color	Gray Scale
Dimensions	50x50
Height	50 pixels
Width	50 pixels
File type	JPEG

3.5 Capturing Images for Dataset

To detect hand gesture using skin colour, there are different approaches including skin color-based methods. In this thesis, after detecting and subtracting the face and other background, skin recognition and a contour comparison algorithm were used to search for the hand and discard other background colour objects for every frame captured from a webcam or video file. First, we need to extract and store the hand contour and skin color which is used to compare with the frame in order to detect the hand. After detecting the skin area for each frame captured, the contours of the detected areas were compared with the saved hand histogram template contours to remove other skin like objects existing in the image. If the contour comparison of the spotted skin area complies with any one of the saved hand histogram contours, then it captured hand gesture only.

Chapter 4
Hand Gesture Detection

Generating our own dataset is important in hand gesture detection. In this chapter, we explain different hand detection approaches using skin detection and contours comparison algorithms.

4.1 Introduction

Dataset creation plays an important role in the object detection. To analyze and extract relevant data about an object of interest from an image, one needs first to detect the object in the image. Hand posture detection refers to finding the place and size of hand within a sequence of images. Nowadays it is a very popular computer vision problem and has numerous applications, such as gesture recognition, sign language recognition, computer graphics games, and human-computer interaction (HCI).

Skin colour [21, 22, 23] is an important property to detect hand and tracking. However, Colour image has a different problem of removing other objects with similar colours such as a face and a human arm. To solve this problem, we introduce a new method to detect hand postures only using face detection and subtraction, skin detection, and hand postures contour detection and comparison algorithm [24]. The face was removed because the skin detection will detect the face and the face's contours are very similar to the first-hand gesture contours. The area of interest n captured, and another same skin colour is removed from an area of interest. After removing an unwanted area of the face, the skin area was detected using the hue, saturation, value (HSV) colour model since it has real-time performance and it is strong against alternations, scaling and lighting conditions. Then, the interesting area of contours was compared with all the existing

hand posture template contours to eliminate the unwanted area of interest, like objects existing in the image.

4.2 Image Processing

Computer vision is an important topic in image processing. Investigation on the subject has been to approach the problem from bottom-up perception for many years. This was the before-mentioned effort to validate directions guiding the vision of living organisms. This method was actually very successful in certain environments.

A general explanation of computer vision in image processing can be briefly explained in the following steps:

- Image capture - Image is captured using a camera or similar device and digitized.
- Pre-processing – A captured image is modified to highlight important features such as noise reduction, contrast, normalization etc.
- Segmentation detection - Selection of the region of interest like edges, similar surfaces.
- Description – Feature extraction of radiometric, photometric descriptors and so on.
- Classification - Some means to categorize the object.

The steps of image processing are shown in Figure 4.1

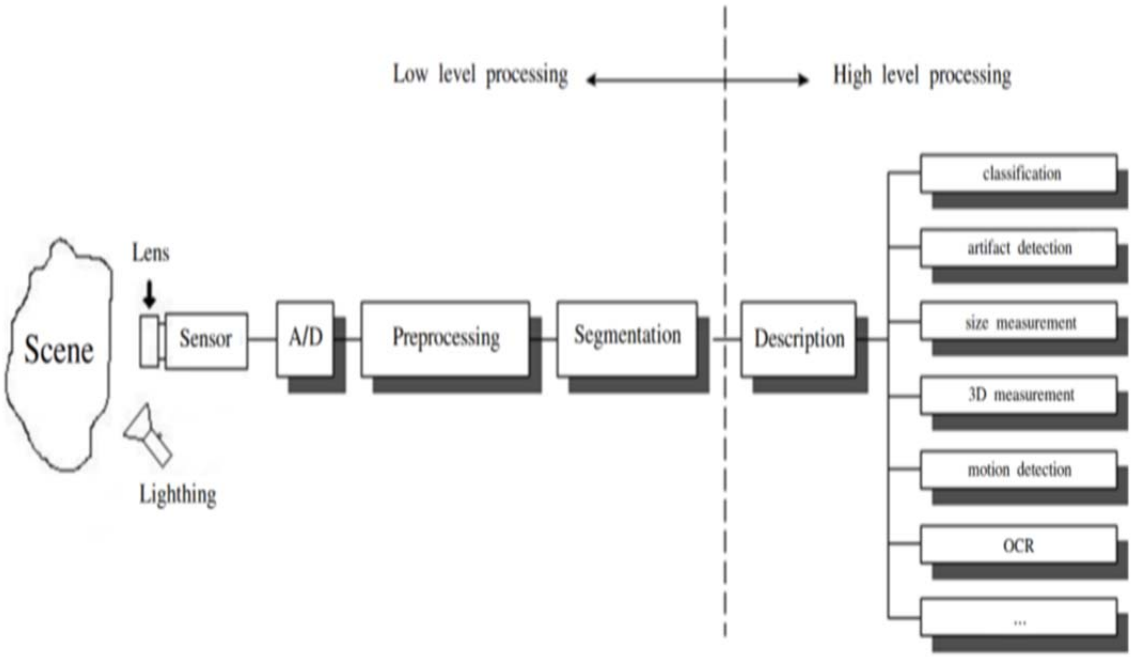

Figure 4.1: Diagram of the image processing pipeline. [55]

Even though machine learning has been a field of study since the second half of the 20th century, there was no wider implementation of its techniques in image processing for a very long time. It was first announced in the classification step of the processing pipeline. In other words, complex difficulties were simplified by reducing the visual information contained within the image into a handful of simple features that were nourished into a machine learning model.

This carries the problem that these applications are not very useful. Each application is usually only capable of resolving the very narrow problem, and any deviation from ideal circumstances can mean failure. An application can have complications with varying contrast, brightness, scaling, rotation etc.

The second problem is often the fact that because the image must be pre-processed numerous times before it is fed into the machine learning model, it requires additional time and computational resources. This is less of a problem with existing hardware innovations, but it is

still not an unimportant factor and it can have a negative effect on the cost of the result. This is where machine learning in general indicates a noteworthy advantage.

A conventional approach to computer vision can find success in applications that are deployed in very limited environments with rigid constraints. In a controlled location, it is usually very simple to define the problem in formal guidelines. Even though it can be still available in certain places, it starts to be forced out by the application of machine learning simply because of the barrier of entry for extensive adoption reductions every day.

4.3 Hand Detection Approaches

There are different approaches for hand detection. They have been introduced in the literature to employs different visual features and in many cases their combination. These are different approaches such as motion, skin colour, shape, and 3D models of hands. Hand detection methods were discussed in [25] and will be discussed later in this chapter.

4.3.1 Colour

Skin colour detection has been used in many hand gesture recognition projects. The main objective of giving a model of skin colour is the choice of the colour space to be utilized. Different colour spaces have been introduced such as RGB, normalized RGB, HSV, YCrCb, and YUV. Colour spaces that effectively divide the chromaticity from the luminance parts of colour are typically regarded as preferable as I did in my approach by removing the Value (V) section in the HSV model. This is because of employing chromaticity-dependent mechanisms of colour only.

Generally, skin colour detection can be disordered by background objects; they have skin colour distribution like human skin colour. Some projects have been done on this problem by using background subtraction [26, 27]. On the contrary, it was expected that unwanted background

subtraction normally depends on the camera system that does not move with respect to a fixed background. Another solution [28, 29] has been applied with dynamic modification of background compensation techniques.

4.3.2 Shape of Hand

Shape property is defined in terms of the set of contours that describe the boundary of a hand. More details can be acquired by reducing the contours of objects in the frame. If the contour is perfectly detected, it provides a good presentation of the hand gesture which is indirectly related to a viewpoint, skin colour, and lighting. Typically, contour extraction based on edge recognition uses many edges fitting to the hand image area but also to distinct background objects.

Accordingly, sophisticated post-processing techniques are needed to develop the presentation of this method such as our approach in [24] by combining skin colour detection with contour detection and comparison after face subtraction. A second method that has been used in fingertip finding is pattern matching. Patterns can be images of fingertips [30] or fingers [31] or generic 3D cylindrical models [32]. These pattern matching methods can be upgraded by using extra image features such as contours [26].

4.3.3 Learning Detectors from Pixel Values

The present technique uses a machine learning method, which has shown remarkably strong results in face recognition and good results in hand recognition [33]. In [34], an object recognition method was proposed in which a weak classifier may be a simple finder that uses basic image block differences efficiently calculated using an integral image. On the other hand, this technique may provide an unnecessary number of weak classifiers. The AdaBoost technique has a drawback because it does not consider the removal of chosen weak classifiers that no longer take part in the recognition procedure. The strong classifier is composed of a set of weak

learners with associated weights. Each weak learner uses a single image feature to produce a hypothesis. The object detection system uses simple features to build weak learners. Viola and Jones suggest the use of features rather than pixel values directly. Also, there is some problem to distinguish the hand using the Viola-Jones method [34, 35] related to rotation and cluttered background.

4.3.4 3D Model-Based Detection

One major advantage of a 3D model-Based method is that it can allow for view-independent detection. The used 3D models must have enough degrees of freedom to adapt to the dimensions of the hand that exist in an image. Different mock-ups use different image features to build feature-model communications. Line and point features are applied in kinematic hand models for recovering angles created at the links of the hand [36, 37, 38]. The hand gesture is then estimated based on the relations between the 3D model and the observed image features.

4.3.5 Motion

Motion is a dynamic feature employed by some procedures for hand detection. Motion-based recognition requires a highly controlled arrangement, and it adopts that the only movement in the image resulted from hand motion. In the case of fixed cameras, the issue of movement assessment is solved with background maintenance and successive removal. This explanation is employed in [50,51] for recognizing the hand from other skin- coloured objects, and for testing with lighting conditions resulting from coloured lights. The difference in pixel strength between two consecutive frames is close to zero for the background pixels. Motion objects are placed by selecting and maintaining a suitable threshold.

4.4 Approach for Hand Detection in this Research

An integrated system is proposed for detection, segmentation, and tracking of the hand in a gesture recognition system using a single webcam. Some other methods use colour gloves [39, 40], whereas our method can detect the plain hand posture by integrating two useful features: skin colour detection and contour matching. My proposed hand posture finding algorithm has real-time performance and is strong against rotations, scaling, cluttered background, and lighting conditions. Section 4.4.2 shows the strength of the proposed hand posture detection algorithm based on comparison with other methods. Detecting the human hand in a plain background will boost the performance of hand gesture recognition systems. In this method, the speed and result of recognition will be the same for any frame size taken from a webcam such as 640×480, 320×240 or 160×120 and the system will also be robust against a plain background as only the hand posture area is detected. A smaller image size that holds the detected hand posture area is suitable for training image size of a training stage of the classification.

To detect the hand gesture in the image, a four-phase system was designed as shown in Figure 4.2. First, load hand contour template which will be used to compare and detect hand skin area pattern from webcam using the contours comparison algorithm. Then we will open a camera which has a square box to capture hand gesture. The hand must be placed fully within the square box. The skin colour locus (captured skin contour template) for the image is removed from the user's skin colour after face deletion. In the last step, the hand gesture is spotted by removing false positive skin pixels and identifying hand gesture and other real skin colour regions using contour matching with the loaded hand gesture pattern contours.

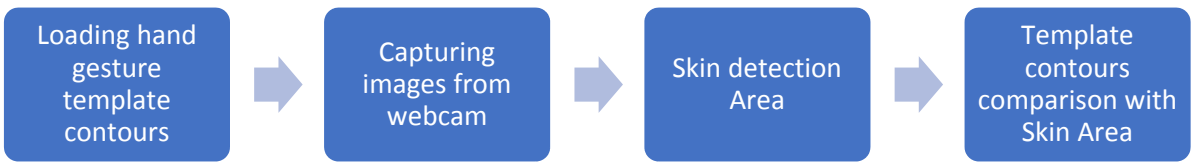

Figure 4.2 Hand posture detection steps

4.4.1 Skin Detection

Skin detection is a useful approach for many computer vision applications such as face recognition, tracking and facial expression, abstraction, or hand tracking and gesture recognition. There are recognized procedures for skin colour modelling and recognition that will allow differentiating between the skin and non-skin pixels based on their colour**.** To get a suitable distinction between skin and non-skin areas, a colour transformation is needed to separate luminance from chrominance [42].

The input images normally are in Colour format (RBG), which has the drawback of having components dependent on the lighting situations. The misunderstanding between the skin and non-skin pixels can be decreased using colour space transformation**.** There are different approaches to detection of skin colour components in other colour spaces, such as HSV, YCbCr, TSL or YIQ to provide better results in parameter recovery under changes in lighting conditions. Researches have shown that skin colours of individuals cluster tightly in the colour space for all people from different societies, for example, colour appearances in human faces and hands vary more in intensity than in chrominance [41, 43]. Thus, take away the intensity V of the original colour space and working in the chromatic colour space (H, S) provides invariance against illumination situations. In [42], it was shown that removal of the Value (V) component and only using the Hue and Saturation components, can still permit detection of 96.83% of the skin pixels. In this research, hue, saturation, value (HSV) colour model was used, since it has been shown to be one of the most adapted to skin-colour detection [44]. HSV model is easier to represent a color than RGB color space. It is also well-matched with human colour perception.

Also, it has real-time execution and it is more robust in cases of rotations, scaling, cluttered background, and changes in lighting condition. The projected hand gesture detection algorithm is real-time and sturdy against the mentioned previous changes. The other skin like objects existing in the image are removed from contour comparison with the loaded hand posture prototype contours**.**

The HSV colour space is gained by a nonlinear transformation of the essential RGB colour space. The conversion between RGB and HSV was described in [45]. Hue (H) is a section that characterizes a pure colour such as pure yellow, orange or red, whereas saturation (S) provides a measure of the degree to which a pure colour diluted by white light [46]. Value (V) attempts to represent brightness along the gray axis such as white to black, but since intensity is subjective, it is thus difficult to measure [46].

According to [47] and Figure 4.3, Hue is estimated in HSV colour space by a position with Red starting at 0, Green at 120 and Blue at 240 degrees. The black mark in the diagram at the lower left on the screen determines the hue angle.

Saturation is a ratio that ranges between 0.0 along the middle line of the cone (the V axis) to 1 on the edge of the cone. Value ranges, string from 0.0 (dark) to 1.0 (bright).

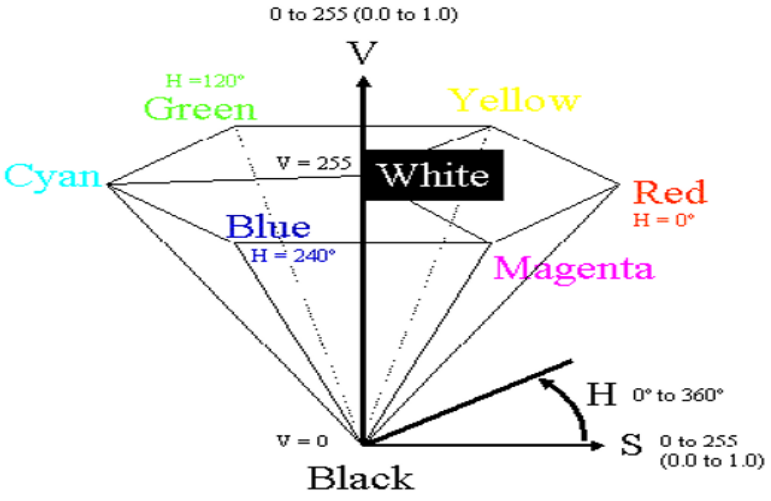

Figure 4.3 HSV Colour Space [47]

According to [41], the HSV model can be resulting from the non-linear transformation from an RGB model according to the calculations in equations 4.1 – 4.4.

$$H = \begin{cases} \theta, G \geq B \\ 2\pi - \theta, G < B \end{cases} \quad (4.1)$$

$$S = \frac{\max(R,G,B) - \min(R,G,B)}{\max(R,G,B)} \quad (4.2)$$

$$V = \frac{\max(R,G,B)}{255} \quad (4.3)$$

$$\theta = \arccos\left\{\frac{[(R-G)+(R-B)]/2}{[(R-G)^2 + (R-G)(G-B)]^{1/2}}\right\} \quad (4.4)$$

Once image converts to HSV then we use a histogram of an image containing our object of interest. After that, Gaussian kernel is applied to remove noise from Image. As per a classification point of view, skin-colour detection is divided into two class problem: skin-pixel vs non-skin-pixel classification. Currently, different known classification approaches exist such as thresholding, Gaussian classifier, and multilayer perceptron [48, 52, 53].

In this research, thresholding technique is used that allows getting a good result for higher computation speed when compared with other techniques, given our real-time requirements. In image processing, Gaussian blur is used to reduce noise and reduce details. Thresholding classification is used to find the values between two components H and S in the HSV model as the Value (V) component was removed. Usually, a pixel can be observed as being a skin-pixel when the following threshold values are satisfied: $0° < H < 20°$ and $75° < S < 190°$. In the last step, skin detected object is stored into pickle object, called contour template. Contour template

is a curve joining all continuous points having the same intensity. As we stated that contours are the boundary of a shape with the same intensity. It stores the values of x and y coordinates of boundary of the shape, to store all coordinate values.

4.4.2 Contour Comparisons

Once the contour template has been detected, the contours of the detected skin colour are recovered and then compared with the contours of the hand gesture patterns. Once skin colour contours are recognized as belonging to the hand gesture contour patterns, that area will be identified as a region of interest (ROI) which will then be used for tracking the hand movements and saving the hand posture in JPEG format in small images as shown in Figure 4.4. After that, stored images are further used to extract the features needed to recognize the hand postures in the testing stage.

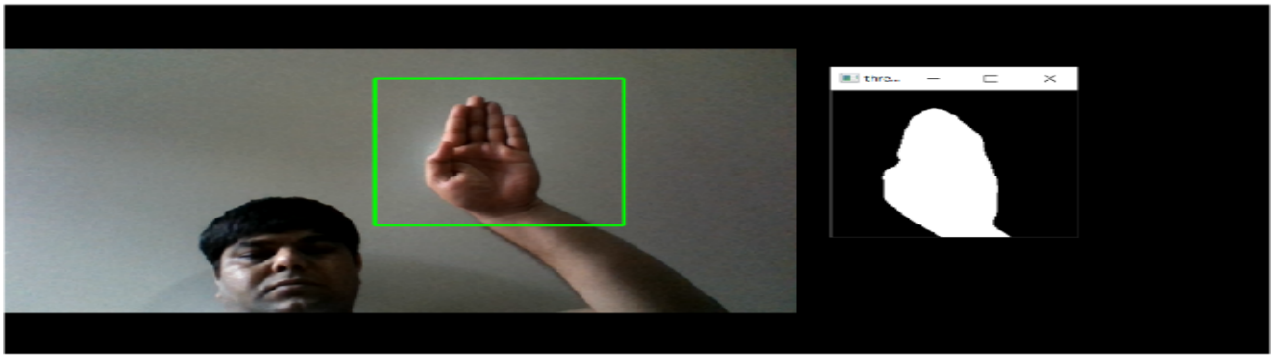

Figure 4.4: Image of detecting hand postures.

If there are two hand gestures in the image, the system will substitute in detecting one of the two hands for every frame captured as the Open Computer vision function cvBoundingRect will circle one rectangle only around the detected hand, which has the main matching contours with the overloaded hand posture templates contours. The single frame will circle the detected hand posture for one frame and may enclose the other hand posture for the next frame if it has a higher matching contour.

Chapter 5

Machine Learning Approach

Supervised machine learning approach was used in this research because each image has an assigned label.

5.1 Artificial Intelligence

Research in Artificial Intelligence (AI) is very general and spans across numerous different areas such as mathematics, computer science, philosophy, economics and even ethics. This field is very wide and can be attempted by many different viewpoints. Therefore, this explanation will not be very exhaustive. For comprehensive and more complex description into the subject, refer to [54].

One of the many possible definitions of AI can be briefly explained as a search method to develop an artificially intelligent agent. In other words, it is an effort to create intelligent machines that are either intelligent or can be perceived as intelligent ones. One of the most important skills of intelligent agents is a sense of vision. A sense of vision is usually required for a positive degree and not always, it is necessary that it rivals the abilities of the human visual apparatus.

First, it tries to resolve the vision problems which were tackled from the so-called bottom-up approach in which the system was instructed with a hard-coded set of protocols describing the vision. It was expected that as the understanding of an instrument allowing humans to abstract

information from the visual scene, the hard-coded systems could be fed this understanding and thus more skilled systems can be created. The problem with this method was that it highly underestimated the difficulty of reinforcement of these protocols. Therefore, it mainly failed to devise such a system.

This main idea the investigators postulated was that in order to resolve the problem of deploying vision capabilities for the artificially intelligent system, it is compulsory to introduce a procedure that would allow AI systems to extract patterns from provided data. It is an overview of systems that can learn. A process that enables systems to learn is usually called machine learning.

Machine learning is again a relatively extensive term that can be used in different frameworks. In this work, it is meant to be understood as a technique that is used to create mathematical representations for image detection. There are numerous types of machine learning models that are useful for different tasks. The most common task attempted is called the task of classification, in which it classifies the occurrence of input into a correct discreet and mainly predetermined class. One most common type of machine learning task is called regression, which is based on the input data trying to estimate unknown continuous valued quantity.

5.2 Machine Learning

Machine Learning is a process that is used to create models that can abstract information from data to resolve the given problem and consequently repeatedly improves their performance.

The interesting viewpoint that can be used to view machine learning as encoding information using fewer bits than the original representation, where the machine learning model is trying to get more details from input to evaluate model summary.

5.2.1 Machine Learning Approaches

There are mainly two types of machine learning approaches:

- Unsupervised Learning
- Supervised Learning

Unsupervised Learning

In the unsupervised learning method, the model is trained by detecting new data and find patterns in the data without being instructed on what they are. Contrasting to supervised learning, the benefit of this method is that the model can learn from data without supervision. This means that there is no need for input data to be labeled. Therefore, it takes less time and resources to deploy these models in practice.

The biggest difficulty of the supervised learning method in real-world applications is to obtain appropriate data. Appropriate data in this context means, data that were somewhat classified into different categories, which may not exist in all situations.

The mainstream of unsupervised learning procedures belongs to a group called clustering algorithms. These algorithms are centred on the idea to analyze ordered clustering of data in the input space to determine their relationship. This is achieved by the belief that data point clustering in input space is likely to exhibit similar properties.

Illustrations of unsupervised learning models are:

- K-means -clustering model
- Self Organizing Maps (SOMs)
- Principal Component Analysis (PCA) - dimensionality reduction

Image classification usually does not depend on the use of unsupervised learning methods.

Supervised Learning

Supervised learning was used in this research as the dataset images have assigned labels. The supervised learning method is more commonly used when data is known. This method needs training data with a specific format. Each instance must have assigned label. These labels make available supervision for the learning algorithm. The training process of supervised learning is constructed on the following principle. First, the training data are fed into the model to produce estimates of output. This estimate is compared to the assigned label of the training data in order to evaluate the model error. Based on this error the learning algorithm alters model's parameters in order to reduce it.

5.2.2 Structure of Machine Learning Algorithm

Although there are numerous machine learning algorithms, all have a common structure that can be generalized. Structure of nearly all machine learning algorithms can be defined as composed of the following components:

• Dataset description

• Model

• Cost function

• Optimization technique

Almost all supervised learning algorithms use the same Dataset description. The other three components can vary intensely. This level of analysis is suitable for developing the intuition for Neural Networks (NNs) and a description of its individual components.

Dataset description

Supervised learning requires data sets with detailed properties. Each dataset holds a set of n instances which contain a pair of the input vector x_i and output scalar y_i. Input vector

$$x_i^T = [x_1, x_2, \dots, x_p], \tag{5.1}$$

Where i is an index of an instance, p, is the length of the input vector

Specific components of the input vector must be of a uniform type. In the case of the image as input data, it is the value of individual pixels (e.g. 0-255). In other cases, they could be real values. As a general rule, input data should be normalized. This holds in images automatically since each pixel must have its values in a fixed range. It is very important in other types data, where this is not guaranteed.

Output scalar y_i characterizes a class of the given instance. The type of this output value thus must obtain only certain values. To put it differently, it must be a set of cardinalities equal to the number of all possible classes.

Model

The model is predicted by taking input x_i to predict values of its output y_i. Each model has parameters represented by vector θ, which are used during the training process. The simplest example of the model type is a linear model, also called linear regression. Parameters θ of this model are

$$\theta^T = [\theta_1, \theta_2, \dots, \theta_p], \tag{5.2}$$

where p is the number of parameters equal to the size of the Input vector x_i.

Prediction $\hat{y}_i$ of the model on instance i is computed as

$$\hat{y}_i = \sum_{j=1}^{p} x_{ij}\, \theta_j \tag{5.3}$$

Estimate of the model on the entire dataset in matrix notation is given by:

$$\hat{y} = X\theta. \tag{5.4}$$

Estimates in expanded notation are given as:

$$\begin{bmatrix} \widehat{y_1} \\ \vdots \\ \widehat{y_n} \end{bmatrix} = \begin{bmatrix} X_{11} & \cdots & X_{1p} \\ \vdots & \ddots & \vdots \\ X_{1p} & \cdots & X_{np} \end{bmatrix} \begin{bmatrix} \emptyset_1 \\ \vdots \\ \emptyset_n \end{bmatrix} \tag{5.5}$$

Cost Function

To achieve the learning accuracy of the machine learning algorithm, it is necessary to approximate the error of its predictions. This is assessed with so-called cost function and called loss function.

This function must have specific properties. The ability of the machine learning algorithm to learn rests on the approximation of its improvement with the change of its parameters. Therefore, cost function must be at least partially differentiable. In the case of linear regression, it is most common to use sum of square error. The main aim is that derivative of this function for a linear model has only one global minimum.

The cost function is defined as

$$J(\theta) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}(y_i - x_i^T\theta)^2 \,. \tag{5.6}$$

For the optimization determinations, it is usually useful to express the cost function in matrix notation

$$J(\theta) = (y - X\theta)^T\,(y - X\theta). \tag{5.7}$$

Optimization technique

The last part of the learning algorithm is the optimization technique. It consists of an update of the model's parameters θ in order to increase prediction accuracy. In other words, to find θ such that the value of cost function $J\,(\theta)$ for given dataset is as small as possible.

To examine the change of the cost function on given dataset, it is necessary to compute the derivative of $J(\theta)$ with respect to θ

$$\frac{\partial J(\theta)}{\partial \theta} = \frac{\partial}{\partial \theta}[(y - X\theta)^T\ (y - X\theta)]$$

$$= \frac{\partial}{\partial \theta}[y^T y + \theta^T X^T X\theta - 2y^T X\theta]$$

$$= 2X^T X\theta - 2X^T y. \tag{5.8}$$

For the linear model, it is possible to find an optimal result which is a global minimum of the cost function. The optimal result

$$\theta = (X^T X)^{-1} X^T y\,, \tag{5.9}$$

is found by comparing the partial derivative of $J(\theta)$ to 0. The only condition is that $X^T X$ must be non-singular.

Unfortunately, only very simple problems can be handled using the model as simple as linear regression. The more complex model usually means more complex cost function. The optimization process of more complex cost functions cannot assure to find the global minimum. In this case, the optimization technique must have an iterative character. To put it in a dissimilar way, the algorithm must be a method with minimum iterations. Many of the iterative approaches belong to the group called gradient-based optimization.

5.2.3 Model Complexity

In the first calculation, it could be said that the task of supervised machine learning is to model the relationship between input output data most correctly. The problem with this definition is that in the real-world application, there have never been enough data to capture the true relationship

between the two. Therefore, the task of machine learning is the attempt to suppose the true relationship by detecting incomplete picture.

Hence the most significant property of the machine learning model is its generalization capability. That is the capability to produce meaningful outcomes from data that were not previously detected.

Generalization ability is reliant on the complexity of the model and its relationship to the complexity of the underlying problem. When the model does not capture the complexity of the problem appropriately, it is defined as underfitting. In case the complexity of model surpasses the complexity of the underlying problem, then this phenomenon is called overfitting.

In both extremes, the generalization ability suffers. In the earlier case, the model is unable to capture true intricacies of the problem and therefore is unable to predict wanted output reliably. In the last case, it tries to capture even the subtlest data perturbation that might be in fact an outcome of the stochastic nature of the problem and not the real underlying relationship. This can also cause the fact that input data is lost some variable necessary to capture the true relationship. This fact is inescapable, and it thus must be careful when designing a machine learning model. Representation of this phenomena in the case of two variable inputs is shown in Figure 5.2.

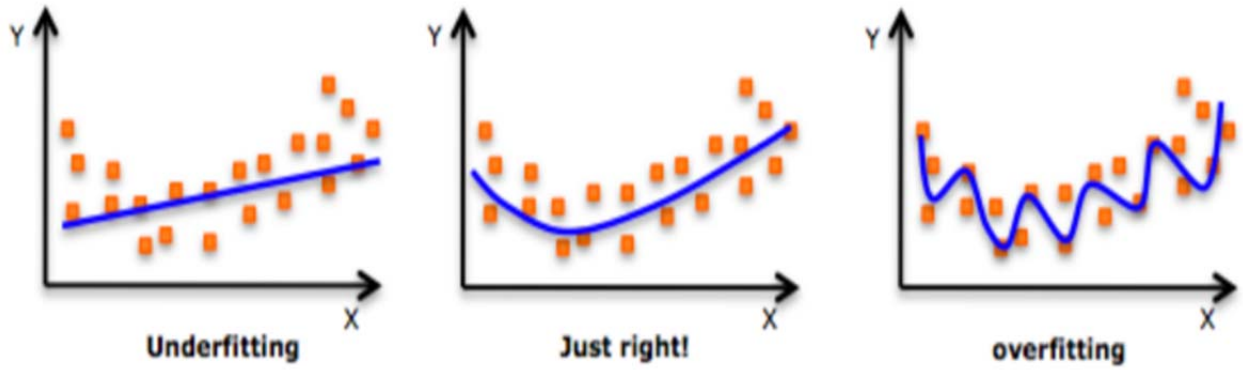

Figure 5.1: Figure shows different levels of generalization of the model [56]

Typically, the machine learning model is trained on as much input data as possible in order to reach the best possible performance. At the same time, its error rate must be verified with independent input data to check whether the generalization ability is not deteriorating. This is typically accomplished by splitting available input data into training and testing set, frequently in 4:1 fraction of training to test data. The model is trained with training data only and the presentation of the model tests on the test data. A connection between test and train error can be found in Fig. 5.3. Even though the true generalization error can never be truly detected, its estimate of the test error rate is enough for most machine learning tasks.

Regularization

Regularization is any alteration that is made to the learning algorithm that is intended to decrease its generalization error, but not its training error [57]. As it has already been stated, the most significant aspect of machine learning is striking the stability between over and under fitting of the model. To support this problematic concept of regularization was devised. It is a method that helps to penalize the model for its complexity. The basic idea consists of adding a term in the cost function that increases with model complexity. When this is applied to cost function from equation 5.7

$$J(\theta) = (y - X\theta)^T(y - X\theta) + \lambda\theta^T\theta\,, \tag{5.10}$$

where λ is a parameter that controls the strong point of the preference [57].

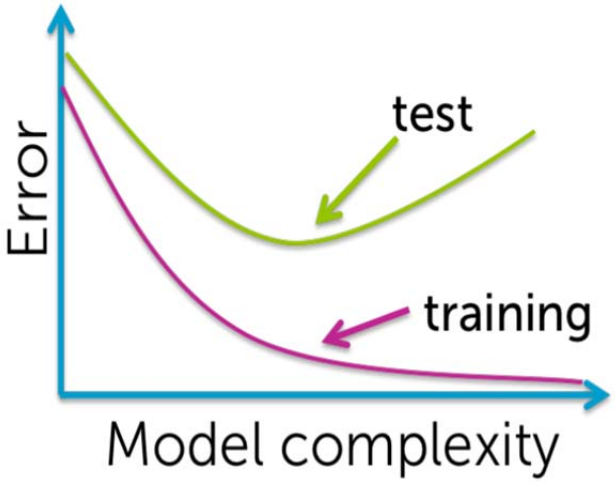

Figure 5.2: Relationship between the model complexity and its ultimate accuracy is the relationship between training and testing error [58].

CHAPTER 6

NEURAL NETWORKS

This chapter is devoted to the description of NN in general and its special type called CNN. Also, provided more details about cost function and optimization. I applied the categorical cross entropy cost function for multiple classes classification. I have used Adam optimizer in my research.

6.1 History

The history of Neural networks can be arguably dated from 1943 when Warren McCulloh and Walter Pitts invented mathematical model encouraged by the Biology of central nervous systems of mammals [59].

This encouraged the invention of Perceptron, created in 1958 by Frank Rosenblatt. Perceptron used very modest model mimicking biological neuron that was based on the mathematical model of Pitts and McCulloh. Definition of the Perceptron model also defined an algorithm for direct learning from data.

In the beginning, Perceptron looked very promising, but it was soon discovered that it had severe restrictions. Most projecting voices of criticism were Marvin Minsky. Minsky published a book in which he laid out a case that the Perceptron model was unable to resolve complex problems [60]. Amongst others, the book contained mathematical proof that Perceptron is incapable of solving a simple XOR problem. More generally the Perceptron is only proficient of solving linearly separable problems. However, according to Minsky, this criticism wasn't malicious; it in effect stifled the interest in NNs for over a period.

Awareness in NNs was rejuvenated in the early '80s when it was shown that any previously raised up deficiencies could have been resolved by the usage of multiple units. This was later exacerbated by the development of the back-propagation learning algorithm, which allowed the possibility to gather neurons into groups called layers, which can be weighted into hierarchical structures to form a network. NN of this type was generally called Multilayer Perceptron (MLP). In the 80s and 90s, the awareness in NNs plateaued again, and general research on AI was more focused on other machine learning methods. In the field of classification problems, it was particularly SVM and ensemble model. AI research communities also established several other paradigms of NNs that was likewise inspired by Biology of a certain aspect of the central nervous system but took different methods. Most significant examples were Self Organizing Maps and Recurrent Neural Network (RNN).

By the year 2000, there were very few research groups that were applying enough attention to the NNs. There was also a certain disdain for NNs in the academic world and AI research

community. The success of NNs that was promised almost half a century ago was finally coming across around 2009 when the first networks with a huge number of hidden layers were effectively trained. This led to a typical adaptation of umbrella term deep learning which by and large refers to Deep Neural Network (DNN). The term deep indicates that networks have many hidden layers.

The key theoretic vision was to learn complex functions that could represent high-level abstractions such as vision recognition, language understanding, etc. There is a requirement for deep architecture.

NNs in the times before Deep Neural Networks had only one or two hidden layers. These are currently called shallow networks. Typical Deep Networks can have a number of hidden layers in order of 10's but in some cases even hundreds [61].

Still, the progress of Neural Network into the direction of structures with a high number of hidden layers was obvious; its training was an unresolved technical problem for a very long time. There were fundamentally three reasons why this invention didn't come sooner

1. There was no procedure allowing the number of hidden layers to measure.

2. There wasn't enough of labels data required to train the NN.

3. The computer hardware wasn't powerful enough to train adequately large and complex networks successfully.

The first problem was tackled by the creation of CNN's [62]. The second problem was explained simply when there were more data presented. This was primarily achieved thanks to an effort by large companies like YouTube, Google, Facebook, etc. But also, with the support of a large community of experts and hobbyists in data sciences.

Both inventions in computational hardware and improvement of training methods were needed to resolve the third problem. One of the technical revolutions was the use of Graphics Processing Units (GPUs) for the demanding computation involved in the training of a complex network. Thanks to the fact that the training process of NNs is typically a large number of simple resulting computations, there is a great possibility for parallelization.

6.2 Structure of Neural Networks

The term NN is very general and it defines a comprehensive family of models. In this framework NN is distributed and parallel model that is capable of approximating complex nonlinear functions. The network is made from multiple computational components called neurons assembled topology.

Explanation of the NN structure will follow the convention laid out in the explanation of the learning algorithm. Meaning that an explanation of the learning algorithm is composed of the model, cost function and optimization technique. The difference comes into performance with the fact that the model of NN is much more complicated than the model linear regression. Therefore, the investigation is divided into a model of neuron and topology of the network.

6.2.1 Model of Neuron

A neuron is a computational unit carrying out the nonlinear transformation of its inputs

$$y = g(w^T x + b). \tag{6.1}$$

Argument $w^T x + b$ of function g is often observed as z. Therefore, the equation can be rewritten as

$$y = g(z). \tag{6.2}$$

The typical schema is shown in Figure 6.1, which describes the inputs, weights bias and activation function.

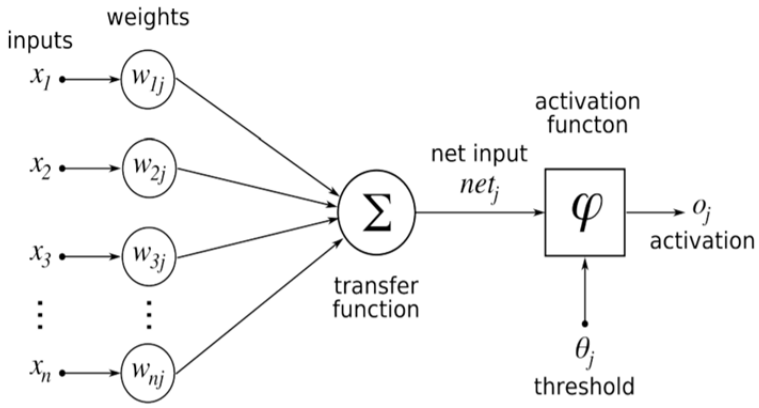

Figure 6.1: Diagram of the artificial neuron [63].

As it was already stated, model of the neuron was stimulated by biology. First attempts to make a model of a neuron had multiple elements equivalent to neurons of the human brain. As research proceeded, this equality ceased being as important and modern NN models correspond to their biological matching part only superficially.

Inputs

For each neuron has multiple inputs *x* that are combined to execute some operation. Each input has selected weight assigned to it. Here, we have considered an input of images with the size 50x50x1 (Height X width X Channel) pixels. If we input this to our model Convolutional Neural Network, we will have about 2500 weights in the first hidden layer itself.

Weights

Inputs of a neuron are weighted by parameters *w* that are changed during the learning process. Each weight gives strength to each individual input into the nerve cell. The basic awareness is that when the weight is small the input doesn't affect the output of the neuron very much. Its effect is large in the opposite case.

Bias

Another changeable parameter is bias b that controls the impact of the neuron.

Activation Function

For NN to estimate nonlinear function, each neuron must perform the nonlinear transformation of its input. This is completed with activation function $g(z)$ that performs the nonlinear transformation. There are numerous different normally used activation functions. Its usage depends on the type of network and on the type of layer in which they activate.

One of the oldest and historically most frequently used activation functions if sigmoid function. It is defined by

$$g(z)=\frac{1}{1+e^{-z}} \qquad (6.3)$$

Problem with sigmoid is that its gradient becomes flat on both extremes and as such it reduces the learning process [64].

One more activation function is the hyperbolic tangent. It is defined as

$$g(z) = tanh(-z). \qquad (6.4)$$

The hyperbolic tangent function doesn't use that much in feedforward NN, but it is mostly used in RNN. Currently, the most commonly used activation function is restricted Linear Unit (ReLU). It is very generally used in both convolutional and fully connected layers. It is defined by

$$g(z) = max\{0, z\}. \qquad (6.5)$$

It has a disadvantage because it is not differentiable for $z = 0$, but it is not a problem in software execution and one of its biggest advantages is that it can learn very speedily. In this research,

ReLU activation function was used as it is generally used in convolution and fully connected layers.

All three activation functions are illustrated in Figure 6.2.

6.2.2 Topology of the Network

There are several different generally used topologies. The two most frequently used in deep learning are feed-forward and recurrent. Feed forward networks are categorized by the fact that during activation the information moves only in a forward direction from inputs to outputs. A recurrent network has provided some sort of feedback loop.

Another principle of topology is how are individual neurons in the network linked. Most commonly are NNs ordered in layers. In each layer, there can be from one to n neurons. Layers are hierarchically fixed. The first layer is called the input layer, the last layer is called an output layer and the layers intermediate are called hidden.

Description of the network recreations on interconnections between individual layers. The most common structure is called fully connected where to each neuron in hidden layer l has input associates from all neurons from previous layer $l - 1$ and its output is associated with the input of each neuron in following $l + 1$ layer. The entire structure is illustrated in Figure 6.3.

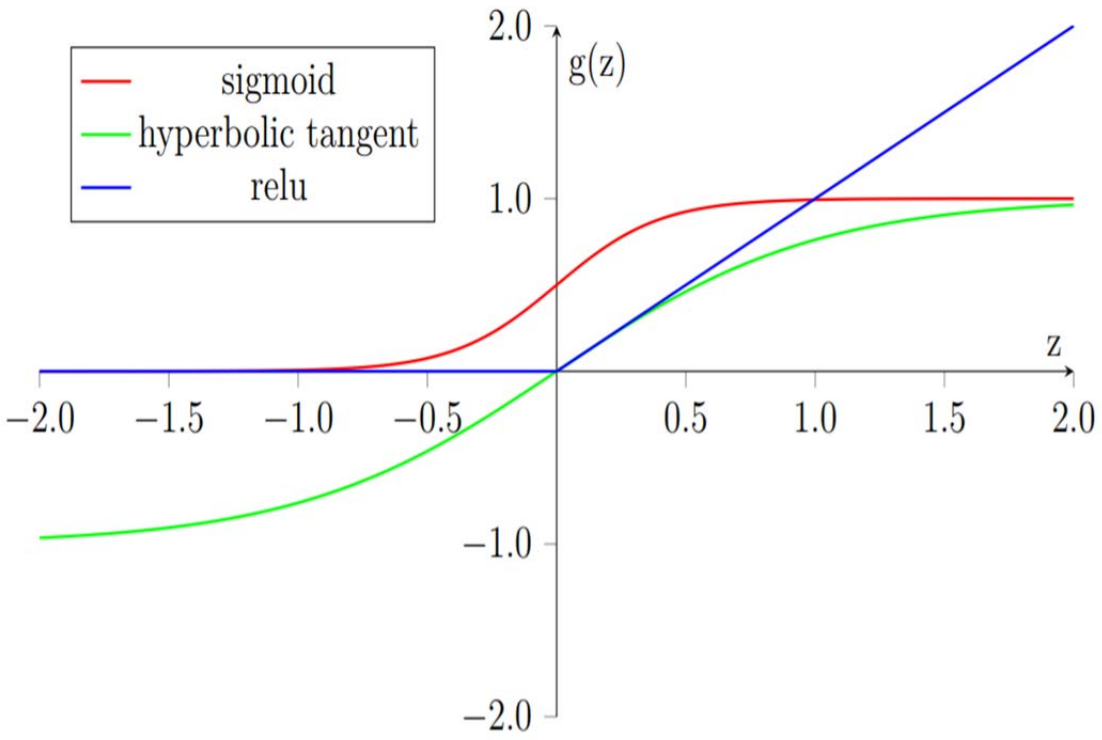

Figure 6.2: Activation Functions [65]

After this point on the term, NN will refer to Feed-forward Fully Connected Neural Network.

Types of neurons are dependent on the type of layers provided to the network. Currently, the core difference is in their activation function, which wasn't the case for a long time. In history, all layers had neurons with a sigmoid activation function. It was mostly because the output sigmoid layer can be easily mapped onto probability distribution since it obtains values between 0 and 1. Only relatively recently it was found that network composed of neurons with ReLU activation function in the hidden layers can be trained very speedily and are more resistant against over-fitting. Activation functions are still subject to ongoing research.

Neurons in output layer necessitate output that can produce a probability distribution

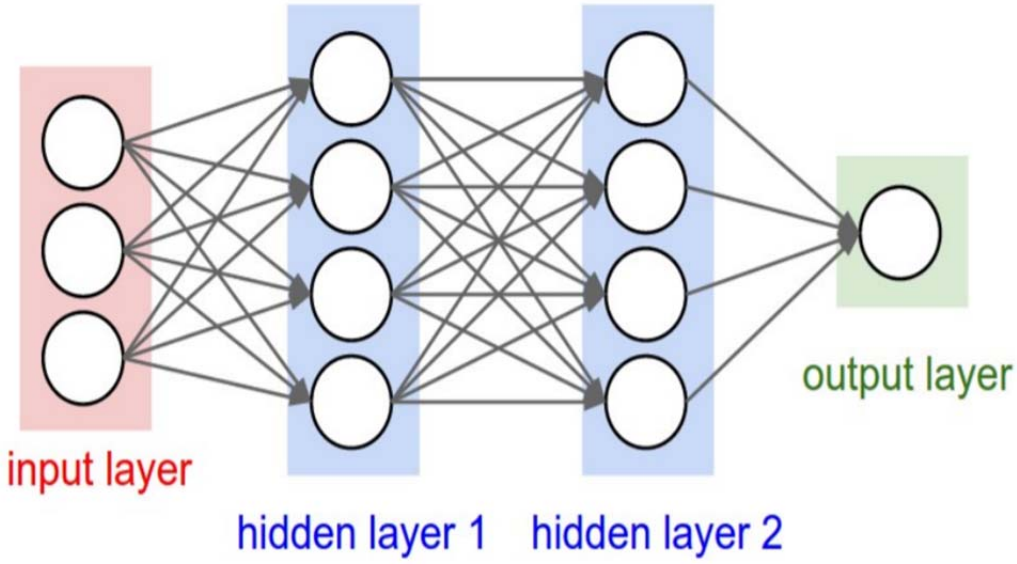

Figure 6.3: Fully connected Feed Forward Neural Network [65].

that can be used for approximating the probability of individual classes. For this reason, most frequently used activation function of output neuron is called SoftMax.

SoftMax is a standardized exponential function. It is used to represent the probability of an instance existence member of class *j* as

$$g(z)_j = \frac{e^{z_j}}{\sum_{K}^{k=1} e^{z_k}}, \tag{6.6}$$

*w*here K is the total number of classes.

6.2.3 Cost Function

Cost functions of NNs are a complex subject that exceeds the scope of this thesis. One of the most common cost functions used in NNs for classification in multiple classes is categorical cross entropy. In SoftMax activation function from Equation 6.6 is a cost function defined as

$$C = -\frac{1}{n}\sum_{i=1}^{n} y^{(i)}\, ln\, g\left(z^{(i)}\right) + \left(1 - y^{(i)}\right) ln\left(1 - g\left(z^{(i)}\right)\right), \tag{6.7}$$

where $y^{(i)}$ is the correct class of the instance and n is the total number of instances.

6.2.4 Optimization Procedure

Every optimization technique for NN is constructed on gradient descent. It is an iterative process to lower training error of the network by differentiating the cost function and adjusting parameters θ of the model by following the negative gradient.

The problem is that the cost function of the whole network is very complex and has many parameters. To find the gradient of the cost function, it is essential to go through all the units in the network and estimate their contribution to the overall error. A method that is used to solve this problem is called back-propagation.

Back-propagation is frequently confused to be a complete learning algorithm which is not the case, it is only the method to compute the gradient [57].

Back-propagation

To approximate the influence of individual units in a network the back-propagation is used to compute a delta δ_j^l, where l is layered and j is an index of neurons in that layer. The algorithm starts at the output of NN, more exactly its cost function.

$$\delta^L = \nabla_x \mathrm{C} \odot g'(z^L) \tag{6.8}$$

where L is the last layer of the network and $\nabla_x C$ is the gradient of the cost function with respect to x and $\odot$ is the Hadamard product[3].

In subsequent lower layers the deltas are calculated as

$$\delta^l = ((\omega^{l+1})^T \delta^{l+1} \odot g'(z^l) \tag{6.9}$$

where $(\omega^{l+1})^T$ is from Equation 6.1.

Each neuron has two changeable parameters b and ω. To estimate the rate of change in parameter b_j^l from Equation 6.1 needs to be computed as

$$\frac{\partial C}{\partial b_j^l} = \partial_j^l \tag{6.10}$$

Change of weight w_{jk}^l from Equation 6.1 needs to be computed as

$$\frac{\partial C}{\partial w_{jk}^l} = x^{l-1}\delta_j^l \tag{6.11}$$

Gradient Descent Optimization

Back-propagation estimates gradient of all modifiable parameters b and w in the network. These parameters can be denoted by vector θ. Thus, the gradient of the role to be minimized can be written as $\nabla_{\theta_{t-1}} f(\theta_{t-1})$.

The modest learning algorithm is called gradient descent. Even though simple, it is a very robust learning algorithm.

$$g_t \leftarrow \nabla_{\theta_{t-1}} f(\theta_{t-1}) \tag{6.12}$$
$$\theta_t \leftarrow \theta_{t-1} - n g_t \tag{6.13}$$

The algorithm has meta-parameter η, which is often called the learning rate. It determines how fast θ parameters are updated. Modest gradient descent has the shortcoming that update of parameters has always been closely proportional to the change of gradient. This might turn out to

be a problem when the gradient change slows down. This process is often called Stochastic Gradient Descent (SGD). The word stochastic indicates that during training the algorithm is using a random choice of instances to train. There are various variations on the gradient descent method. Following explanations are taken from [66].

Adam

It is a more complex learning algorithm that combines L_2 norm and classical momentum-based optimization. It should converge quicker than classical Gradient Descent. In this research, Adam optimizer was used.

$$g_t \leftarrow \nabla_{\theta_{t-1}} f(\theta_{t-1}) \tag{6.14}$$

$$\mathrm{m}_t \leftarrow \mu \mathrm{m}_{\mathrm{t}-1} + (1-\mu) g_t \tag{6.15}$$

$$\widehat{m_t} \leftarrow \frac{m_t}{1-\mu^t} \tag{6.16}$$

$$n_t \leftarrow v n_{t-1} + (1-v) g_t^2 \tag{6.17}$$

$$\widehat{n_t} \leftarrow \frac{n_t}{1-v^t} \tag{6.18}$$

$$\theta_t \leftarrow \theta_{t-1} - n \frac{\widehat{m_t}}{\sqrt{n_t} + \epsilon} \tag{6.19}$$

6.3 Convolutional Neural Networks

Convolution Neural Networks (CNN) are a specialized type of Neural Networks that was initially used in image processing applications. They are arguably the most effective models in AI inspired by biology.

Even though they were shown by many different fields, the main design principles were drawn from neuroscience. Since its' achievement in image processing, they were also very successfully implemented in natural language and video processing applications.

Stimulation in biology was based on the scientific work of David Hubel and Torstein Wiesel. Neurophysiologists Hubel and Wesel studied vision system of mammals from late 1950 for several years. In the research, that might be measured little gruesome for today's standards, they linked electrodes in the brain of an anesthetized cat and measured brain response to visual stimuli [67]. They discovered that feedback of neurons in the visual cortex was triggered by a very narrow line of light shined under a specific angle on a projection screen for the cat to see. They determined that specific neurons from the visual cortex are responding only to very specific patterns in the input image. Hubel and Wiesel were given the Nobel Prize in Physiology and Medicine in 1981 for their discovery.

In the following text is assumed that convolutional layer is working with rectangular input data (e.g. images), even though the Convolutional networks can also be used to classify one-dimensional or three-dimensional input.

6.3.1. Architecture

Our architecture is commonly used in CNN architecture. This architecture consists of multiple convolution and dense layers. The CNN architecture includes three types of three convolution layers, and each layer has its max pooling layer and one group of fully connected layer followed by a dropout layer and the output layer as shown in Figure 6.4.

The CNN architecture was implemented using the Kera's deep learning framework with Python and Tensor flow backend. The same network was later used for alphabet classification. The proposed CNN contains three convolutional layers and three max-pooling layers. The only fully-connected layer is the final layer and the layer after the last convolutional layer. The input layer takes in batches of images of size 50×50×1. ReLU activation function was used between the hidden layers as explained in Section 6.2, and a SoftMax activation is used in the output layers multiple class classification and detection. The first convolutional layer has 16 filters of size 3×3.

The filter is also called the weight. Filter is used to extract feature from the image. It is followed by a pooling layer that uses the max pooling operation with the size of 2×2. When pooling is applied in a forward direction, it is called Forward Propagation. Max pooling is used for sub sampling from an image. The second convolutional layer has 32 filters with a capacity of 3×3.

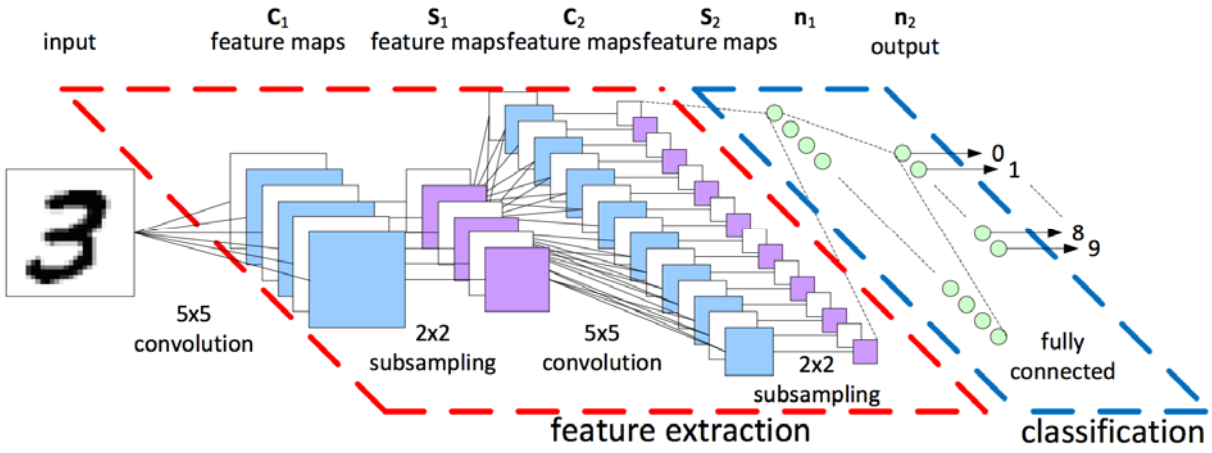

Figure 6.4 Convolutional Neural Network (CNN) Architecture[68]

Similarly, to the first convolutional layer, it is followed by a max-pooling layer with the kernel size of 2×2. The third convolutional layer has 64 filters with the same kernel size as a previous convolutional layer. The 2×2 max pooling is applied yet again. The parameters of the described layers are also illustrated in Figure 6.5.

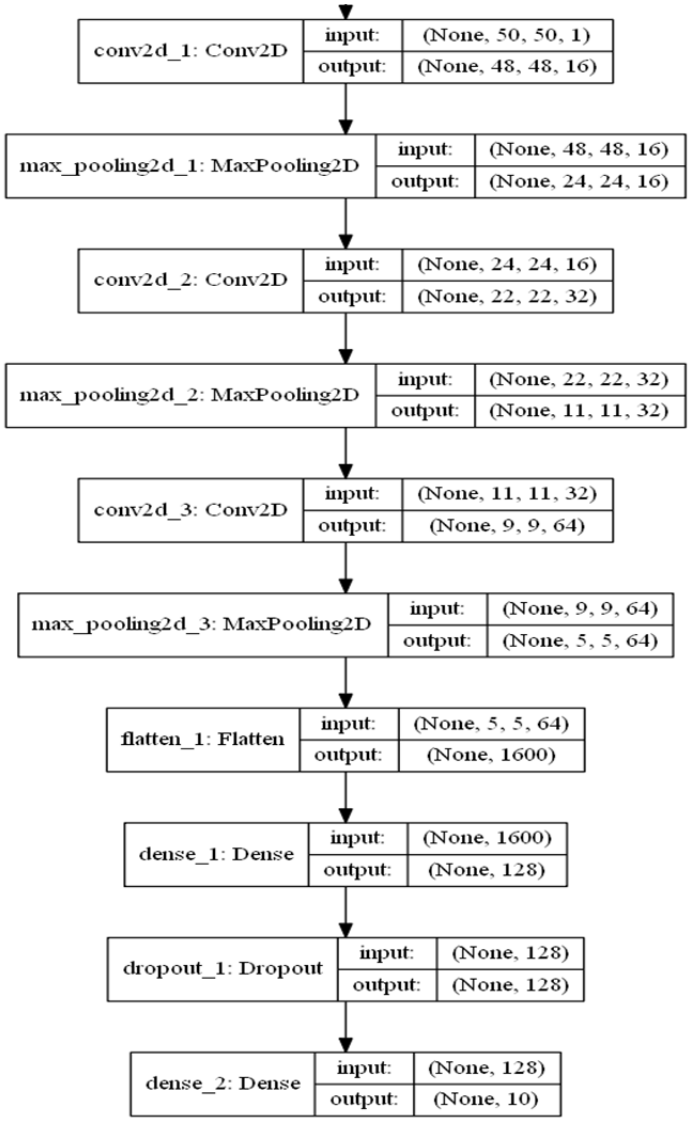

Figure 6.5. CNN network architecture for Numbers.

1. Convolutional layer with 16 feature maps of size 3×3.
2. Pooling layer taking the max over 2*2 patches.
3. Convolutional layer with 32 feature maps of size 3×3.
4. Pooling layer taking the max over 2*2 patches.
5. Convolutional layer with 64 feature maps of size 3×3.
6. Pooling layer taking the max over 2*2 patches.
7. Dropout layer with a probability of 20%.
8. Flatten layers.

9. Fully connected layer with 128 neurons and rectifier activation.
10. Fully connected layer with ten neurons and rectifier activation.
11. Output layer.

6.3.2 Training the CNN

An optimization procedure for CNN is similar to Fully Connected Neural Network. CNNs are more complex because the network is made of different types of layers. Forward signal propagation and backward error propagation follow a special protocol for each layer. Calculations used in this section were inspired by [71]. The first stage is called forward-propagation, where the signal is broadcast from the inputs of CNN's to its output. In the last layer, the output is associated with the desired value by cost function and error is estimated in the second stage uses the backpropagation algorithm to estimate the error contribution of individual units. Inconstant parameters of the network are again optimized by gradient descent algorithm.

Forward Propagation of Convolution Layer

Each convolutional layer is performing a convolution process on its input. Presuming that the input of a layer is of length N × N (50x50) units and the kernel is of length m × m (3X3). Convolution is calculated over (N−m+1) × (N−m+1) units without zero paddings.

Calculation of convolution output x_{ij}^{l} is defined as 0

$$x_{ij}^{l} = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} w_{ab} y_{(i+a)(j+b)}^{l-1} \tag{6.21}$$

where $i, j \epsilon (0, N-m+1), 1$ is index of current layer, w_{ab} are weights of the kernel and $y_{(i+a)(j+b)}^{l-1}$is output of previous layer. Output of convolutional layer y_{ij}^{l} is computed by squashing of output of convolution operation x_{ij}^{l} through non-linearity:

$$y_{ij}^{l} = g\left(x_{ij}^{l}\right) \tag{6.22}$$

where g represents this non-linear function.

Backward Propagation of Convolution Layer

Backward propagation for the convolutional layer follows similar principles as described in Section 6.2.4. The difference is in the fact that convolution kernel shares weights for the entire layer and kernels do not have bias described in Section 6.2.1. Given partial derivative of error from the previous layer with respect to the output of the convolutional layer $\frac{\partial \mathrm{C}}{\partial y_{ij}^{l}}$, influence of the kernel weights on the cost function has to be calculated as

$$\frac{\partial \mathrm{C}}{\partial w_{ab}} = \sum_{i=0}^{N-m} \sum_{j=0}^{N-m} \frac{\partial \mathrm{C}}{\partial x_{ij}^{l}} \frac{\partial x_{ij}^{l}}{\partial w_{ab}} \tag{6.23}$$

From Equation 6.21 it follows that $\frac{\partial x_{ij}^{l}}{\partial w_{ab}} = y_{(i+a)(j+b)}^{l-1}$,Thus

$$\frac{\partial \mathrm{C}}{\partial w_{ab}} = \sum_{i=0}^{N-m} \sum_{j=0}^{N-m} \frac{\partial \mathrm{C}}{\partial x_{ij}^{l}} y_{(i+a)(j+b)}^{l-1} \tag{6.24}$$

To calculate deltas (equivalent to Equation 6.9) $\frac{\partial C}{\partial x_{ij}^{(l)}}$ using the chain rule

$$\frac{\partial \mathrm{C}}{\partial x_{ij}^{(l)}} = \frac{\partial C}{\partial y_{ij}^{l}} \frac{\partial y_{ij}^{l}}{\partial x_{ij}^{l}} = \frac{\partial \mathrm{C}}{\partial y_{ij}^{l}} \frac{\partial}{\partial x_{ij}^{l}} \left(g'\left(x_{ij}^{l}\right)\right) = \frac{\partial \mathrm{C}}{\partial y_{ij}^{l}} g'\left(x_{ij}^{l}\right) \tag{6.25}$$

Since $\frac{\partial \mathrm{C}}{\partial y_{ij}^{l}}$ is already given, the deltas are calculated by the derivatives of the activation function. Last phase comes to propagation of error in previous layers by equation

$$\frac{\partial C}{\partial y_{ij}^{l-1}} = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} \frac{\partial C}{\partial x_{(i-a)(j-b)}^{l}} \frac{\partial x_{(i-a)(j-b)}^{l}}{\partial y_{ij}^{l-1}} \tag{6.26}$$

Again, from Equation 6.21 it follows that $\frac{\partial x_{(i-a)(j-b)}^{l}}{\partial y_{ij}^{l-1}} = w_{ab}$, therefore

$$\frac{\partial C}{\partial y_{ij}^{l-1}} = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} \frac{\partial C}{\partial x_{(i-a)(j-b)}^{l}} w_{ab}. \tag{6.27}$$

The result looks suspiciously like convolution operation and can be taken as the convolution of error with the flip kernel.

Forward Propagation of Pooling layer

Feedforward operation of pooling layer is straight forward as defined in section 6.3.1. The ratio is typically 4:1, which means that the input matrix is divided into non-overlapping sub-matrices of size 2×2 and each of these produces 1 output. Another possibility is to have overlapping sub-matrices, where the length of sub-matrix is larger than the number of pixels between the application of pooling.

Backward Propagation of Pooling Layer

As in forward-propagation, there is no explicit learning process happening in the pooling layer. The error is propagated backwards dependent on how the signal was propagated forward. In Max-Pooling layer the error is propagated only to the unit with maximal output in forward-propagation phase. The error is propagated very sparsely, as an outcome.

6.4 Regularization of Neural Networks

Control of difficulty applies to both NN and CNN. There are several popular regularization techniques that typically consist of adjustment of the cost function or optimization algorithm. The somewhat different approach is to modify the structure of the network during the training stage.

Dropout by far is the greatest regularization technique to combine the predictions of many different models. This technique greatly improves the generalization ability of a combined model while preventing over-fitting. Exactly on this idea is based, ensemble model. The problem with ensemble models is that they require more computational time and are very expensive. Because of this, ensembles are usually made of many simple models [69]. This idea is especially problematic with Deep Neural Networks (DNN), which are models with many parameters that are difficult to train. Moreover, even when trained models exist in some applications, it still isn't viable to evaluate many different models in a production environment. Another issue is that there might not be enough data to train these different models.

All these problems can be resolved by dropout technique. The basic idea is that each neuron in the network has a certain probability of being deactivated during one iteration. The potential for deactivation is estimated in every iteration, to ensure that the network has a different architecture every time. Deactivated means that it will not broadcast any signal through. This forces individual neurons to pick up features that are less dependent on their surrounding.

Possibility for deactivation is a hyperparameter that can be tuned, but a reasonable default value is 0.5. Dropping out is only happening in the training stage. In the testing stage, all weight connections are multiplied by the probability of a dropout. This is completed because the activation of the network must stay roughly equivalent in both training and testing stages. The basic concept is shown in Figure 6.6

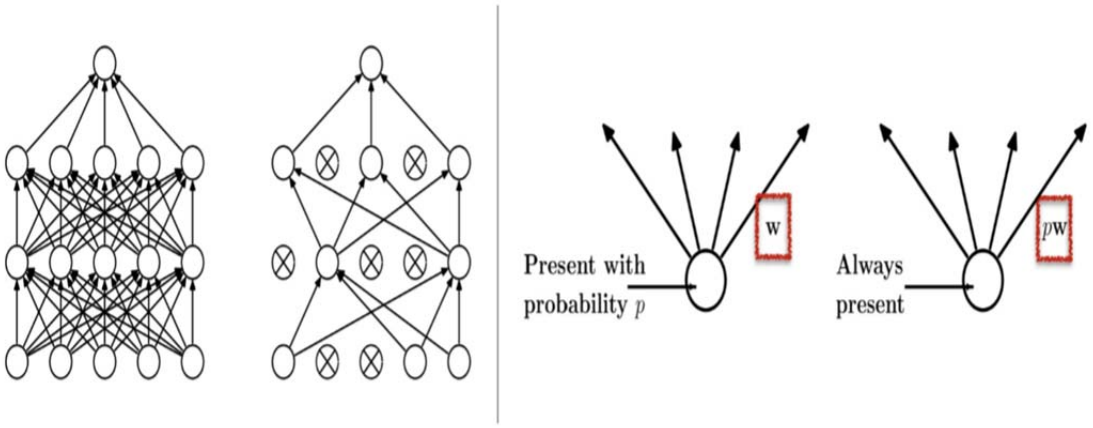

Figure 6.6: Dropout: (a) Standard fully connected network. (b) Network with some neurons deactivated. (c) Activation of a neuron during the training phase. (d) Activation of the neuron during testing phase [70].

Chapter 7

Methods and Results

In this chapter we describe the proposed methods and results. The first part consists of software tools in this study. The next part is a description of the software and hardware configuration of testing equipment. In the next phase preparation of dataset is described. The last portion of this chapter is dedicated to specifics of implementation and results.

7.1 Software tools

To select a suitable tool for the implementation of CNN for classification, the search of available software tools and libraries was conducted. Nowadays there are different software tools available for machine learning. Some of these are universal tools for machine learning, but some are exactly designed for deep learning. In the last decade, the software tools for machine learning have undergone a renaissance. There is a wide selection of available tools, and new tools are announced quite frequently. For example, Caffe21 was presented very recently on April 18th. Almost every commonly used programming language has either some software add-on library or at least some available Application Programming Interface (API). Several aspects influenced the choice of the software tool. Firstly, the programming language should be well known and most common. Enough learning documentation materials had to be available. The most significant factor was excellent support for learning on the GPU.

Theano

It is tested python library. Designed to describe, optimize and evaluate mathematical expressions with multi-dimensional arrays. This makes it suitable for machine learning needs. Theano is made on top of NumPy, which is a python module that enables effective operation with tensors and basic image processing procedure. The mixture of NumPy and SciPy brings a rich set of tools for image processing and data processing. Its abilities can arguably rival MATLAB while being open source and free of cost. Theano's major rival is currently TensorFlow development. One of the drawbacks of Theano is its low-level nature. Development of machine learning algorithms directly can be very complicated. This maybe the reason it is slowly falling by the wayside. This is also the reason why Theano as a tool is not fit for the direct implementation of CNN models.

Tensor flow

Tensor flow is a similar tool like Theano. As the name suggests, this tool is focused on effective work with tensors. It was originally created for internal use in Google for a machine learning project, but it was launched as open source in 2015. Tensor flow calculations are expressed as stateful dataflow graphs, which enables efficient support for GPU supported computation. Tensor flow is currently advertised as one of the fastest frameworks for deep learning needs. Its drawback is like Theano, in the fact that it is very low level and direct usage for the operation of Deep learning models is not perfect.

Caffe

Caffe is a deep learning tool whose goals are modular and faster. It is created by the Berkeley AI Research and by community contributors. Yangqing Jia developed the project during his Ph.D. at UC Berkeley. C++ is the programming language used to implement, but it's also available as API

for several other languages like Python. Its main drawback is its lack of quality documentation and material. This fact is partially improved by the existence of Model Zoo, which is a collection of favourite models that are available freely. Caffe was in the last years used by Facebook for example, mainly because of its performance capabilities. Caffe is more geared towards the development of large production application than it is for study purposes.

Keras

Keras is new software for machine learning written in python. It is a high-level neural network API. It is capable of running on top of either Theano or Tensor flow libraries. It is very simple with an emphasis on quick model development. It is simply extensible. At present, Keras has one of the largest communities among similar tools for deep learning. It has very good documentation and materials containing many code demonstrations and other resources that help users to get started very rapidly.

7.2 Hardware and Software Configuration

Training of Neural Networks is notoriously computationally expensive and it requires a lot of resources. From the bottom level perspective, it translates into many matrix multiplications. Modern Central Processing Units (CPUs) are not capable of such computations and therefore are not very efficient. On the other hand, modern GPUs are designed to perform exactly these operations.

At present there are two main parallel computing platforms, CUDA and OpenCL. They both have their advantages and disadvantages, but the major difference is that CUDA is proprietary, while OpenCL is available free. This divide translates into hardware productions as well. CUDA is mostly supported by NVIDIA and OpenCL is supported by AMD. NVIDIA with its CUDA platform is presently a leader in the domain of deep learning. Therefore, for the training of CNN

models, GPUs from NVIDIA was selected. The selected model was GIGABYTE GeForce GTX 1080. Detailed information about the hardware configuration is given in Table 7.1.

Table 7.1: Hardware Configuration

GPU	GeForce GTX 1080 4GB
CPU	Intel(R) Core (TM) i7-8550 CPU @ 2.00GHz
Memory	DIMM 1333MHz 8GB

From the list of considered software tools, Keras was selected as it is written in python, an easy to program language and as it satisfies all considered factors. Support for efficient GPU implementation in Keras relies on either Tensor flow or Theano back-end. From the different user perspectives, it doesn't matter either way, but Tensor flow was selected because it was observed as faster of the two and has GPU-accelerated library package of primitives for deep neural networks. Detailed information about software configuration is given in Table 7.2.

Table 7.2 Software Configuration

Keras	2.04
TensorFlow	1.1.0
CUDA	7.5
Python	3.53
Operating System	Window 10
Open CV	2.0

7.3 Model Structure

In this study, we applied the Keras sequential model for CNN, which is a concept that is suitable for modelling of a feedforward network. Definition of the network is made of layers. The concept of a layer in the Keras sequential model doesn't fully map into the already described definition of the layer from a topological viewpoint. Keras layers are fine grained and to create an equivalent topological layer, it is essential to use multiple Keras layers. A model is created simply by calling Sequential constructor

from keras.models import Sequential

model = Sequential ()

Layers are added by calling an add method on an object of a sequential model

model.add(NumbersOfLayer),

where NumbersOfLayer is the definition of the layer.

7.3.1 About Keras Layers

All models were created by the configuration of the following layers.

Convolutional layer

Convolutional layer architecture has the following structure

Conv2D(filters=num, kernel_size=(x, x), strides=(s,s), padding='valid', input_shape=shape)

where a *num* is the number of filters that the layer will have, x is the size of the kernel, so is the number of pixels in stride and input_shape description is the size of the input matrix.

Activation Function

To create activation function on the output of the layer, user can require parameter activation of the layer itself or create activation as a layer

Activation(activation_function)

where activation_function can be ’SoftMax’ or ’ReLU’. Both specifications are equal as Keras automatically applies a linear activation function for each layer.

Pooling Layer

Pooling layer can be defined as

MaxPooling2D (Pool_Size= (z, z), strides=(s, s))

where Pool_Size requires the size of pooling kernel and strides requires a number of pixels in a vertical and horizontal direction that are traversed in between application of individual pools.

Fully Connected Layer

A fully connected layer is created using the function,

Dense(total_units)

where total_units is the total number of units, which is a fully connected neuron in a specific layer.

Dropout

Similar to the activation function, to apply a dropout regularization on a layer, it has to be added after the activation function as another layer.

Dropout(prob)

where *prob* is both the probability that a unit is dropped and the coefficient by which the outputs are multiplied through forwarding evaluation.

Other

Feature extraction layers are n-dimensional. Specifically, Convolutional and Pooling layers are 2D (two dimensional). Classification layers that are created by fully connected layers are 1-D (one dimensional). To join the two, it is required to create mappings between them. For this purpose, it is required to use the following layer

Flatten ()

which takes care of necessary connections between these layers.

7.4 Results

On our self-generated dataset, we achieved an accuracy of 99.00% to detect hand gestures for alphabets and 100% accuracy was achieved to detect hand gestures for digits. Real-time testing was done with five different students and estimation per student took 20 minutes for alphabets and approximately 7 to 8 minutes for digits. Testing was done in non-controlled form, i.e. in different lighting condition and with different backgrounds. Figure 7.7 shows real time testing with different users. For alphabets 50 epochs were applied, and for digits 20 epochs were applied. Both networks used Adam optimizer and a learning rate of 0.001. Loss function was cross-entropy due to multiple-class classification. The training and testing sets contained 70:30 ratio, respectively in both models. Figure 7.1 shows the validation accuracy for different epochs for the digits. Figure 7.2 shows the validation accuracy for different epochs for alphabets. The confusion matrices of both networks are illustrated in Figures 7.3 and 7.4. From both confusion matrices, it is evident that the classification accuracy of both models is almost identical. The only difference is the number of false negatives and true positives. Recall (Equation 7.1), precision

(Equation 7.2), accuracy (Equation 7.3) and F measure (Equation 7.4) were used as classification evaluation metrics.

$$Recall = \frac{true\ positives}{true\ positives + false\ negatives} \tag{7.1}$$

$$Precision = \frac{true\ positives}{true\ positives + false\ positives} \tag{7.2}$$

$$Accuracy = \frac{true\ positives + true\ negatives}{Total\ instances\ classified} \tag{7.3}$$

$$F\text{-}Measure = \frac{2 * (Precision * Recall)}{(Precision + Recall)} \tag{7.4}$$

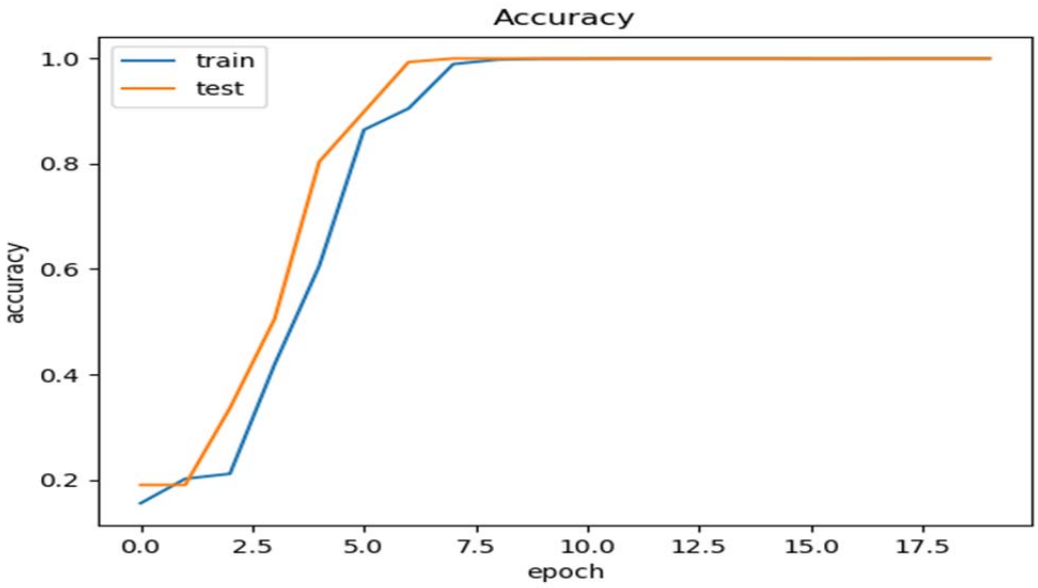

Figure 7.1. Epochs vs. validation accuracy for digits.

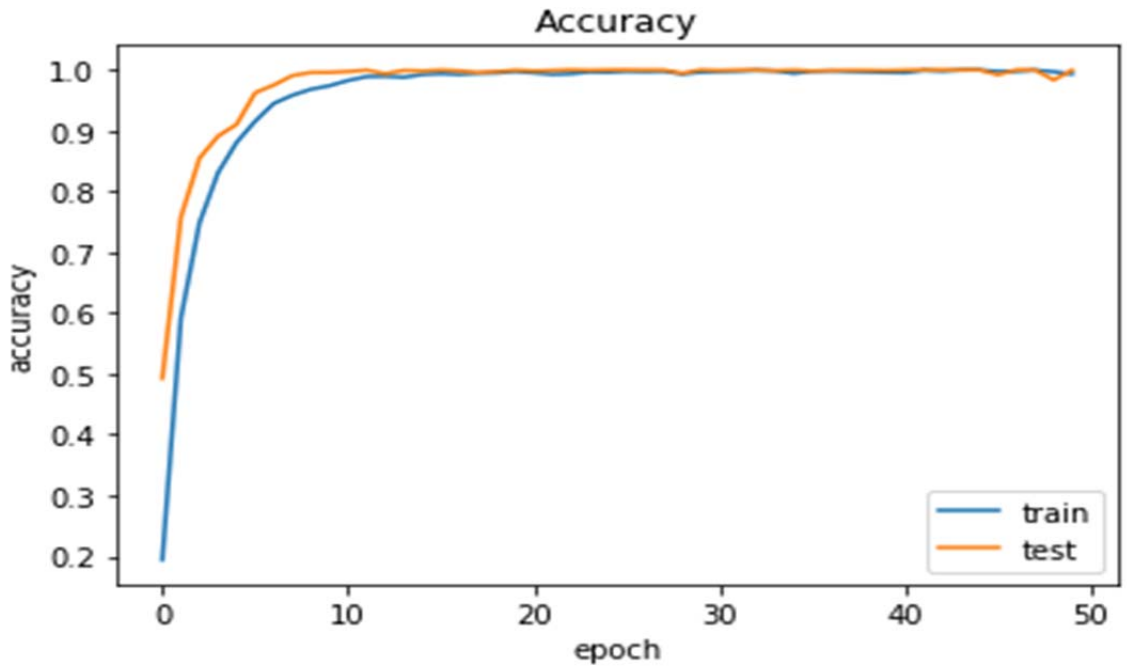

Figure 7.2. Epochs vs. validation accuracy for alphabets.

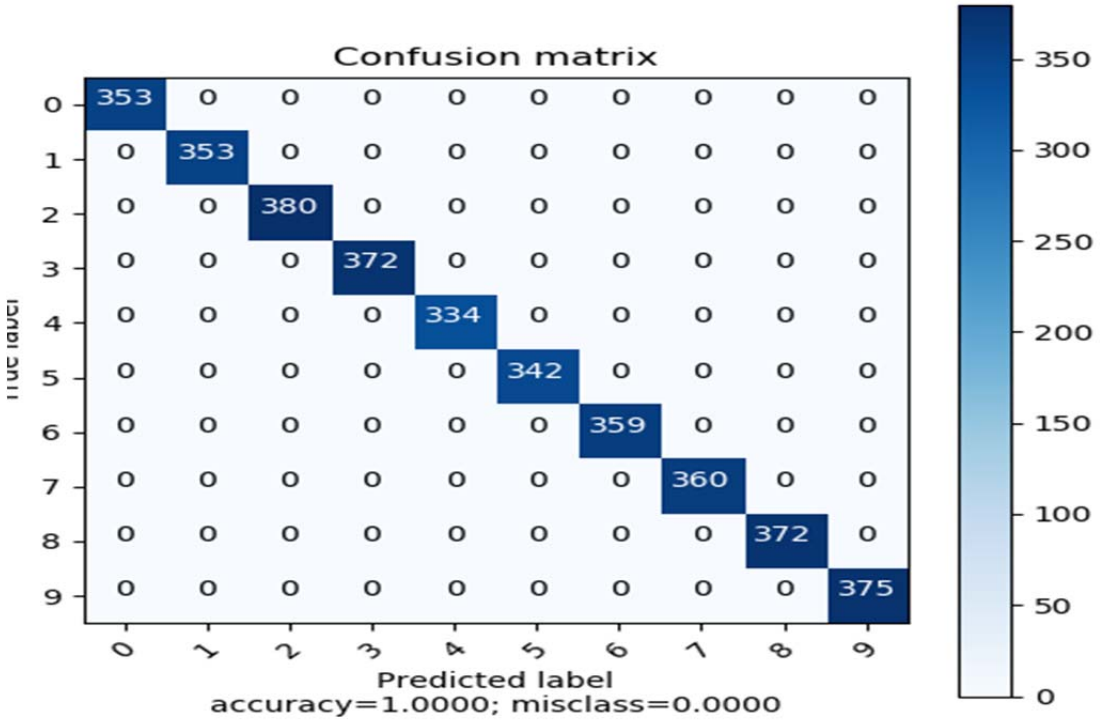

Figure 7.3. Confusion matrix for 0 to 9 digits.

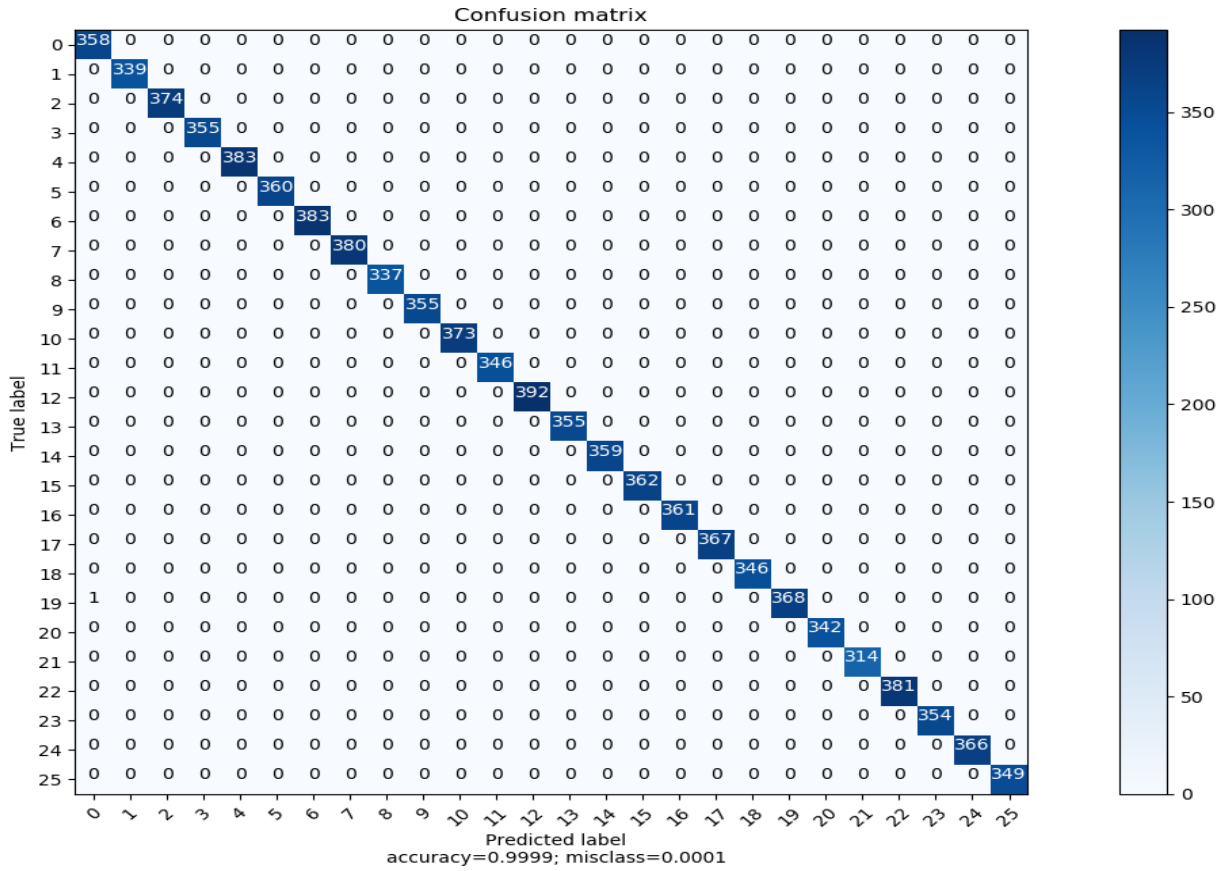

Figure 7.4. Confusion matrix for A to Z alphabets.

Figure 7.3 shows a 10x10 confusion matrix for digits. There are ten different classes for ten digits on that confusion matrix. The number of correctly classified images is the sum of the diagonal elements in the matrix, all others are incorrectly predicted. Figure 7.4 shows the confusion matrix for alphabets. In this confusion matrix, there are 26 different classes for the 26 alphabets. For computation purposes,

- TP_0 refers to the positive tuples that are correctly predicted (POSITIVE) by the classifier in the first row-first column, i.e. 353.
- TP_1 refers to the positive tuples that are correctly predicted (POSITIVE) by the classifier in the second row-second column, i.e. 353.
- TP_9 refers to the positive tuples which are correctly predicted (POSITIVE) by the classifier in the ninth row-ninth column, i.e. 375.

Therefore, the accuracy of the correctly classified images can be calculated by equation (7.3).

$$\text{i.e. Accuracy} = \frac{353+353+380+372+334+342+359+372+375}{3600} = \frac{3600}{3600} = 1.00$$

Table 7.3 shows the precision, recall, F measure, and support for the digits.

Table 7.3: Performance on Digits.

	Precision	**Recall**	**F1-score**	**Support**
0	1.00	1.00	1.00	353
1	1.00	1.00	1.00	353
2	1.00	1.00	1.00	380
3	1.00	1.00	1.00	372
4	1.00	1.00	1.00	334
5	1.00	1.00	1.00	342
6	1.00	1.00	1.00	359
7	1.00	1.00	1.00	360
8	1.00	1.00	1.00	372
9	1.00	1.00	1.00	375
weighted avg	1.00	1.00	1.00	3600

Precision: is the proportion of the classified positive cases that were actually positive. The precision can be calculated using Equation (7.1)

Example of confusion matrix from Table 7.3.

$$Precision_{\,0} = \frac{353}{(353+0)} = \frac{353}{353} = 1.00$$

$$Precision_{\,9} = \frac{375}{(375+0)} = \frac{375}{375} = 1.00$$

Weighted Average for precision for the class 0 to 9 can be given below:

$$\text{Weighted Avg} = \frac{(1*353)+(1*353)+(1*380)+(1*372)+(1*334)+(1*342)+(1*359)+(1*372)+(1*375)}{3600} = 1.00$$

Recall, also called true positive rate and sensitivity is the proportion of the positive tuples that were predicted to be positive. The recall can be calculated using Equation (7.2).

Example of confusion matrix from Table 7.3.

$$Recall_{\ 0} = \frac{353}{(353+0)} = \frac{353}{353} = 1.00$$

$$Recall_{\ 9} = \frac{375}{(375+0)} = \frac{375}{375} = 1.00$$

The weighted average for recall can be calculated by multiplying TP rate of each class with the TOTAL number of images classified for that class and dividing by a total number of samples.

Weighted Avg= $\frac{(1*353)+(1*353)+(1*380)+(1*372)+(1*334)+(1*342)+(1*359)+(1*372)+(1*375)}{3600} = 1.00$

F-Measure: The F-measure score is the harmonic mean of recall and precision. This provides us with an explanation about how the measure recall and precision values behave for the data set.

The F-score measure for all classes can be calculated using Equation 7.4.

$$F - Measure_0 = 2 * (1.0 * 1.0)/(1.0 + 1.0) = 1.0$$

$$F - Measure_9 = 2 * (1.0 * 1.0)/(1.0 + 1.0) = 1.0$$

Weighted Average for the F-Score for the class 0 to 9 can be given as below.

Weighted Avg= $\frac{(1*353)+(1*353)+(1*380)+(1*372)+(1*334)+(1*342)+(1*359)+(1*372)+(1*375)}{3600} = 1.00$

Table 7.4 shows the performance metrics for the alphabets.

Table 7.4 Performance on Alphabets.

	Precision	recall	f1-score	support
0	1.00	1.00	1.00	358
1	1.00	1.00	1.00	339
2	1.00	1.00	1.00	374
3	1.00	1.00	1.00	355
4	1.00	1.00	1.00	383
5	1.00	1.00	1.00	360
6	1.00	1.00	1.00	383
7	1.00	1.00	1.00	380
8	1.00	1.00	1.00	337
9	1.00	1.00	1.00	355
10	1.00	1.00	1.00	373
11	1.00	1.00	1.00	346
12	1.00	1.00	1.00	392
13	1.00	1.00	1.00	355
14	1.00	1.00	1.00	359
15	1.00	1.00	1.00	362
16	1.00	1.00	1.00	361
17	1.00	1.00	1.00	367
18	1.00	1.00	1.00	346
19	1.00	1.00	1.00	369
20	1.00	1.00	1.00	342
21	1.00	1.00	1.00	314
22	1.00	1.00	1.00	381
23	1.00	1.00	1.00	354
24	1.00	1.00	1.00	366
25	1.00	1.00	1.00	349
weighted avg	1.00	1.00	1.00	9360

ROC Area Under the Curve (AUC): The Receiver Operating Characteristics (ROC) charts is a method for organizing classification and visualizing the performance of the trained data using the classifier. ROC AUC for all the digit classes is 1, from which one can say that the performance evaluation of all classes is excellent in the data set. Also, the weighted value of the ROC AUC is 1 (100%) for the data classified using the CNN network, which correctly classified the data with the best accuracy and best performance. ROC AUC curves for the digits and alphabets are shown in Figures 7.5 and 7.6, respectively.

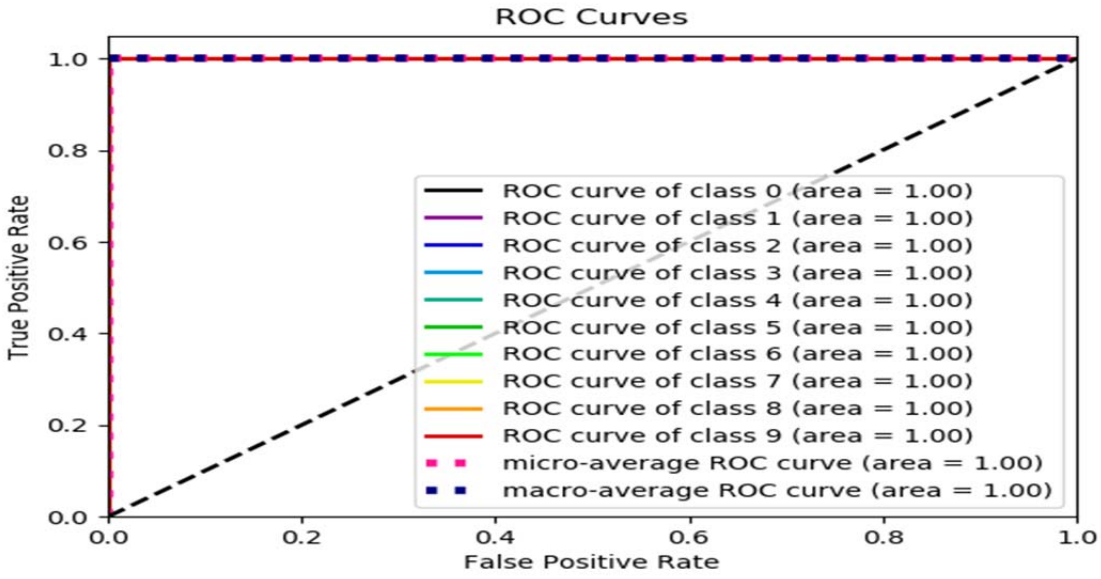

Figure 7.5. ROC AUC graph for 0 to 9 digits.

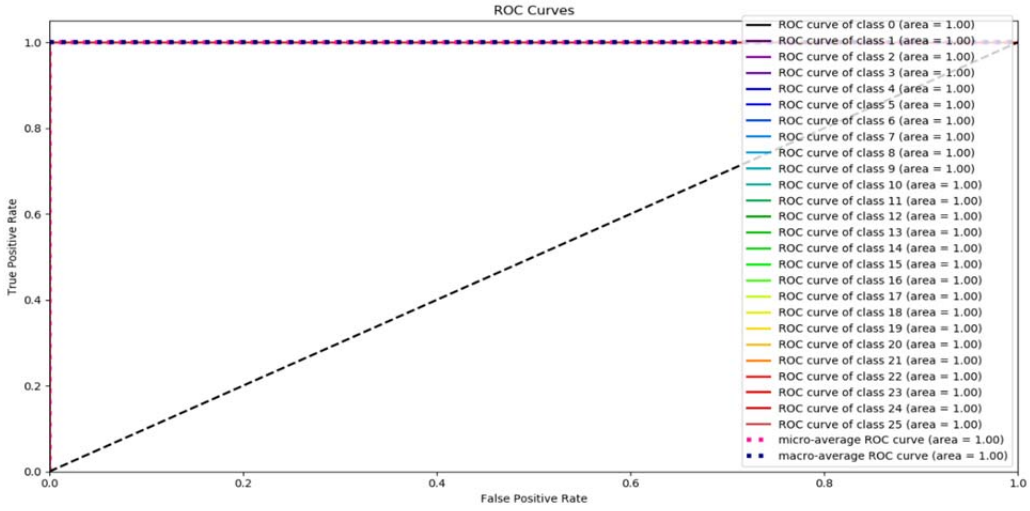

Figure 7.6. ROC AUC graph for A to Z alphabets

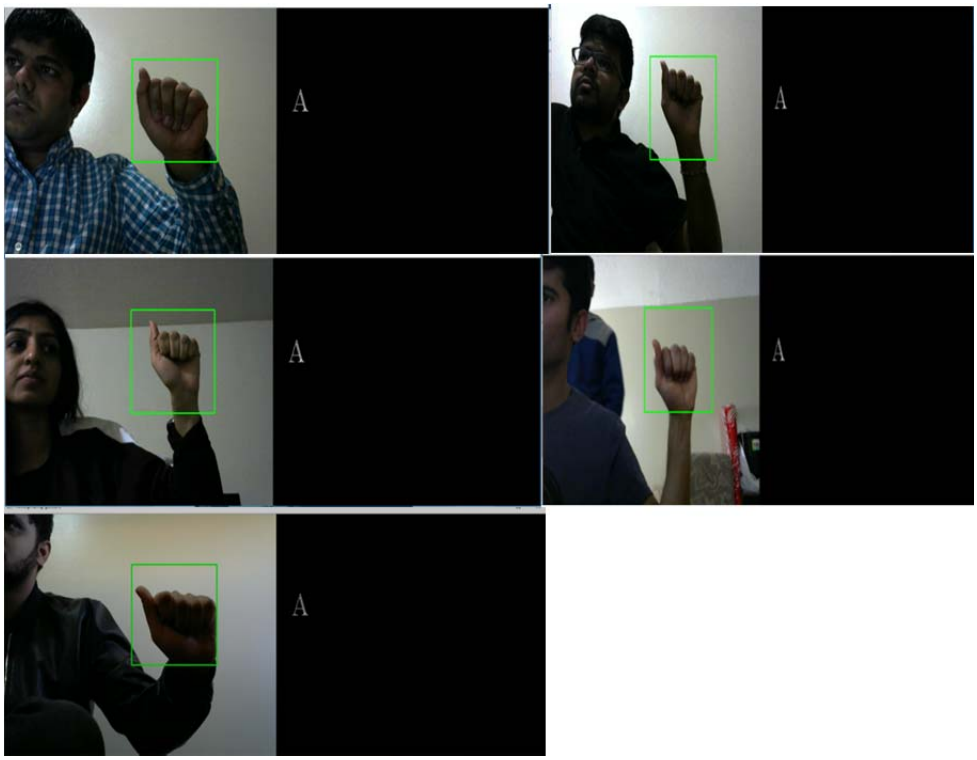

Figure 7.7 Shows real time testing with different users.

Chapter 8

Conclusion and Future work

8.1 Conclusion

One of the main purposes of the interactive virtual environments is to provide natural, efficient, and flexible communication between the user and the computer. Human gestures, including the positions and movements of the fingers, hands, and arms represent one of the richest non-verbal communication modalities allowing human users to interact naturally with the virtual environment. Sign gestures can be static, where the human takes on a specific pose or they can be dynamic with movement of hands and fingers.

Real-time vision-based sign recognition is one of the most challenging research areas in the human-computer interaction area. Vision-based sign gesture recognition can rely on generic video cameras already available on a large variety of computers, tablets, smartphones, etc. to recognize hand gestures.

This thesis proposed a static sign gesture recognition system, which works under different lighting conditions with changed transformations and cluttered background. The Sign gesture recognition systems can run a more efficient and natural interaction, a modality for artistic applications. Another important application can be in the sign language recognition for the deaf people.

A first real-time visual hand gesture recognition system contains two parts: one for hand detection and tracking using face subtraction, skin detection and contour comparison algorithms proposed in Chapter 5, and the second part that performs gesture recognition using Conventional Neural Network (CNN).

To conclude, in this thesis a real-time system was proposed that consists of three modules: hand detection and skin detection and contour comparison algorithm, gesture recognition using deep learning CNN network. The results show that the system can reach satisfactory real-time performance regardless of the frame resolution size as well as high classification accuracy of 99% for alphabets under variable scale, orientation and illumination conditions, and cluttered background. Three important reasons affect the accuracy of the system, which is the good quality of the webcam while capturing images for a dataset, the number of the training images, and choosing Conventional Neural Network model.

8.2 Future Work

Sign gesture recognition still has a long way to go on the research path, especially for 2D systems. This study offers fascinating ideas for future research. Some of these possibilities are defined in this section.

Dynamic Sign Gesture Recognition

As this thesis focused only on static sign gesture recognition, one next step forward is to recognize the dynamic sign gesture for the ASL.

Sign Gesture Recognition from a video

Nowadays videos are generally found on the internet. The idea of categorizing single frames is a start to classifying frames in videos. This can be applied in real time classifications. Extending the algorithms proposed in this thesis to video and building an automatic transcript system is an important step onward. For this purpose, it might be fascinating to explore sequential models that study the time dimension, such as recurrent neural networks or a neural network architecture combining CNN's and RNN's.

Apply to 3 Dimension technique

Nowadays 3-Dimensional cameras and sensors are very easily available in the market and less expensive. This different type of sensor can provide much more information about the hand, making it possible to create more accurate systems for real-time sign language recognition.

Add more gestures in the dataset:

Even though this study introduces a self-generated new dataset with a rather more gestures for American Sign Language, it still does not offer all the possible movements for American Sign Language. Videos with rotation in 3-Dimension, words and expressions are examples of how this dataset can be extended.

8.3 Limitation

- When lighting conditions change, we need to change contour template to get the best result.
- We used rounding box for hand gesture detection. User has to put his or her hand within that box to detect hand gesture.

References:

1] WHO calls on private sector to provide affordable hearing aids in developing world. [Internet]. Current neurology and neuroscience reports. U.S. National Library of Medicine; 2001 [cited 2019Jan22]. Available from: https://www.ncbi.nlm.nih.gov/pubmed/11887302

[2] Das K, Singha J. Hand Gesture Recognition Based on Karhunen-Loeve Transform. 2013Jan17;:365–71.

[3] Aryanie D, Heryadi Y. American sign language-based finger-spelling recognition using k-Nearest Neighbors classifier. 2015 3rd International Conference on Information and Communication Technology (ICoICT). 2015;

[4] Sharma R. Recognition of Single Handed Sign Language Gestures using Contour Tracing descriptor. Proceedings of the World Congress on Engineering 2013 Vol II ,WCE 2013. 2013Jul3;.

[5] Starner T, Pentland A. Real-time American Sign Language recognition from video using hidden Markov models. Proceedings of International Symposium on Computer Vision - ISCV. :227–43.

[6] Jebali M, Dalle P, Jemni M. Extension of Hidden Markov Model for Recognizing Large Vocabulary of Sign Language. International Journal of Artificial Intelligence & Applications. 2013;4(2):35–44.

[7] Suk H-I, Sin B-K, Lee S-W. Hand gesture recognition based on dynamic Bayesian network framework. Pattern Recognition. 2010;43(9):3059–72.

[8] Mekala P, Gao Y, Fan J, Davari A. Real-time sign language recognition based on neural network architecture. 2011 IEEE 43rd Southeastern Symposium on System Theory. 2011;:14–6.

[9] Admasu YF, Raimond K. Ethiopian sign language recognition using Artificial Neural Network. 2010 10th International Conference on Intelligent Systems Design and Applications. 2010;:995–1000.

[10] Atwood J, Eicholtz M, Farrell J. American Sign Language Recognition System. Artificial Intelligence and Machine Learning for Engineering Design. Dept of Mechanical Engineering. 2012;.

[11] Pigou P. Sign Language Recognition Using Convolutional Neural Networks. European Conference on Computer Vision. 2014Sep6;

[12] Mitchell, Ross, Young, Travas, Bachleda, Bellamie, et al. "How Many People Use ASL in the United States?: Why Estimates Need Updating" (PDF). Vol. 6. Gallaudet University Press; 2012.

[13] William Vicars / ASL University. ASL [Internet]. Children of Deaf Adults (CODA). [cited 2019 Jan29]. Available from: http://www.lifeprint.com/.

[14] Home Page [Internet]. Province of Manitoba. Government of Manitoba, Water Stewardship Division, Ecological Services Division, Planning and Coordination Branch; [cited 2019Jan29]. Available from: http://www.manitoba.ca/index.html

[15] Mitra S, Acharya T. Gesture Recognition: A Survey. IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews). 2007;37(3):311–24.

[16] Nagi J, Ducatelle F, Caro GAD, Ciresan D, Meier U, Giusti A, et al. Max-pooling convolutional neural networks for vision-based hand gesture recognition. 2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA). 2011;

[17] Bedregal B, Dimuro G, Costa A. Hand Gesture Recognition in an Interval Fuzzy Approach. TEMA - Tendências em Matemática Aplicada e Computacional. 2007;8(1):21–31.

[18] Sahoo, Mishra A, S G, Ravulakollu, K K. Sign Language Recognition : State of the Art. Asian Res. State of the Art Asian Res. 9(2):116–34.

[19] Phi LT, Nguyen HD, Bui TQ, Vu TT. A glove-based gesture recognition system for Vietnamese sign language. 2015 15th International Conference on Control, Automation and Systems (ICCAS). 2015;:1555–9.

[20] Emond A, Ridd M, Sutherland H, Allsop L, Alexander A, Kyle J. The current health of the signing Deaf community in the UK compared with the general population: a cross-sectional study. BMJ Open. 2015;5(1).

[21] Bretzner L, Laptev I, Lindeberg T. Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering. Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition. 2002;:405–10.

[22] Mckenna SJ, Morrison K. A comparison of skin history and trajectory-based representation schemes for the recognition of user-specified gestures. Pattern Recognition. 2004;37(5):999–1009.

[23] Imagawa I, Matsuo H, Taniguchi R, Arita D, Lu S, Igi S. Recognition of local features for camera-based sign language recognition system. Proceedings 15th International Conference on Pattern Recognition ICPR-2000. 2000;:849–53.

[24] Dardas N, Georganas N. Real-time Hand Gesture Detection and Recognition Using Bag-of-Features and Support Vector Machine Techniques. IEEE Transactions on Instrumentation and Measurement. 2011Nov;60(11):3592–607.

[25] Zabulis X, Baltzakis H, Argyros A. Vision-based hand gesture recognition for human-computer interaction. Lawrence Erlbaum Associates, Inc; 2009.

[26] Rehg J, Kanade T. DigitEyes: vision-based hand tracking for human-computer interaction. Proceedings of 1994 IEEE Workshop on Motion of Non-rigid and Articulated Objects. :16–24

[27] Gavrila D, Davis L. 3-D model-based tracking of humans in action: a multi-view approach. Proceedings CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 1996;:73–80.

[28] Utsumi A, Ohya J. Image segmentation for human tracking using sequential-image-based hierarchical adaptation. Proceedings 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat No98CB36231). :911–6.

[29] Blake A, North B, Isard M. Learning multi-class dynamics. In Proc Advances in Neural Information Processing Systems (NIPS). 1999;11:389–95.

[30] Crowley J, Berard F, Coutaz J. Finger tracking as an input device for augmented reality. In International Workshop on Gesture and Face Recognition. 1995Jun;.

[31] Rehg J, Kanade T. Model-based tracking of self-occluding articulated objects. Proceedings of IEEE International Conference on Computer Vision. 1995;:612–7.

[32] Davis J. Visual gesture recognition. IEE Proceedings - Vision, Image, and Signal Processing. 1994;141(2):101–6.

[33] Chen Q, Georganas ND, Petriu EM. Real-time Vision-based Hand Gesture Recognition Using Haar-like Features. 2007 IEEE Instrumentation & Measurement Technology Conference IMTC 2007. 2007;

[34] Viola P, Jones M. Robust real-time face detection. Proceedings Eighth IEEE International Conference on Computer Vision ICCV 2001.

[35] Viola P, Jones MJ. Robust Real-Time Face Detection. International Journal of Computer Vision. 2004;57(2):137–54.

[36] Wu Y, Lin J, Huang T. Capturing natural hand articulation. Proceedings Eighth IEEE International Conference on Computer Vision ICCV 2001. :426–32.

[37] Shimada N, Shirai Y, Kuno Y, Miura J. Hand gesture estimation and model refinement using monocular camera-ambiguity limitation by inequality constraints. Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition. 1998;:268–73.

[38] Wu Y, Huang T. Capturing articulated human hand motion: a divide-and-conquer approach. Proceedings of the Seventh IEEE International Conference on Computer Vision. 1999;:606–11.

[39] Aran O, Keskin C, Akarun L. Computer Applications for Disabled People and Sign Language Tutoring. Proceedings of the Fifth GAP Engineering Congress. 2006Apr;:26–8.

[40] Tokatlı H, Halıcı Z. 3D Hand Tracking in Video Sequences. MSc Thesis. 2005Sep;

[41] He J, Zhang H. A Real Time Face Detection Method in Human-Machine Interaction. 2008 2nd International Conference on Bioinformatics and Biomedical Engineering. 2008;

[42] Zhu Q, Wu C-T, Cheng K-T, Wu Y-L. An adaptive skin model and its application to objectionable image filtering. Proceedings of the 12th annual ACM international conference on Multimedia - MULTIMEDIA 04. 2004;

[43] Kelly W, Donnellan A, Molloy D. Screening for Objectionable Images: A Review of Skin Detection Techniques. 2008 International Machine Vision and Image Processing Conference. 2008;:151–8.

[44] Zarit B, Super B, Quek F. Comparison of five colour models in skin pixel classification. Proceedings International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems In Conjunction with ICCV99 (Cat NoPR00378). :58–63.

[45] Ford A, Roberts A. colour space conversions. Westminster University,London,UK. 1998Aug11;

[46] Gonzalez R, Woods R, Eddins S. Digital Image Processing Using MATLAB. Englewood Cliffs, NJ. 2004;

[47] Hughes JF. Computer graphics: principles and practice. Upper Saddle River, NJ: Addison-Wesley; 2014.

[48] Nallaperumal K, Ravi S, Babu K, Selvakumar K, Fred A, Seldev C, et al. Skin detection using colour pixel classification with application to face detection: A comparative study. Proc IEEE Int Conf Comput Intell Multimedia Appllication. 3:436–41.

[49] American Sign Language [Internet]. National Institute of Deafness and Other Communication Disorders. U.S. Department of Health and Human Services; 2019 [cited 2019Apr24]. Available from: https://www.nidcd.nih.gov/health/american-sign-language

[50] Cutler R, Turk M. View-based interpretation of real-time optical flow for gesture recognition. Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition.

[51] Martin J, Devin V, Crowley J. Active hand tracking. Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition. 1998;573–8.

[52] Greenspan H, Goldberger J, Eshet I. Mixture model for face-colour modeling and segmentation. Pattern Recognition Letters. 2001;22(14):1525–36.

[53] Phung SL, Chai D, Bouzerdoum A. A universal and robust human skin colour model using neural networks. IJCNN01 International Joint Conference on Neural Networks Proceedings (Cat No01CH37222).

[54] Russell S, Norvig P. Artificial Intelligence: A Modern Approach. Computer Vision Group. 2015;

[55] Kalová I, obrazu P. . VUT Brno Scriptum, Pocitacove videni. Computer Vision Group. 2015;

[56] Knewton. Over under fitting [Internet]. blog developer blog [online]. [cited 2017May13]. Available from: https: //18784-presscdn-0-49-pagely.netdna-ssl.com/wp-content/uploads/ 2014/09/Gizem1.jpg.png

[57] Heaton J. Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning. Genetic Programming and Evolvable Machines. 2017;19(1-2):305–7.

[58] Test vs. training error. [Internet]. Stack Exange [online] . Available from: https://i.stack.imgur.com/ IpI8U.png

[59] Palm G. Warren McCulloch and Walter Pitts: A Logical Calculus of the Ideas Immanent in Nervous Activity. Brain Theory. 1986;:229–30.

[60] Minsky M, Papert S. : an introduction to computational geometry. MIT Press. 1969.

[61] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016;

[62] Cun YL, Jackel L, Boser B, Denker J, Graf H, Guyon I, et al. Handwritten digit recognition: applications of neural network chips and automatic learning. IEEE Communications Magazine. 1989;27(11):41–6.

[63] Sadhu T. Machine learning: Introduction to the artificial neural network. http://durofy.com/ machine-learning-introduction-to-the-artificial-neural-network.

[64] Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. Communications of the ACM. 2017;60(6):84–90.

[65] Karpathy K. Convolutional neural networks for visual recognition. Available from: http://cs231n.github.io/convolutional-networks

66] Dozat T. Incorporating nesterov momentum into adam. 2015;

[67] Hubel DH, Wiesel TN. Receptive fields of single neurones in the cats striate cortex. The Journal of Physiology. 1959;148(3):574–91.

[68]An image of a traffic sign is filtered by 4 5×5 convolutional kernels [Internet]. Nvidia Developer; Available from: https://devblogs.nvidia.com/parallelforall/ wp-content/uploads/2015/11/fig1.png

[69] Srivastava N, Hinton G, Krizhevsky I, Sutskever I. Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research. 2014;:1929–58.

[70] Dropout [Internet]. 2015. Available from:

http://lamda.nju.edu.cn/weixs/project/CNNTricks/imgs/ dropout.png

[71] Gibiansky A. Convolutional neural networks. 2014.

[72] Matheson G. 6 Myths About Sign Language [Internet]. Ai. [cited 2019Apr24]. Available from: https://blog.ai-media.tv/blog/6-myths-about-sign-language.

APPROVAL FOR CONDUCTING RESEARCH INVOLVING HUMAN SUBJECTS

Research Ethics Board – Laurentian University

This letter confirms that the research project identified below has successfully passed the ethics review by the Laurentian University Research Ethics Board (REB). Your ethics approval date, other milestone dates, and any special conditions for your project are indicated below.

TYPE OF APPROVAL / New X / Modifications to project / Time extension

Name of Principal Investigator and school/department	Sandipgiri Goswami, Dept. of Math and Computer Science, supervisor, Kalpdrum Passi
Title of Project	Real-time static gesture detection using machine learning
REB file number	6017156
Date of original approval of project	April 17, 2019
Date of approval of project modifications or extension *(if applicable)*	
Final/Interim report due on: *(You may request an extension)*	April 17, 2020
Conditions placed on project	

During the course of your research, no deviations from, or changes to, the protocol, recruitment or consent forms may be initiated without prior written approval from the REB. If you wish to modify your research project, please refer to the Research Ethics website to complete the appropriate REB form.

All projects must submit a report to REB at least once per year. If involvement with human participants continues for longer than one year (e.g. you have not completed the objectives of the study and have not yet terminated contact with the participants, except for feedback of final results to participants), you must request an extension using the appropriate LU REB form. In all cases, please ensure that your research complies with Tri-Council Policy Statement (TCPS). Also please quote your REB file number on all future correspondence with the REB office.

Congratulations and best wishes in conducting your research.

Rosanna Langer

Rosanna Langer, PHD, Chair, *Laurentian University Research Ethics Board*