# Real-time New Zealand sign language translator using convolution neural network.

**A thesis**

**submitted in fulfilment.**

**of the requirements for the degree**

**of**

***Master of Engineering***

**at**

**The University of Waikato**

**by**

**Mathes Kankanamge Chami Jayasekera**

THE UNIVERSITY OF
WAIKATO
*Te Whare Wānanga o Waikato*

**2021**

# Abstract

Over the past quarter of a century, machine Learning performs an essential role in information technology revolution. From predictive internet web browsing to autonomous vehicles; machine learning has become the heart of all intelligence applications in service today. Image classification through gesture recognition is sub field which has benefited immensely from the existence of this machine learning method. In particular, a subset of Machine Learning known as deep learning has exhibited impressive performance in this regard while outperforming other conventional approaches such as image processing. The advanced Deep Learning architectures come with artificial neural networks particularly convolution neural networks (CNN). Deep Learning has dominated the field of computer vision since 2012; however, a general criticism of this deep learning method is its dependence on large datasets. In order to overcome this criticism, research focusing on discovering data- efficient deep learning methods have been carried out. The foremost finding of the data-efficient deep learning function is a transfer learning technique, which is basically carried out with pre-trained networks. In this research, the InceptionV3 pre-trained model has been used to perform the transfer learning method in a convolution neural network to implement New Zealand sign language translator in real-time.

The focus of this research is to introduce a vision-based application that offers New Zealand sign language translation into text format by recognizing sign gestures to overcome the communication barriers between the deaf community and hearing-unimpaired community in New Zealand. As a byproduct of this research work, a new dataset for New Zealand sign Language alphabet has been created. After training the pre-trained InceptionV3 network with this captured dataset, a prototype for this New Zealand sign language translating system has been created.

# Acknowledgements

This research would not have been completed without the exceptional guidance and support given by my research supervisor Dr. Chi Kit Au. His passion, exacting attention and knowledge have been a motivation and it kept my research work on track from the beginning to the final draft of this thesis. He continuously guided me by giving advice to optimize the research and deliver the best results. I am also grateful to Cheryl Ward, the librarian of the sciences of University of Waikato  for sharing invaluable information about the literature resources and thesis formatting.

It is with great pleasure that I acknowledge the support and the contribution of my amazing spouse who has given up everything and moved to New Zealand so I could pursue my higher studies. A special thanks goes to my family, especially my father for supporting me financially and giving me the strength to achieve my academic goals.

# Declarations

I certify that this master thesis report does not incorporate without acknowledgement, any material previously submitted for a degree or a diploma in any university, and to the best of my knowledge and belief it does not contain any material previously published or written by another person, except where due reference is made in the text.

Signed:Chami

February 11, 2021

# Table of Contents

# Abbreviations

| | |
|---|---|
| SDK | -software development Kit |
| API | - Application programming Interface |
| NSL | – New Zealand Sign Language |
| ASL | – American Sign Language |
| TSL | - Turkish sign language |
| BSL | -British sign Language |
| PSL | -Pakistani Sign Language |
| ISL | - Indian sign Language |
| CNN | – Convolution Neural Network |
| ANN | _ Artificial Neural Network |
| AI | – Artificial Intelligence |
| ReLU | - Rectified Linear Unit |
| GPU | - Generally Graphical Processing Unit |
| CPU | – Central Processing Unit |
| Relu | - Rectified Linear Unit |
| LC | – Learning Curve |

# List of Figures

# List of Tables

# Chapter 1 : **Introduction**

"Good communication is the bridge between confusion and clarity."
Nat Turner
A Black American Hero

Communication is considered as the core of interaction of humans. Therefore, language can be the most effective communication method for humans. However, this effective method can be insufficient as sometimes it creates barriers due differing the languages. In addition, different modalities of language, for example sign Language and spoken Language create communication barriers. As per the introductory quote, clear and decent communication can enhance clarity and the effectiveness. On the contrary ineffective communication may lead to community breakdown, and lack of intimacy and understanding. In this chapter, background to the research problem, the importance of a sign language research gap and research objectives will be discussed.

## 1.1 Background

Many researchers from different part of the world have identified a criteria list for defining the word, "language" in contrast to another kinds of systems in communication such as Morse code. This specific criterion for defining that word "language" should be observed as a collective feature which have been identified as sufficient and necessary in order to be classified in the natural language category. Since mid-20th century, many researchers in linguistic field have tried to define the word, "Language" and this has allowed them to go beyond from traditional definition of the word, "language" and to reconsider other features which were once considered vital to the definition. While language normally commences as speech, which depends on the vocal - auditory channel [1], language can be also expressed a visual-gestural mode. This is basically fulfilled by recognizing human gestures and identifying them as symbols or signs. Deaf people make use of hand signs in order to communicate, so hearing-unimpaired people of the community encounter difficulties in recognizing the signs made by the deaf person as language. On the other hand, deaf people also undergo the same problem when they use their sign language to pass on a message to a hearing-unimpaired person. Therefore, it is very important to build up a system which can allow both deaf people and hearing-unimpaired people to communicate with each other. Sign Language is not a universal Language as it differs from country to country and region to region and one gesture sign can build up different meanings in different places of the world. American Sign Language (ASL), Turkish sign language (TSL), British sign Language (BSL), Pakistani Sign Language (PSL), Indian sign Language(ISL), and New Zealand sign Language(NZSL) are some of the available sign languages in the world.

## 1.2 New Zealand Sign Language

New Zealand sign language (NZSL) is specially designed for the deaf community living in New Zealand. This NZSL is very important to countless deaf new Zealanders, allowing them to communicate, learn and take part in modern society. This language is crucial to the representation of deaf identity and culture. Culture of the deaf people is well documented and contains shared values, behaviors, norms, humor, history, stories, arts, tradition, and poetry. Generally deaf culture is passed on from one generation to another generation. New Zealand sign Language is considered to one of the three official New Zealand languages, along with English and Te Reo Maori. This NZSL contains grammatical structure and features that permit signers to express emotions and thoughts, and communicate fully. However, New Zealand sign language (NZSL) diverges from audio/ spoken languages as it is uniquely visual. There are around 11,000 deaf New Zealanders who utilize NZSL as the primary method of communication and around 20,000 New Zealanders in total who use NZSL. Signers include family members who use this language to communicate with a deaf family member [2].Even though It has been used for over hundreds of years, it was only declared an official language by the government of New Zealand through their “New Zealand Sign Language Act” in 2006 [3]. This recognition was the key forward step in improving Deaf New Zealanders’ lives.

The majority of users of the deaf New Zealand sign language are deaf either birth or early childhood. Approximately 5% of the parents of the deaf are also deaf and those 5% acquire sign language as their native language from birth. Majority of the deaf children come from the families where there no other deaf people in their family tree, so the family has neither prior experience of using a sign language or no link to the deaf or deaf community. However, families have no choice: they have to learn sign language to communicate with their deaf love ones. Irrespective of requirement or choice, some hearing-unimpaired people and deaf children learn sign language later. This enhances the usage of sign language in a country. [4]Dr. Marianne Collins was the first to document the lexicon and the grammar of the New Zealand sign language and from her documentation, the abbreviation, NZSL and the name itself appeared in 1980s. This led to the establishment of a Research unit for deaf studies at Victoria University, Wellington, New Zealand. This unit engaged with the Deaf Association in order to produce a New Zealand Sign language research-centered dictionary. In 1985, first training for NZSL interpreters were able boost NZSL awareness, and later, an American sign language interpreter, Dan Levitt, engaged with the deaf community and guided the students to interpret first New Zealand sign Language dictionary [4].

New Zealand sign Language has some roots with British Sign Language (BSL) and Australian Sign Language (ASL), and so may be theoretically considered as "British, Australian and New Zealand Sign Language (BANSL). There is a 62.5% of match between New Zealand sign language and British Sign language (BSL); while 33% in New Zealand sign language (NZSL) matches American Sign language (ASL). New Zealand sign language uses a two-handed manual alphabet as does British Sign Language and Australian sign language. Sign languages like American sign language, Indian sign language on the other hand use one-handed manual alphabets [5]. New Zealand sign Language, like other numerous sign languages, is expressed with the help of hands, face, torso, eyes and the shoulders of the person using the sign language. Sign language is demonstrated in 3-dimensional space called the 'signing-space' as shown in the Fig 1.1.

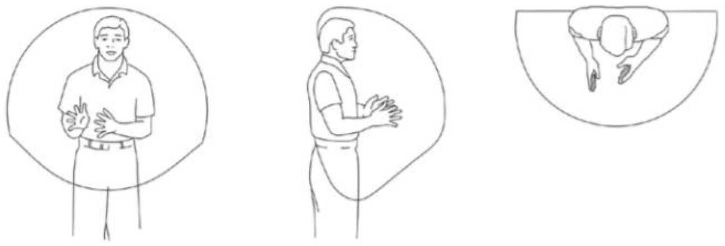


**Figure 1.1: Signing space.**

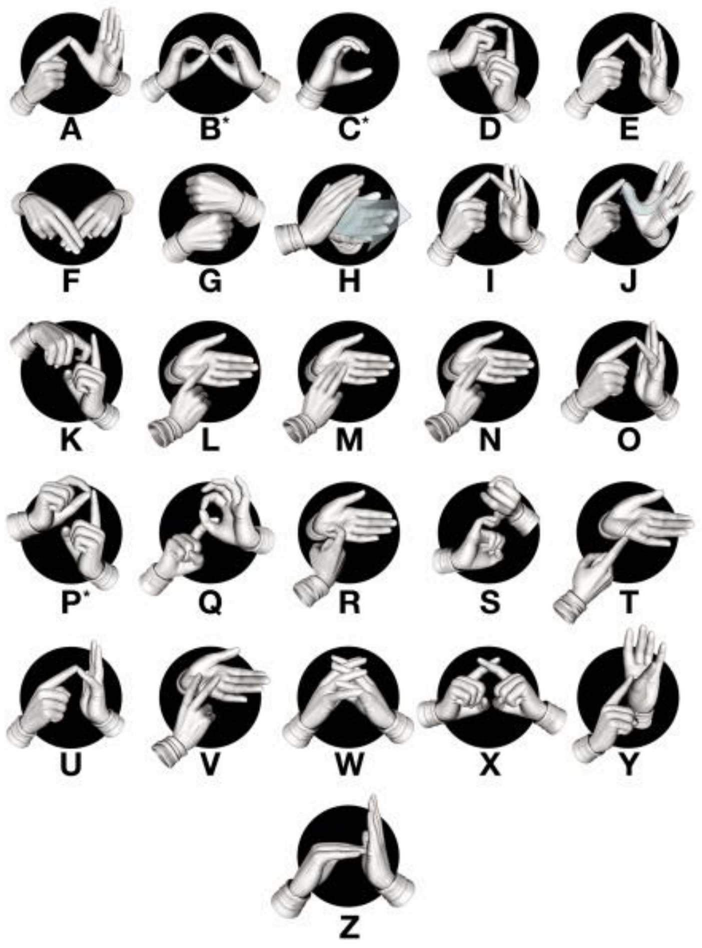


**Figure 1.2: New Zealand Sign Language Alphabet.AS SEEN BY THE VIEWER. [6]**

In New Zealand Sign Language, signs can be made with either the right or left hand and either one can be the dominant hand. Additionally, the roles of each does not switch usually when doing the fingerspelling. Whatever the dominant hand, it functions as the pointer or pen while the non-dominant hand function as the paper. In New Zealand sign language, the vowels in the English language, A, E, I, O, U are signed by touching digits individually starting with the thumb. The letters C, D, J, K, P, Q, T, V, X, Y respectively are formed from the letter shape. Further, the letters B, F, G, L, M, N, R, S, W also make the letter's shape. Letters G, L, R form only the letter's lowercase. The letters 'H' and 'Z' do not come with any particular relationship to either its shape or position in English alphabet. A few letters, such as R and D appear backwards to the audience.

Recognizing of hand gestures provides a natural, convenient, and intelligent approach for human-computer interaction. Human gesture-based controlling systems and sign language identification are two key applications for the technologies of hand gesture recognition. Sign Language recognition systems are aimed at interpreting the trained sign languages by a computer program automatically to help and guide the deaf community to communicate with hearing society effectively. As sign language is an extremely structured and highly symbolic human gesture pack, Sign Language recognition serves as the basic for gesture-based human-computer interaction. [7].

## 1.3 Research Gap

At present, much research has been conducted to implement systems for sign language translation adhering to human hand gestures. However, they all have been conducted only for one-handed sign languages (which use only one hand when performing hand signs), including American Sign Language (ASL), Chinese Sign Language (CSL), French Sign Language (LSF), Japanese sign Language (JSL), Indian sign Language (ISL). A few countries such as United Kingdom, Australia and New Zealand share Dual-handed sign language called BANZSL (which uses both hands when performing hand signs). Unlike American sign language, dual-handed sign languages use both dominant and non-dominant hands to represent their signs [8]. Performing signs in dual-handed sign language is a difficult task compared to single handed sign language as performing dual-handed signs requires complex hand movements & shapes and less widespread. Therefore, designing a dual-handed sign language system for dual-handed sign language is much more complicated. There are no resources verifying that the New Zealand sign language has gotten through any structured and accepted gesture-based sign language identifying system [9].

Sign language translation was initiated with the development of the robotic communication. In 1977, the first finger spelling robotic hand project known as RALPH ("Robotic Alphabet") invented a robotic hand that was capable of translating alphabets automatically into certain finger-spellings [10]. Later on, data transferable gloves along with motion sensors turned out to be the mainstream, and other projects like VPL data transferable glove and CyberGlove were introduced [11]. Even today, most researchers use a glove-based data collection system to translate the sign languages into spoken language or written language. Nevertheless, growth of computer vision has allowed cameras to substitute for those wearable instruments due to high efficiency and lower physical restrictions for signers. Little research has been conducted to implement hybrid systems by combining non-wearable technologies using Leap motion device controllers and cameras. This method has been able to improve the automatic sign language detection ability and translate them into spoken language and written language.

The novelty of the research of this thesis is to study how far it will be successful to implement an artificial intelligence-based computer program to identify dual-handed sign languages especially New Zealand sign language and interpret them into text mode or written format. To validate this study, a New Zealand sign language interpreting system based on human hand signs will be implemented. As a consequence, the deaf community in New Zealand will be able to communicate with the hearing-unimpaired community who are not familiar with New Zealand sign language. It is expected to be a practical solution to the language barrier between deaf community and the hearing-unimpaired civilians.

## 1.4 Research Objectives

The aim of this study is to find out how successful is translating New Zealand sign language into text format by training Convolution Neural Network (CNN) with human hand gestures. As this task is considered to be broad in scope, this study has been limited to training the New Zealand sign language alphabet and some other basic symbols. This research study is conducted to bridge the communication gap between the hearing community and the deaf community with the support of new technological endeavors. This system is designed to translate the New Zealand sign language in real time while adhering to the movements of the human hand.

The Objectives of this study are:

1. To study effective gesture communication with deaf people in New Zealand by devising a technique that converts the hand gestures of New Zealand sign Language (NZSL) into Digital Text format.
2. To analyze the performance of the deep neural network, Convolutional Neural Network (CNN) and then transcribe the data gathered from human hand gestures into Convolutional Neural Network (CNN).
3. To obtain an overall training accuracy of 90% or above and higher and reliable performance metrics values in order to validate the method while conducting the study in natural environment without artificial setups.

4. To develop an unobtrusive conversion system for both the audience and the deaf signer.
5. To discuss the results and propose recommendations and openings for future improvements in telecommunication technologies.

This proposed New Zealand Sign Language translation system based on human hand gestures is an ongoing topic amongst Artificial Intelligence researchers. This study will prove how much deaf community can communicate with hearing-unimpaired people using an artificial intelligence technique called transfer learning. This idea can be improved upon by introducing a phone application that is able to recognize and translate different kinds of sign languages into text mode and ultimately into voice mode. This can be developed as a vice versa version: After inputting a text or voice message, this system can produce relevant hand gestures according to that text or voice message. A web application can be developed to produce hand gestures according to input voice. Then deaf people would be able to understand the content of any video in Websites, on social media etc.

The next chapter, (chapter 2) will introduce relevant published literature. Chapter 3 will discuss the methodology of the research. Chapter 4 will review the system design and determine the work needed to achieve the research goal. Chapter 5 will display the obtained results, and discussion of these will be in the chapter 6. The last chapter, (chapter 7) will the conclusion of the research and include future suggestions for improving the designed system.

# Chapter 2 : **Literature Review**

Human communication evolved over the time, and has involved pictograms, petroglyphs, alphabet, ideograms, signals, sounds, and gestures as aspect of communication. At present, the dominant communication method depends on alphabet expression as orally by writing or using sign language. The people who are suffering from either speaking or listening disorders (or both), are not able to communicate with each other orally [12]. The deaf are defined as people who do not have functional hearing for purposes of communication in their day to day lives. However, there are different categories of deafness: the congenially deaf who have not had functioning hearing from their birth, and the adventitious deaf, who have not been deaf from birth but became deaf later due to various causes [13].

According to the statistics provided by the World Health Organization, there are approximately 450million people worldwide with some sorts of a disabling hearing loss which is nearly 5% of the world's population. They expect the numbers will rise to around 900 million by the end of 2050. Their focus is on prevention and protection methods to increase awareness and to enhance the public health measures [14]. In New Zealand alone, according to the New Zealand Sign Language Board, there are roughly 11000 deaf persons [15].

There are many consequences that needs to be handled for deaf people to ensure their environment is safe and normal. The communication barrier affecting healthcare and in public is one of the most important issues that deaf people have to face [16]. The main method of communication for the deaf is the sign language, and thus it plays a major role removing the barrier to communicating. Sign language consists of movements of the upper body including the head and a unique fingerspelled alphabet for nouns. A portion of the community who do not have hearing problems will also need to be educated in sign language in order to communicate with the deaf. Sign Language-based education, and the development of sign languages gives the opportunity for all the deaf people who were left isolated to be educated. Eventually, deaf people have been trained to take the role of teachers for deaf children as well. Nowadays, sign language is considered as one of the musts by the responsible authorities in educational platforms and is essential for the development of the language usage worldwide [17]. New Zealand sign Language (NZSL) has been made as an official language by the New Zealand Sign Language Act 2006. In spite of that difficulties such as lack of services for deaf community in most public places and the lack of opportunities in education should also be taken into consideration worldwide. Advancements made in New Zealand for deaf education is a prime example of advancing the empowerment progress. All these steps and improvements need to be made with a clear view of how they affect the mental health of the deaf community (defined as the impacts of hearing loss socially and emotionally). The economic impacts also considered with rising unemployment rates of deaf people; however, improvements in education and vocational training facilities are expecting to decrease the unemployment rates [14].

## 2.1 Introduction to Sign Languages.

Sign language is a rich one consisting all the key features of a language; it can also be viewed as the most natural languages among all the languages in use. Literature since the early 16th century suggests there has been interest shown by humans in various levels of sign language usage as a visual communication method. Language has had considerable community exposure with ups and downs and went through a systematic approach in 17th,18th. After adhering of a curriculum to the language in the late 18th and early 19th centuries, Language proclaimed where it is now [18]. However, there is no universal sign language; differs from spoken language and has regional variation.

Researchers began linguistic studies on sign language in the 1970s [19]. It has been shown to contain lingual information including different letters and symbols. Symbols of sign language are capable of indicating all of the sign parameters and involves hand shapes, location, movements and orientation of the palm. Fig 2.1 indicates the classification of sign language symbols. The principal classification is into single handed sign languages and double handed sign languages. These are divided into static signs and dynamic signs [20].

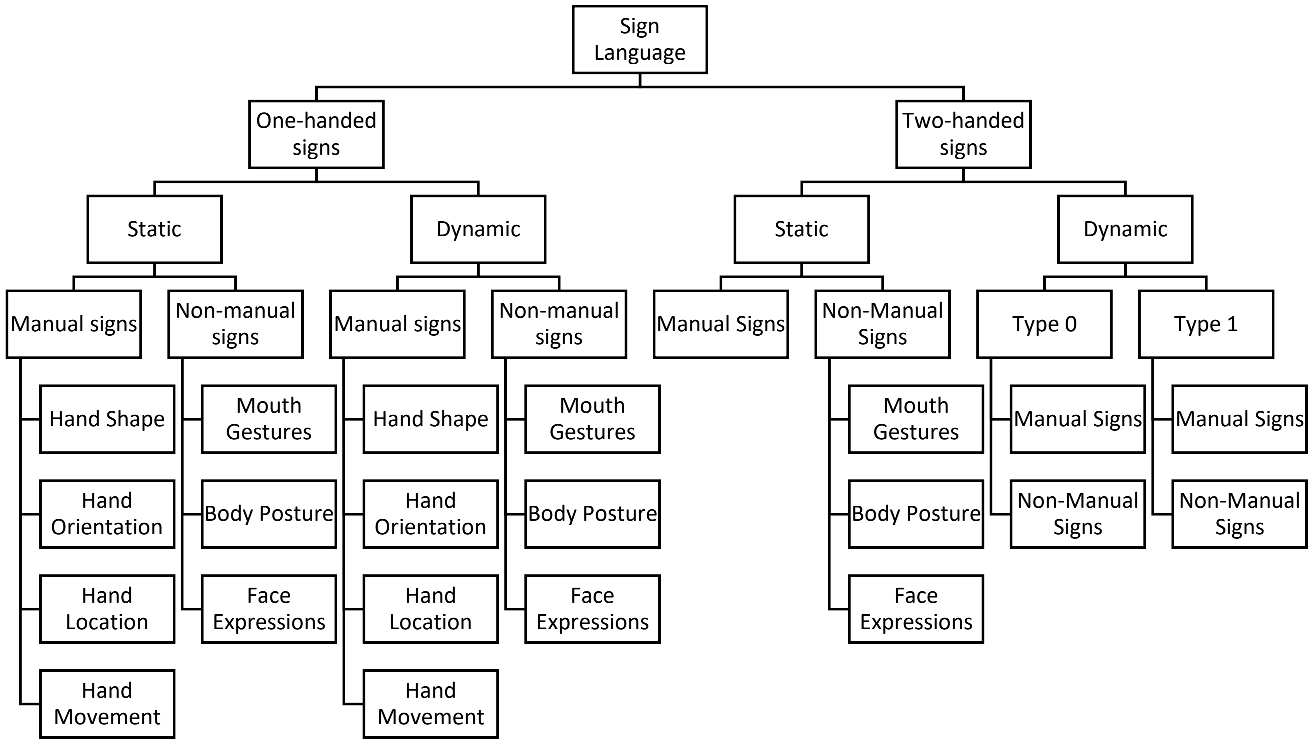


**Figure 2.1: Sign Language Hierarchy. [20].**

In order to represent one-handed signs, only one hand is used. It can be either a static gesture or a motion gesture. For dual-handed signs, both of the dominant hand and non-dominant hand are used when signing. As examples, American sign language, Indian sign language, Pakistan sign languages are categorized into one-handed sign languages, while Sign languages like New Zealand sign language and British sign language are classified as dual-handed sign languages. Dual-handed sign language can be further classified into type 1 and type 0 signs. As shown in Fig 2.2, both dominant hand and non-dominant hand are active when signing, whereas as per the Fig 2.3 the dominant hand can be more active when compared to the non-dominant hand in type 1 signing. Sign languages contain manual elements but also non-manual elements as well [21]. In manual hand signs, only hands are used to convey any kind of sign and in non-manual sign language, mouth gestures, face expressions and body postures are used. In Fig 2.4 and Fig 2.5 one-handed static manual signs and single-handed static non-manual signs are indicated, respectively.

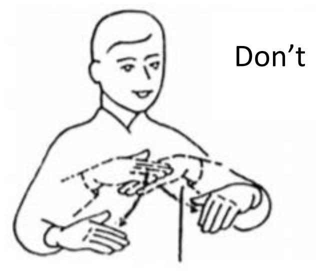


**Figure 2.2: Type 0, dual-handed sign (both dominant and non-dominant hands are active) [20].**

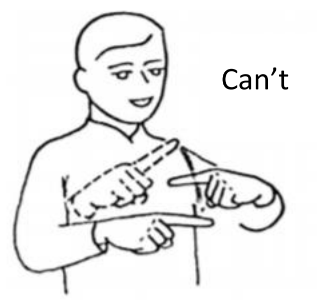


**Figure 2.3:Type 1,dual-handed sign (only the dominant hand is in active stage) [20].**

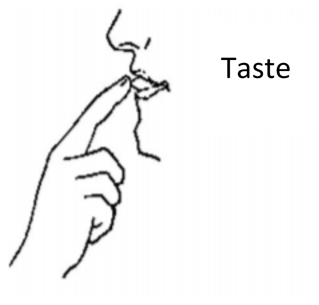


**Figure 2.4: One-handed, Static, Manual sign. [22]**

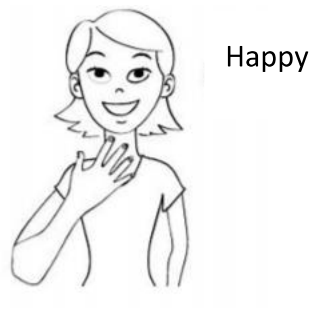


**Figure 2.5:One-handed, Static, Non-manual sign. [22]**

## 2.2 Introduction to Gesture Recognition

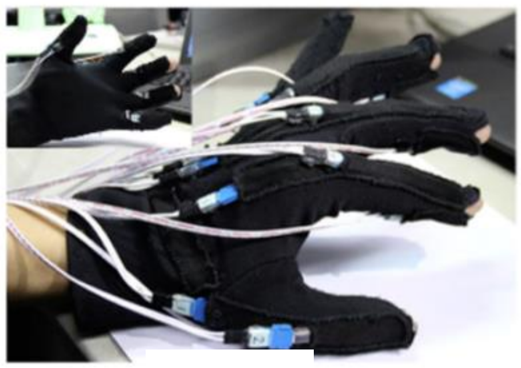
Even though hearing-impaired persons master sign language, few hearing-unimpaired persons understand and are able to utilize sign language. This has affected communication with people who are deaf and results in a sort of isolation between hearing-unimpaired people and the deaf. This gap can be reduced by utilizing a system which allows the sign language translation automatically into text mode, and vice versa. Numerous recent paradigm shifts in several technology fields are helping researchers to suggest and implement various systems using sign language recognition based on human gestures [23], [24], [25], [26], [27], [28]. Thus, several sign language recognition systems have been proposed for various sign languages, including American Sign Language, Korean Sign Language, and Chinese Sign Language [29] . The proposed sign recognition systems rely on either image-based or sensor-based solutions.

The importance of and the requirement for gesture recognition has been proposed by Myron W. Kruger in the 1970s to ensure a human-computer interaction and interpretation was more systematic [18] .Especially in an age when the computer field is rapidly changing, human-computer interaction and interpretation automatically has become an interesting and fast-escalating area of research interest. Furthermore, related works that made a difference in these systems are: pattern recognition technique-based methods to recognize hand gestures have been introduced by Freeman and Roth; a static hand gesture recognition system based on a vector system which would be effective even with complex backgrounds by Naidoo and Glaser; and finally, Triesch and Malsburg's method of recognition, which is based on user-independent recognition without hand segmentations, has gone on to achieve around 86% accuracy in complex backgrounds [30]. All above work can be categorized into the approaches mentioned below.

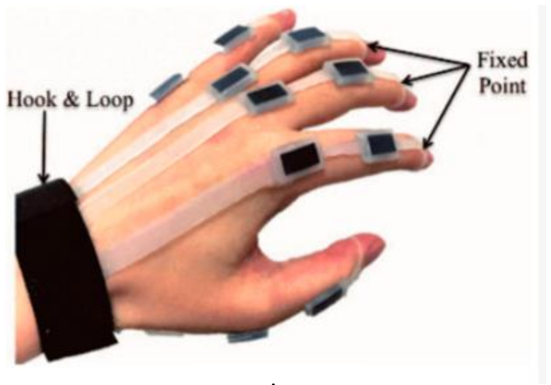
Two main approaches have been named in sign language recognition: Sensor-based and vision-based. A combination of both is also used as a different approach in modern-day research. Motive for using these approaches, in general, is to avoid communication difficulties caused to people who can talk and hear properly and for deaf people [31]. In all those mentioned approaches, five key elements are combined as elementary units, namely articulation point, hands configuration, movements type, hands orientation, movements type, hands orientation and facial expressions. These are essential in developing intelligent systems for sign language recognition [32].

## 2.3 Sensor-based Approach

Sensor-based approach is a platform created to collect data using a variety of sensors. The procedure behind this approach is firstly to grab the data using a specifically equipped with sensors, secondly to process the signals of that sensors to a pre-determined level and thirdly to produce an output which should be an identified gesture. According to the orientation determined for data gathering, sensor types such as bend, accelerometer, proximity are commonly used in gloves that are commonly available. Importantly, understanding of the degrees of freedom in movement of the glove increases with the number of embedded sensors and the type of the sensors, which ensures extracting more accurate data [32]. Gloves with data collection capabilities were invented back in 1977 [31]. High accuracy in understanding fingerspelling and in sign motions, direct recording of data without any additional tracking can be seen as most important advantages of this approach; the downsides of a sensor-based approach are the inability of performing complex gestures and its high cost [33]. This approach has been developed, tested, and even made commercially available. Some examples relevant to sensor-based approach have been displayed in the Fig 2.6 below.



a



b

**Figure 2.6:Data glove examples for a Sensor-based gesture recognition system. (a) Data glove presentation of Zheng [34] (b) Soft rubber data glove. [35]**

## 2.4 Vision-based Approach

Vision-based approach is considered to be the more convenient way for modern sensor recognition systems. Basically, this method has reduced costs in manufacturing sensor heavy gloves and adding to that, it is more geared towards optimization that should be done in optimized tracking, and in processing of the tracking data. The accuracy of this approach might be not as high as the sensor-based approach, but it is more flexible and adaptive to the data acquired with the framework. Strategies such as "*utilization hand crafted shading gloves and in light of skin-color recognition*" are identified with the vision-based approach according to the situation [33]. The primary tool for this vision-based approach is the camera as the data input gesture recognition device. The system could be a single camera such as a webcam, video camera or a smartphone camera; stereo camera setup to provide depth measuring information; structured light projection methods to capture active motions and the inclusion of invasive techniques such as body marking, LED light systems, wrist bands etc. [36]. The process includes machine learning stages, motion modeling and analysis alongside pattern recognition that are computer-performance based, and the accuracy of the sign language recognition depends on these features. It should be mentioned that most of the recognition processes have an accuracy more than 90% [32].

Studying these main approaches for interpretation has suggested that a unique approach should be determined for the purpose according to the degree of difficulty of the language that needs to be interpreted. In addition, gestures and the word combining which are different from one language to another may increase the time needed to classify the key language elements. Comparing all the attributes of those approaches, with the accuracy, with the less environmental concerns in recognition the vision-based approach is more effective even though it has the expansiveness and the complexity behind it.

The Fig 2.7 indicates the block diagram for vison based human hand gesture recognition procedure.

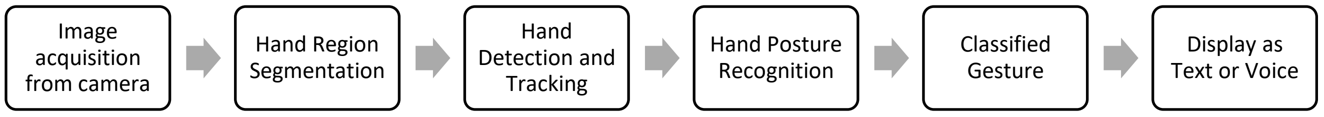


**Figure 2.7:Block Diagram for vision-based human hand gesture recognition system. [37]**

## 2.5 Classification of Hand Gestures

Even though the process is complex in gesture recognition systems, the main logic behind it depends on the stage of identification in tracking where the trajectory is a predefined gesture which has been defined earlier with vocabulary that have been officially accepted or as a random hand movement. Technically, with various optimizations have been used, such as neural networks, support vector machines, Hidden Markov models have been used for classifications of hand gestures.

Furthermore, while investigating the steps in hand gesture recognition systems, it has shown that there are four main steps that is being followed: preprocessing with hand segmentation; hand detection and tracking; hand posture recognition; and then classification of the hand gestures as the final step. The classification process is based on these steps and systems of different caliber will be using different sorts of techniques to adhere with those main steps [38].

There are two main groups of gestures, which consist of six dynamic gestures, defined in the gesture vocabulary, namely control and navigation. Difference between the two gesture groups depends on time taken to perform the gesture. Normally, a control gesture should take around 2 seconds and a navigation gesture around 1 second. The tricky part of this process of gesture recognition is determining the start and the end points of each gesture [39].

Two gesture recognition classifiers should be mentioned as examples because of their contribution towards the gesture vocabulary. These are the K-means rule based classifier and Longest Common Substring (LCS) classifier. K- means rule based classifiers is a system consisting of two stages: one depends on the aspect ratio and the standard deviation of the available space in between the trajectory boundary to its center of gravity; and in the other stage, filters recognize control or navigation group gestures using a sequence of rule filters. Some gestures in the gesture vocabulary are not consistently or precisely detected in this classifier, so an improved version of the system, named Longest common subsequence method has been invented. Longest Common Subsequence (LCS) classifier method is governed by dynamic programming principles which ensure the precise angle of the structure of motion vectors and the trajectory. A comparison of the angle of the structure or in other words, the structure of the motion vectors output with the pre-defined model patterns of the gesture vocabulary seeks the desired output. Alternative starting points and user variations could affect the results which are almost eliminated by the use of more than one model [39].

## 2.6 Applications of Gesture Recognition

Gesture recognition and their applications can be seen in various fields with many specifications attached to them. Considering only the medical system and the assistive technologies attached to that sector, there are some major implementations that have been generated benefits, such as resource distribution, medical instrument interaction and some therapy stages rehabilitation. Regarding the systems used, a common factor that can be highlighted is that all the manufacturers or researchers are eager to add components that the satisfy “come as you are”, “comfort” and “intuitiveness” requirements, and they are enthusiastic about improving user adaptability and feedback. Specific examples come from the medical industry lights in operation theaters which can be controlled using hand gestures, laparoscopes that can be controlled with facial and hand gestures, pointer movements together with button pressers which can be performed with hand gestures all help surgeons. Entertainment based technology development such as video games have also had a hype in recent times with the implementation of gesture controls. For instance, gesture recognition techniques have improved sport-based games by allowing subtle changes in movements to be tracked. Augmented reality-based technology has also evolved alongside with gesture recognition in this field.

Another main field that has focused on gesture recognition systems is crisis management and disaster relief. User to user communication is enhanced through this application which helps in collaborative decision making between individuals without having to learn new terms [40]. The automobile industry has also started researching and implementing Advanced AI systems at different levels by detection through image acquisition systems, processing it through a reference gesture set, and then to send a command to them. However, that is only the basic stage where few functions have gone through to the market; there needs to be continuous research and development [41]. Finally, as one of the applications of gesture recognition, the trending topic "Human- Robot interaction" should be mentioned as well. Hand gesture recognition is an important phase of robotic interaction nowadays, and in combination with voice command exaggerates gesture spotting, navigational tasks, and common commands. Importantly, through these uses of gesture recognition, usability will be enhanced and also the ability of data validation would also improve in a more reliable manner. These will encourage various fields to invest in research on gesture recognition schemes. [40].

## 2.7 Sign Languages by Regions and Interpretations

New Zealand is a country where 20000 to 30000 people either directly or indirectly benefit through the use of sign language. Since the New Zealand Sign Language (NZSL) is considered as one of the three main languages, many patterns in teaching and socializing the language have been implemented throughout New Zealand. Many technology related programs have been instigated all over the country, such as online teaching, video coaching, all of which can be taken as an all-round plus [9]. Even though there is literature on initializing some segmentation using the available data with the NZSL, there are no such resources confirming that the exact language has gone through a structured and approved protocol into a sign language recognition system [9].

From an examination of Indian Sign Language (ISL), it can be identified that there are slight similarities with the British Sign Language. Generally, ISL has its impact in the South Asia region since it is the primarily used language there. Reportedly there are around 1.8 million deaf and mute people in India who can be considered as the most needed set of people to the ISL to be developed and socialized more than it is. As standout completed work under the ISL, Kamal Preet Kour and Dr. Lini Mathew's system has helped in recognizing the 26 English alphabet letters in ISL with feature extraction using SURF algorithm and feature matching using Minimum Euclidean Distance Classifier. Omkar Vedak,  Parsad Zavare, Abhijeet Todkar  and Manoj Patil's system has showed a 88% accuracy with the English alphabet in ISL which was using Support Vector Machine

(SVM). G. Anantha Rao, K. Syamala, P. V. V. Kishore, A. S. C. S. Sastry's Convolutional Neural Network (CNN) based image processing mechanism where more than 200 Indian Signs have been detected with the accuracy around 93%. Researchers have developed methods of recognizing different variants of ISL, which is a positive sign for the future of the sign language and for all people who are willing to familiarize themselves with this method of communication [42].

American Sign Language (ASL) have more than 500000 beneficiaries across America and Canada. As a language that has a one-handed alphabet, it is developing into the fourth most commonly used language across the country. Its history dates back to 1817. ASL has 18-19 hand shape signs, 24 movements and 12 locations which have slight variations with race, ethnicity and various gender and age considerations. Most importantly, ASL maintains a structure which consists of three sentence structures: namely question, command and as declarative sentences with a rich grammar orientation as well [43]. There have been many systems implemented using template matching technique, hand segmentation and with boundary tracing etc. to interpret the American Sign Language more precisely and to automate the process of sign recognition to the betterment of the society [44].

## 2.8 Understanding Sign Language Databases

From regional sign languages learning resources, there have been databases developed as projects which have a direct impact on further research work and on the implementation of new techniques into the recognition process [45]. For ASL referencing, RWTH-BOSTON-50 and RWTH-BOSTON-104 databases created at Boston University can be considered as the key libraries. RWTH-BOSTON-50 consists 483 expressions of 50 signs and RWTH-BOSTON-104 consists 201 sentences with 104 different signs. German Sign Language (DGS) can be referred to using RWTH German Fingerspelling Database that contains 35 signs in video sequences and with SIGNUM Database which contains more than 450 isolated signs and sign sequences used by different people. ECHO database stands for Dutch, English, and Swedish language variations [46]. Italy has also developed their own databases with video sequences such as large-scale ATLAS project which has a tri- lingual structure containing Italian text, AEWLIS sequence and LIS corpora-based videos. It has more than a thousand of signs executed and the DIZLIS project [44].

## 2.9 Gesture recognition through Convolution neural network (CNN)

Gesture recognition can be regarded as one of the bridges to human- computer interaction. This important tool of communication using the hands has had an enormous impact on machine learning or to human-machine interaction with image and video processing techniques [47]. As mentioned earlier in this literature review, among the vision-based and sensor-based approaches in gesture recognition, the more preferred version should be much more compatible and capable of real time interactions which brings vision-based techniques into play. However, for a system which is heavily based on vision-based techniques; independent robustness, efficiency shown in computational aspect, error tolerance and scalability should be continuously researched and developed [48]. To progress the recognition part in machine learning, it is important to get to know about the approaches such as the Kinematic Model which consists kinematic constraints that use a 3D model of a hand through a 2D. The model-based approaches have disadvantage of not having a reliable edge on the captures unless they captured against a high contrast background [49]. Hand gesture classification into Rule based approaches and Machine learning based approaches have opened more research opportunities and have gained more attention as well. In Machine learning techniques consistency, fault tolerances, communication, storage, resource management and the programming model have a huge impact and the efficiency, because the effectiveness depends on these parameters [50].

Artificial Intelligence (AI), which has garnered the highest research interest in recent years, has also been taken into consideration in gesture recognition with applications related to human-computer interaction. Developments in accurate algorithms to extract any form of gesture which initiates from face or hand have been considered in AI based systems. As mentioned earlier in this literature review, hand gesture recognition techniques in a Vision-based approach with its extraction method and image preprocessing will have to go through more intense segmentation initially. Then it moves on towards gesture classification process where grouping would be done and in that phase neural networking comes into play [51].

### 2.9.1 Artificial Neural Network (ANN):

An artificial Neural Network is a computer system designed to simulate and analyze the way that the brain of the human being processes and analyzes information. It is the basis of the Artificial

Intelligence (AI). It can be a solution for solving problems which would prove difficult or impossible to solve by a human. An artificial Neural Network has self-learning abilities that make them to deliver a better result output [52]. An artificial Neural Network is generally defined by three kinds of parameters.

- The interconnection pattern among the distinct layers of neurons.
- The learning procedure for updating interconnections weights.
- The activation function which converts the weighted input of a neuron to output activation.

A neural network system is generally represented as a fully connected forward network where every neuron in a relevant layer is linked to all neurons in the next layer, as demonstrated in Fig 3.2. The connections among neurons are weighted, and the network ‘s learning process modifies each weight by feeding the collected training data into that network. Every training operation is been conducted while calculating the output error along with slightly adjusting the relevant weights to decrease the error in the following iteration. Every component in the training dataset is used numerous times to conduct the training and obtain convergence.

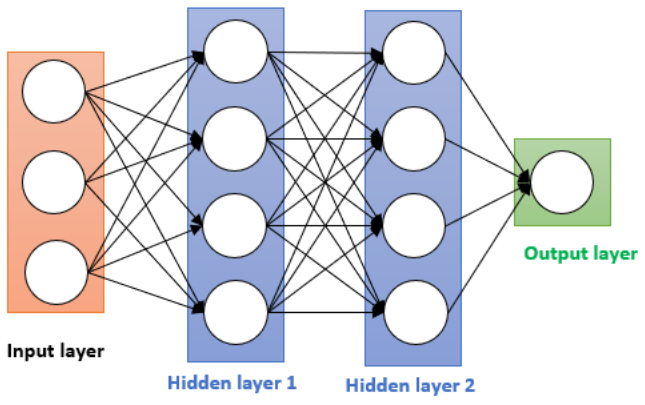


**Figure 2.8: Example for a three-layer Neural network. (University., 2016).**

Fig 2.8 demonstrates a neural network with four layers and three input, two hidden layers including four neurons in each, and one output layer. From the figure it is clear that the synapses or the connections only occur between neurons across layers not with in layers [53].

### 2.9.2 Convolutional Neural Networking (CNN):

In late 1980s Neural Networking has gained some interest in research regarding deep learning which has so many errors to start with, has obtained an evolutionary advance with Convolutional Neural Networking (CNN) which is a concept architecture created by LeCun and Bengio [50].Convolutional Neural Networking (CNN) is a class of deep learning neural networking where extraction and layer connection is used for classification as the major steps. This enhances the performance by reducing the memory requirements and the computational complexity. It can also be highlighted as one of the data-driven methodologies that has a data augmentation method which supports in deep learning [54]. The Deep Learning based processes of identification in CNN have two main steps to be completed namely, the convolutional layer which the input image, is separated into many images as a matrix, then a set of neurons consisting the split images independently connected to a filter starts the pooling process. The pooling process is responsible to perform simplification via reducing the size to ensure that the output will be a fully connected layer [55]. Several CNN platforms need to be architecturally built up to even predict a hand's joint location in a 3D plan [56]. The bottleneck features or the bottleneck layer provides the platform with more subtle retaining information which is useful in the process of classification. This also helps the network growth and protects the system from tolerances. The error rate is reduced in more than 10% of the times as it is mentioned in the reviewed literature, that has used bottleneck features in CNN [57]. CNN has a significant impact on influencing important ideas such as sparse interactions, parameter sharing and equivariant representations which are traditional Multilayer Perceptron techniques [50].

### 2.9.3 Convolution Neural Network (CNN) Architecture

Convolution Neural network is group of neural networks which are exceptionally accurate once applied to the task of image recognition. The training of Convolution Neural network is normally performed by adhering to backpropagation method [58]. Fig 2.9 represents the general structure of Convolution Neural network.

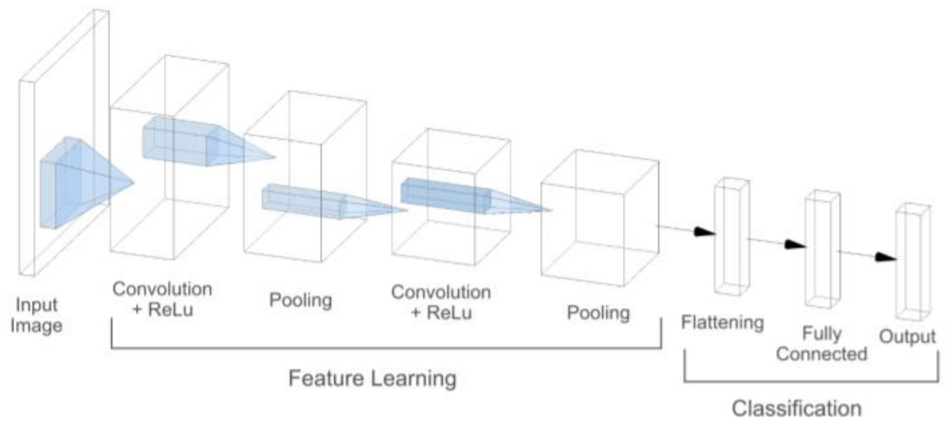


**Figure 2.9: General Structural representation for CNN.**

Fig 2.9 represents the general architecture of the convolution neural network (CNN). CNN can carry out both feature extraction and classification. The feature extraction comes with two different process, convoluting which includes ReLu, and pooling.  The data is input to the convolution layer, which is the first layer of the structure, as small squares of the image (matrices of pixels). The square pyramids of the Fig 2.9 represent the small squares of the image data. The Convolution layer is responsible for performing the mathematical operation (Fig 2.10) to combine the image matrix and the kernel matrix. Then the data transfers to Relu(Rectified Linear Unit) layer to set all the negative values, which are in the matrix, to zero.  After passing through the convolution and ReLu layer, data enters to the pooling layer where the image is downscaled when the image is too large. The pooling layer performs an essential role in image pre-processing. This output passes through repetitions of convolution, ReLu and pooling layers [59] and later enters to the flattering layer where the classification of the images begins. The Flattering layer converts the data into a one-dimensional array (vector format) and then pass it to the next layer called the Fully Connected layer. This fully connected layer is a neural network, and it is responsible for combining the features of the feature map matrix to generate the model.

***Convolution Layer and ReLu layer of CNN:***

The Convolution layer is the first layer, the image enters, and that layer split the image into matrix of pixels. This Convolution Layer contains neurons, and each neuron is connected to a filter. A filer is weight matrix. The input image’s matrices have been assigned to one separate neuron. All neurons in each layer apply the exact same filter to those matrices. This permits the different neurons to be active in different patterns on the input image. The filter itself performs dot multiplication along with the appointed section of that input matrix. All of those scalar values are be stored to represent that section in a brand-new matrix. In summary, the Convolution layer maintains the relationship between each pixel through learning image features utilizing small squares of feed data. It runs as a mathematical operation which obtain two inputs: namely, filter or kernel and image matrix. That mathematical operation is represented in the Fig 2.10. The dimension of the image matrix is given by h x w x d while the dimension of filter or kernel matrix is given by $f_h$ x $f_w$ x d. The output is given by ($h$-$f_h$ +1) x ($w$-$f_w$ +1) x 1.

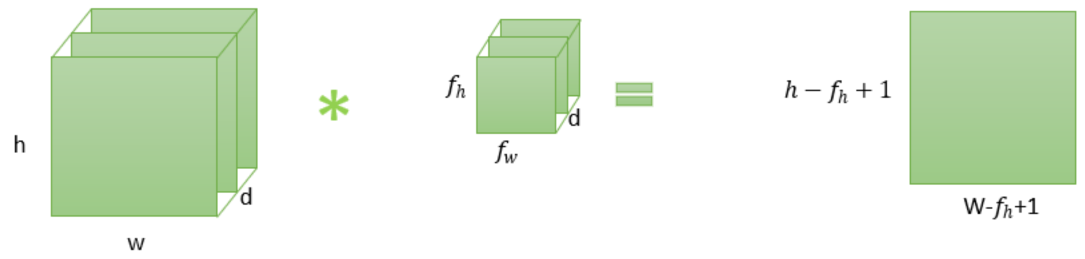


**Figure 2.10: Mathematical operation representation between image matrix and filter matrix in CNN.**

Thereafter, the Convolution layer conduct the Rectified Linear Unit (ReLu) technique, which decreases the unwanted pixels and enhances the non-linearity while replacing negative values by zero. Those values are placed in another new matrix called the new feature map, which then transferred to the following layer, the pooling layer [60].

***Pooling Layer of CNN:***

The ReLu's(Rectified Linear Unit) output matrix is passed into next layer, the pooling layer of the CNN network [58]. This pooling layer further reduces the convolved feature map's size [60]. It gets the input image, then splits that into small patches [58]. Then that pooling layer gets a certain value from each patch and locates that in a new matrix. This particular value will be either average value, minimum value or maximum value. This layer reduces the size of the matrices and hence operates as a regulator of the network. That new matrix is called the new pooled feature map, which is then transferred to the following layer [60].

***Flattering Layer, Fully connected Layer, and output Layer:***

After passing through all the above steps for feature learning, the final pooled map should be flattened [61]. This is achieved by converting the feature map matrix into the column matrix. The fully connected layer is the penultimate layer of the CNN network [58].It conducts the multiplication with weight matrix and flattened matrix, then includes a bias vector to classify the input images into predefined classes. This output layer (the final layer) contains the class-belonging probability of each class [60] . Nevertheless, prior to output the results and scores, in order to make sure the class probabilities are in between zero and one, the Softmax function is used [61]. The Softmax function is regularly used as the final activation function of the CNN neural network in order to normalize the network output to  a certain probability distribution over output classes which have been predicted by the final layer.

## 2.10 Transfer learning technique for gesture recognition

The main problem with Convolution Neural Network (CNN) is that such a large network requires higher GPU performance when doing the training process. That is quite difficult to achieve due to limited resources [62], but without high performing GPU, the task will be very slow. CNN also requires large amounts of data for a training from scratch, which takes quite a long time for training. Transfer Learning method has become the solution for those challenges [63]. Transfer Learning method is applied with a pre-trained model and it can leverage the knowledge from a source to enhance learning on another [63], which means the transfer learning technique is a method to enhance the domain knowledge by learning and recognizing more classes. As an example, a Convolution Neural Network (CNN) capable of recognizing dogs and cats can be retrained to enhance its knowledge and be capable of recognizing monkeys, or else it can be re-trained to identify deer and goats separately while reusing the knowledge gained in learning to identify the dogs and cats. Due to the massive volume of computational power, which is required, transfer Learning method is also widely used in computer vision, and natural language processing, such as sentimental analysis.

In the transfer learning method, the last few layers or the very last layer of the pre-trained model will be unfrozen and re-trained in the new dataset to obtain the required outcomes. The learned parameters of the network from the pre-trained resources are saved and stay in vector format which has been transferred to new model along with specific and new data in order to train the model. Transfer learning eliminates the requirement of training the whole network by transferring pre-trained model's knowledge, and this helps to reduce the requirement for large datasets [64]. Unfreezing and re-training multiple last layers will provide lower learning rate compared to unfreezing and re-training one last layer as this will support the last few layers' weights. The selection of how many layers and which layers are needed to be unfrozen for re-training depends on the new dataset size and the how similar it is to the original dataset [65]. Conducting the transfer Learning method on multiple layers using relatively small datasets, results in overfitting high risk. There is a specific size reference related to the number of network layers that needs to be considered when using the Transfer learning technique [66], [67] [68]. More specific and most important features of data are contained the bottom layers of Convolution Neural Network (CNN). When the original data is similar to the new data, the best result can be obtained by re-training the last layers or the bottom layers while freezing the top layers as they signify the generic features of the model.

Transfer learning is a method which was implemented to overcame problems of CNN and to increase the usability of the system by leveraging knowledge from one source to another via a pre-trained model. However, a pre trained model is only a variant of transfer learning which includes a general method as well. A general method is an approach of modelling a pre-trained Deep Learning model on a problem. This method cannot ensure the levels of performance since it depends on the pre-trained model that will be applied. All the pre-trained models consist of many sorts of image classification data [55]. So originally it would be a reference which could be used as a pre trained models.

### 2.10.1 Transfer Learning Models

With the rapid evolution of the Artificial Intelligent field, dozens of top performing pre-trained models have been developed for image recognition tasks, and those pre-trained models can be directly downloaded from several resources. They are being used as the base for image recognition task and some other related tasks in computer vision. Among those, three models are the most popular models in the AI world [69].

- GoogleNet (ex: InceptionV3)

- VGG (ex : VGG 19 or VGG16)
- Residual Network (ex: ResNet50)

These are widely used in the transfer learning method due to their high performance and the architectural innovations in their structure. As examples, VGG (Visual Geometry Group) model comes with repeating structures, GoogleNet models come with inception modules and ResNet models come with residual modules. Those pre-trained models are very useful in performing transfer learning in Convolution Neural Network (CNN).

Fig 2.11 illustrates the concept of transfer learning procedure and how it works in Convolution Neural Network (CNN). In Fig 2.11, the transfer learning procedure is presented graphically by using the pre-trained network A and customized network B. In the transfer learning technique, concept of Layer Freezing is used. As per that Fig 2.11, the initial layers which are displayed as rectangles are frozen to prevent updating the layer weights. That means, parameters of the Network A transfer to Network B and then initiate training the neurons of the specific layer set at the end of the network A or adding a new layer set to the same. After completing the retraining process of Network A, Network B can be obtained.

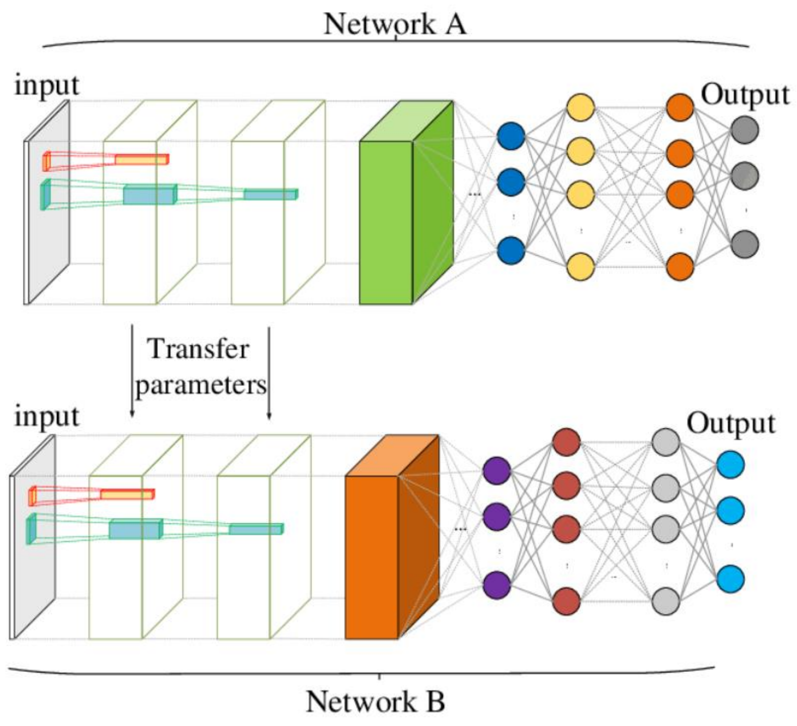


**Figure 2.11:Concept of Transfer Learning in Convolution Neural Network (CNN).**

Table 2.1 illustrates the higher-ranking datasets in ImageNet and their predicted accuracy and the parameters which are obtained by the 'Keras' official website. Further Top1 and the Top 5 accuracies denote the performance of each model on the validation dataset of ImageNet [70].

**Table 2.1:Some of the main Pre-Trained models in ImageNet and their performance. [70]**

| Model | Size | Top-1 Accuracy | Top-5 Accuracy | Parameters | Depth |
|---|---|---|---|---|---|
| NASNetLarge | 343 MB | 0.825 | 0.960 | 88,949,818 | - |
| InceptionResNetV2 | 215 MB | 0.803 | 0.953 | 55,873,736 | 572 |
| InceptionV3 | 92 MB | 0.779 | 0.937 | 23,851,784 | 159 |
| ResNet152 | 232 MB | 0.766 | 0.931 | 60,419,944 | - |
| ResNet50V2 | 98 MB | 0.760 | 0.930 | 25,613,800 | - |
| VGG19 | 549 MB | 0.713 | 0.900 | 143,667,240 | 26 |
| MobileNetV2 | 14 MB | 0.713 | 0.901 | 3,538,984 | 88 |
| MobileNet | 16 MB | 0.704 | 0.895 | 4,253,864 | 88 |

### 2.10.2 Inception Model

The pre-trained model called Inception network has been a major achievement in the growth of Convolution Neural Network (CNN) classifiers. Before the appearance of the Inception model, most of the popular Convolution Neural Networks (CNN) just packed convolution layers much deeper in the hope of obtaining better performance [71]. Inception network was heavily engineered (complex) and has been built using numerous tricks for increasing the performance in terms of accuracy and speed. The constant evolution of an Inception network has led to the formation of a number of versions of that network. The most popular versions are: InceptionV1, InceptionV2 and InceptionV3. Each version of Inception is a significant improvement on the previous version.

***InceptionV1:***

InceptionV1 module is implemented mainly to identify the proper kernel size for a certain convolution operation. Then the developers designed an Inception model which included filters with different sizes while operating in the same level. That makes the network a bit wider rather than deeper. The network architecture is shown in the Fig 2.12 and is called “Naïve Inception Model”. It is able to act upon convolution inputs with three unique sizes of filters namely, 1x1, 3x3, and 5x5. Addition to that it can perform maximum pooling. Then it can concatenate and send the outputs to the inception module which is next in line.

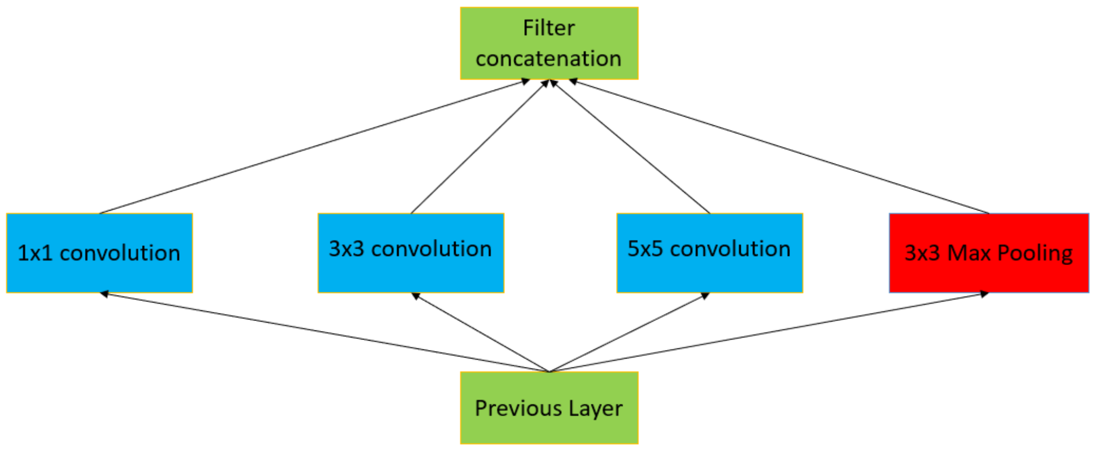


**Figure 2.12:Naive Version of Inception Module. [72]**

As mentioned earlier, Deep Neural Networks have been computationally expensive. In order to decrease the computational cost, the number of input channels has to be limited by adding additional 1x1 convolution prior to both 3x3 convolution and 5x5 convolution. Even though adding an additional operation seems to be counterintuitive, convolutions like 1x1 are cheaper than convolution like 5x5, and it reduces the input channel quantity. Therefore, the naïve version of the inception module was modified to obtain the neural network architecture of GoogleNet Inception module which is mentioned in Fig 2.13. This module is named GoogleNet (InceptionV1).

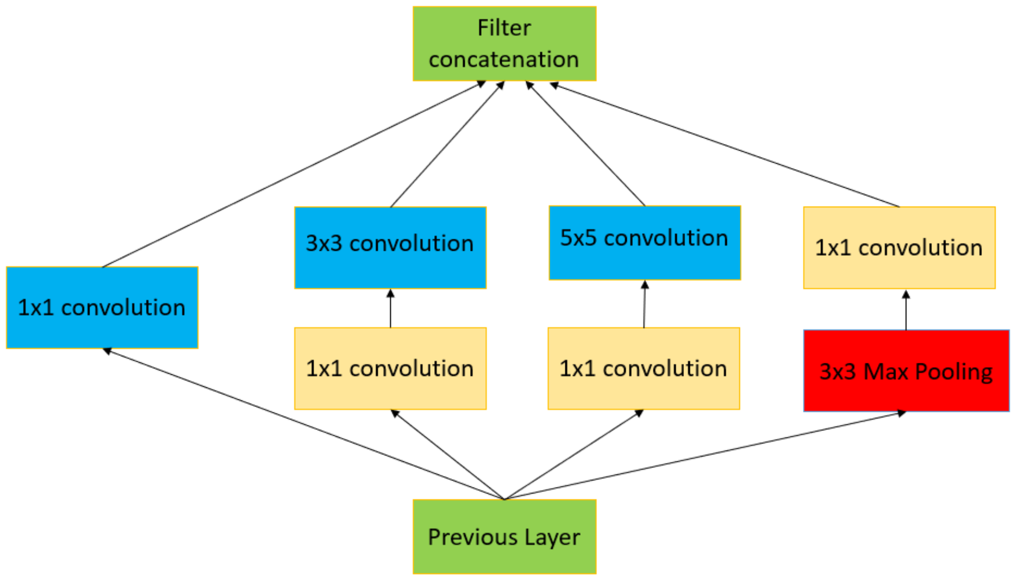


**Figure 2.13:Inception Module -dimension reduced. [72]**

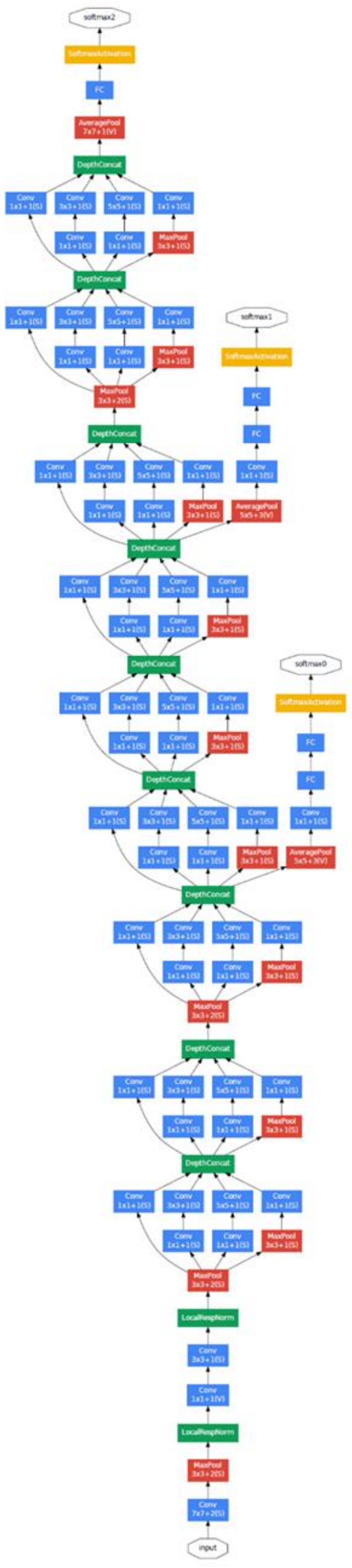


**Figure 2.14: InceptionV1 Architecture. [72]**

The architecture of GoogleNet (InceptionV1) is show in the Fig 2.14. This GoogleNet module has nine separate inception modules piled linearly and twenty-two layers. Including pooling layers, there are twenty-seven layers altogether. The module utilizes a global average pooling type at the end of the final inception module. It is a deep classifier. Two auxiliary classifiers have been introduced in order to avoid dying out the middle section of the network. SoftMax has been applied to the outputs of 2 inception modules while computing an auxiliary loss across the same labels. The total loss of the function includes the weight sum auxiliary loss and real loss.

***InceptionV2:***

Later GoogleNet developers upgraded the Inception module by increasing the overall accuracy and the decreasing the computational complexity. The GoogleNet(InceptionV2) module was introduced by modifying the following [72].

- Factorizing the 5x5 convolution into 3x3 convolution process in order to improve computational speed of the module. This has been shown in the Fig 2.15 below. It can be identified that the 5x5 convolution at the left most corner of the older inception module (refer Fig 2.13) has been replaced by two numbers of 3x3 convolutions.

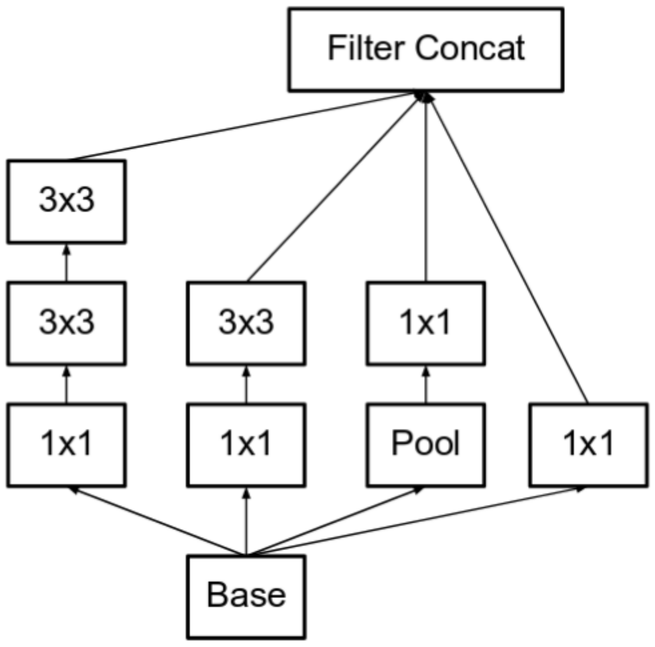


**Figure 2.15: Modified InceptionV2 module according to 1st Principle. [72]**

- The developers have further factorized the filter size nxn convolutions to a mixture of nx1 and 1xn convolutions. They have found that this technique could be 33% cheaper than using a 3x3 convolution. This has been illustrated in Fig 2.16 below.

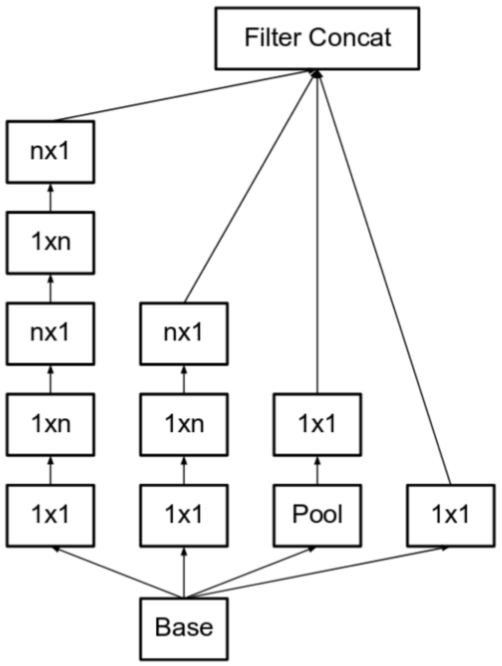


**Figure 2.16:Modified InceptionV2 module according to 2nd principle. [72]**

- The filter banks of the Inception module are expanded widely but not deeply to eliminate the representative bottleneck. If the module is created much deeper, there will be an unnecessary reduction in module dimension which leads to loss of information. This is demonstrated in Fig 2.17.

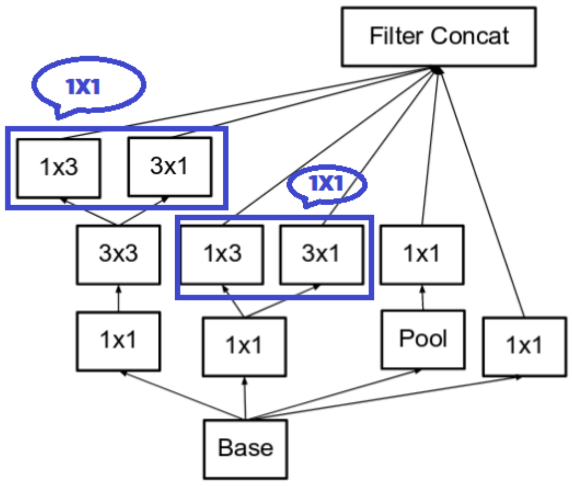


**Figure 2.17: Modified InceptionV2 module according to 3rd principle. [72]**

***InceptionV3:***

The auxiliary classifiers of InceptionV2 did not provide much up to the final part of the training process [72].Then the GoogleNet developers have investigated possible techniques to modify the InceptionV2 module without drastically changing the entire module. They have been able to do the necessary upgrades identified in the InceptionV2 module. In addition to those upgrades, following features have been also added to InceptionV3 module.

1. Factorized 7x7 convolutions.
2. RMSProp Optimizer.
3. Label Smoothing (This is one type of regularizing gadget has been added up to loss formula which avoid the network from turn out to be too confidence on a class. This can prevent over fitting.)
4. Batch Normalization to Auxiliary Classifier.

This InceptionV3 module has been widely used as an image recognition model with higher accuracy at the ImageNet dataset. This model has become the culmination for numerous innovative ideas implemented by various researchers over the past years [72]. Briefly, the InceptionV3 model of GoogleNet consists of asymmetric and symmetric building blocks that include convolution, max pooling (where maximum value for every patch on CNN feature map is calculated), average pooling (where average value for every patch on CNN feature map is calculated), dropout (hidden and visible ignored neurons) fully-connected layers and concatenated. Batchnorm or batch normalization (which is one the techniques used to train deep neural networks which standardizes inputs to certain layer of each mini batch) has been used right through the model and is utilized to the inputs in the activation. Softmax is used to compute the Loss in the IceptionV3 model.

InceptionV3 model has been illustrated in the Fig 2.18. In this model, basic convolutional operations including 1x1 kernels and 3X3 kernels learn the lower-level feature mappings of the model. Further multi-scale module feature representations have been concatenated for feeding into auxiliary classifiers together with diverse kernels in convolution (example: 1 × 1, 3 × 1, 1 × 3, 3 × 3, 5 × 5, 7 × 7 and 1 × 7 filters), which are utilized for producing greater convergence performance. Then, the Softmax classifier is used to generate a vector with a certain class probability.

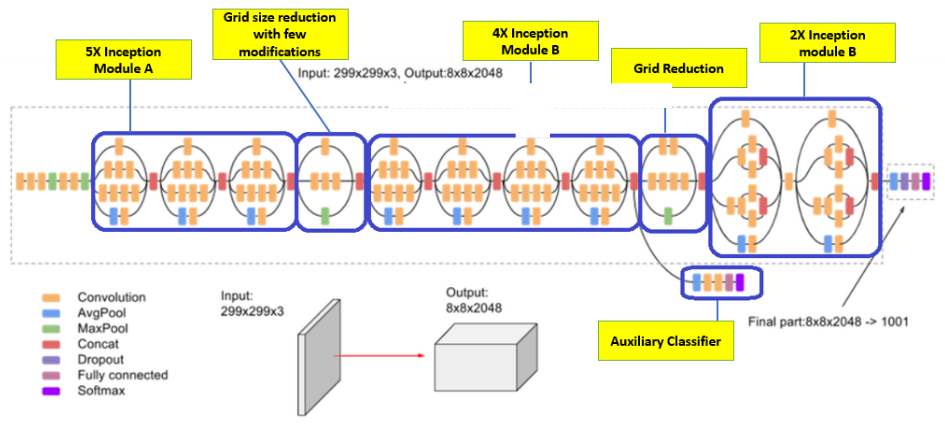


**Figure 2.18: InceptionV3 model Architecture.**

## 2.11 NVIDIA'S Jetson Nano Kit

NVIDIA Jetson Nano can be used as the main computer to conduct the training of image classification through Convolution neural network (CNN). NVIDIA'S Jetson Platform has initiated in 2014 to empower the embedded computing scenario with artificial intelligence. The Jetson Nano Developer Kit was launched in March 2019 and it has become one of the most interesting and affordable pieces of technology [73]. "Jetson Nano is one of the latest offerings from NVIDIA for AI applications in edge computing" [74]. The Jetson Nano Developer Kit is delivering powerful performances in the already used models such as image classification, segmentation, object detection, image processing, and pose estimation. This embedded computing is widely used in the Internet of Things (IoT) based systems, robotics and even in retail and agriculture.

In these sorts of recognition and processing scenarios, a system with a low power embedded Graphic Processing was essential for running various neural networks simultaneously and the computer vision algorithm techniques used for image recognition. Since it was selected for this research project, it should be mentioned that the Jetson Nano is able to run with advanced neural networks in widespread machine learning frameworks such as Pytorch, Keras, TensorFlow which are discussed further in the chapter 3 and chapter 4 of this thesis. Its innovative integrated accelerator hardware can be highlighted as one of the major steps towards the betterment of automatic learning algorithms. Key features of the NVIDIA'S Jetson Nano are the low power consumption due to its lightweight and real time performance in both hardware and algorithms [74].Ubuntu 18.04 LTS is the compatible operating system for this embedded computer, Jetson Nano, which is heavily reliable, even though the competitors such as Raspberry Pi 4 is officially a supported for 3 different operating systems.

Selection of a dedicated deep learning library is also critical for the buildup of the system which will simply the process with fast optimization algorithms. As examples of deep learning libraries, TensorFlow library which is a development of Google and Pytorch library which is a development of Facebook, have a high popularity rate. TensorFlow allows single and distributed architecture in a C++ based lower layer and supports several interfaces with complete functionality. Pytorch is a scientific computing framework which has a straightforward approach that could be easily reused and combined, even though limited resources are a downside. This uses Lua and C++ as the base language. Also, there are libraries such as Theano, Caffe and MXNET garner somewhat less interest compared to the two variants discussed because of the complexity in their techniques, small user bases, and maintenance issues. Developers tend to use fast optimization algorithms to enhance stability in systems [50]. Python is used as one of the interfaces or the language for these systems. Mostly Python scripts dominate the process, and all the deep learning libraries are opened with Python interfaces. The ability of Python to learn and the convenience it provides in training almost every deep learning model ensures the rapid development of Python as a language. It is important to build up strong configurations and arguments in the coding part of Python scripts to ensure image processing perfection. In this research study, TensorFlow has been used as the main open-source library and Python has been the main programming language along with the minicomputer, NVIDIA Jetson Nano.

# Chapter 3 : **Methodology**

The prior chapters elaborated the research objectives and research gap to be addressed in this study. With these in mind, the research Methodology for the study is based on the conviction that,

“**Dual-handed sign Languages, especially New Zealand sign language can be interpreted into text mode based on human hand gestures”.**

This chapter discuss on research strategy and the key performance matrices for evaluating the hypothesis. The identified Key Performance measures of the proposed study will be discussed in the chapter 4, System Design and Development.

## 3.1 Research Strategy.

This study focuses on studying how sign language gesture communication can be translated to written language using an Artificial Intelligence based system. As the initial step of the research, the hypothesis has been developed along with its objectives. The objectives of this research are developed to demonstrate the achievements of this work and how far it verifies the hypothesis. Key performance Indicators (KPI) are identified to provide a quantifiable performance measurement to define and exhibit the success factors and evaluate the progress towards the accomplishment of research goals. In order to verify the hypothesis, Convolution neural network (CNN) training has been initiated to implement a dual-handed sign language translator which would be a solution for the language barrier between deaf community and the hearing-unimpaired community.

A system for New Zealand Sign Language Gesture Recognition through computer vision and translating them into text mode has been designed, developed, evaluated, and compared with actual sign language gestures to verify the hypothesis of this study. Before initiating the system training, the required dataset needs to be collected. This has been done through a video camera to extract the image frames for each predetermined class. After collecting the dataset, training of the system has begun. The training program, which has been developed through Python language, has conducted the feeding operation of the dataset to the system, data pre-processing operation, feature extraction operation and final training of the CNN system. Apart from that,

that program has been designed to obtain the overall final test accuracy of the trained system and each train accuracy, validation accuracy and cross entropy at each step.

KPIs defined at the beginning of the study have been evaluated using the output from the CNN system. The Python training program has been set to provide the confusion matrix and for each letter which has been trained in the CNN network and the learning curves of the trained system. Then adhering to that confusion matrix data, F1 score which is another KPI of the research, has been calculated. These calculated KPIs are used as measurable values which demonstrate how effectively the trained Convolution neural network (CNN) system is achieving the key objectives of the research. As the output of the study, an application has been developed to translate the New Zealand gesture signs into written language in real time. This system has been tested with several approaches to verify the accuracy of each letter prediction. This system is supposed to translate New Zealand signs for the English alphabet and other custom symbols into text format. Most importantly it is expected to translate the signs continuously to display them as entire sentences.

## 3.2 Method Analysis

This proposed sign language translation system has been implemented through a computer desktop with 720P 'Logi' web camera and a NVIDIA Jetson nano Developer kit. The images of the human gestures have been captured by the 'Logi' camera and that images feed into the programmed system. The signer needs to perform their signing to the specific frame space of the desktop program and the system is designed to crop and resize the images captured by the camera in 'jpg' format. The conceptual framework of the designed system is illustrated in Fig 3.1.

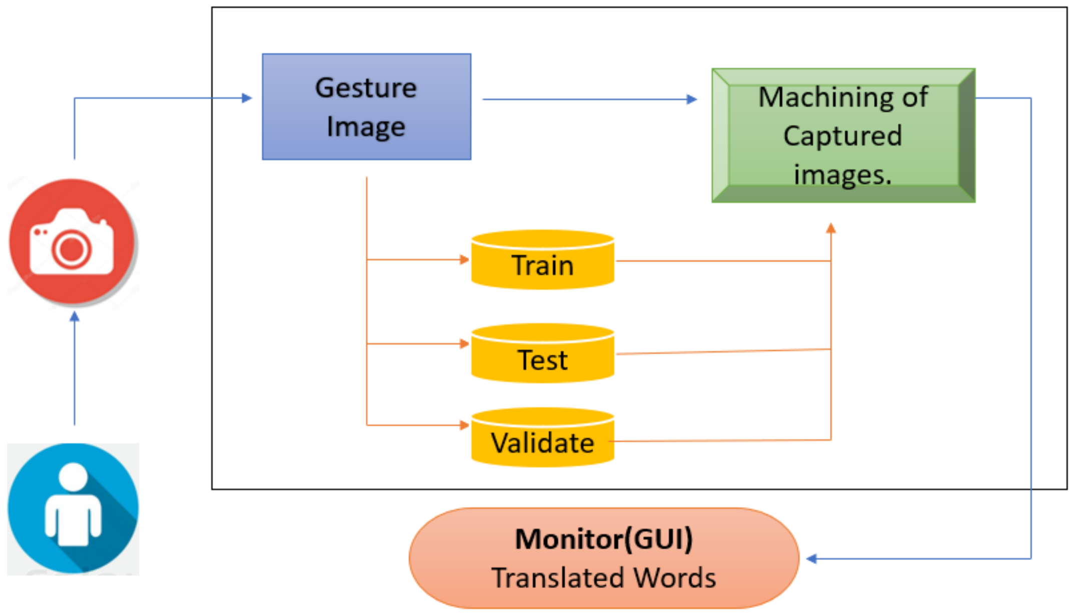


**Figure 3.1:Conceptual framework.**

It is necessary to gather the training dataset for the training of CNN network as the dual-handed sign language gesture image sets is not available in any resource as research on translating dual-handed sign languages, like New Zealand Sign language, has not yet been performed. As per the Fig 3.1, a set of gesture dataset is acquired by a camera, which has been used as training data, testing data and validation data in the training process of the network. Training dataset is the sample of data which is used to train and fit the model. Training of the model is actually conducted by this training dataset. A validation dataset is used to generate the unbiased evaluation for the model, which is fitted on a training dataset and validation data tunes the model hyperparameters. Then the model has used that data indirectly in order to adjust the hyperparameters but never to learn. Test data is sample of data which is used to generate the unbiased evaluation for the final model which is fitted on the training dataset. This test dataset is an untouched dataset which is utilized to evaluate the particular model. As per this specific study, dataset training is conducted using the transfer learning technique. At the moment when the test gesture input is presented to the camera, the trained CNN classifier recognize the relevant class of the performed sign and translate it into text . The translated text is displayed in the monitor.

This proposed research has been undertaken by focusing on signers' hand gestures. Human hand gestures of the signers have been presented as image dataset to train the system by image recognition implemented by a Convolution Neural Network (CNN). Due to the recent introduction of architecture optimizations for Convolution Neural Network (CNN), it has become state of art in the task of image classification [75]. The technique used in this research is transfer learning with a Convolution Neural Network. The Google (InceptionV3) model has been selected as the pre-trained network to conduct the transfer learning. Vision-based human hand gesture recognition method has been used to implement the system and a web camera is the main source for getting the inputs and the datasets to be trained. This has been designed to recognize 26 English alphabets and three other symbols such as 'Delete', 'space' and 'nothing' through human hand gestures using New Zealand sign language. This research is the initial step towards implementing a possible New Zealand Sign Language translator that can communicate in sign language while translating them into text mode or written language.

This research, implementing a New Zealand sign language translator has been conducted based on the image recognition technology, which is a method of evaluating the patterns in image sets to classify each image [76]. This image recognition method usually contains four major steps [77].

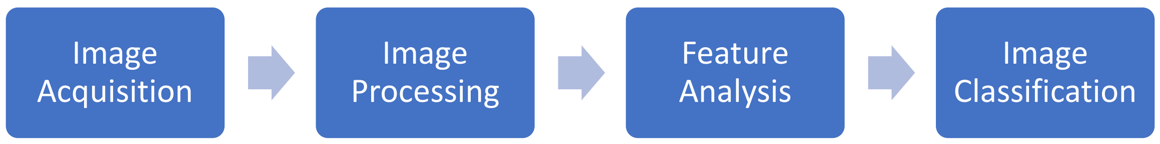


**Figure 3.2:Image recognition Process. [77]**

As per the Fig 3.2, in the image acquisition step of the image recognition method, unprocessed images are retrieved and relevant class for the image is defined. Categorized images and relevant classes denote the dataset for the model [76]. In the image processing process, after completing the skin detection task, images are converted into RGB color model to HSV color model. As the result of that, white pixels of the mask which are considered as the skin region in the frame, are produced. Then applying of thresholding to the images, the boundaries of the hand in the image can be identified. This brings up the external contours of the hand. In this step, other regions of other pixel values except skin color pixel are omitted. The program returns the co-ordinates(points) of boundary and their hierarchy of the hand in the image. Then the image data categorize into three main groups: training data, validation data and testing data sub classes. Then the images in the training sub class are augmented to let more noise to the data. The images in the testing sub class are also augmented to establish more practical real-life experience in testing. However, augmentation is not performed in validation dataset. The Python program written to conduct the training of the CNN, first creates an array for pixel values of dataset images, then, it performs distortions such as, cropping, scaling, flipping over each image to obtain better training results and. The next step is feature analysis.

The next step is to insert the pre-trained architecture to the system in order to conduct the transfer learning. After importing the InceptionV3 model and generating new dataset, the training process of the model of this study begins. While freezing the all the layers except pool_3 layer of the InceptionV3, the model is trained on new processed data with a new layer at the end. Features of the images (for example: edges, regions, corners) and identical patterns are analyzed at the training proc [60]. Pool_3 layer of the InceptionV3 model has been used as the bottleneck layer to conduct the feature extraction and bottleneck creation task. After processing the bottlenecks, the training of the new system begins with 2000 training steps. The prediction results then are compared with the actual trained labels(classes) and the weight of the final layer of the new CNN system is updated by the back-propagation process. Image classification is the last phase of the process, where unseen, new images can be classified to a class from the predefined classes.

The new model is trained and developed with the optimization built on the gradient descent method. The optimizer, stochastic gradient descent (SGD) algorithm and cost function, Categorical Cross entropy have been used as perform the classification of the model.

$${}_{w}\min J(w)$$

Where J(.) denotes the loss function and w denotes the vector of the weight.

By applying SGD, learning rate of the model can be fixed and weight updating can be performed in each iteration as mentioned in the below equation. (Where the η denotes the learning rate and $W_k$ denotes the weight vector.

$$W_k = W_{k-1} - \eta \nabla J(W_{k-1})$$

SGD is an optimization method which uses a fixed learning rate and the weight updation is done in every iteration as described in the equation. A sub class of that SGD, the mini-batch stochastic gradient descent (SGD) is applied in this model to split the training dataset into various small batches which are capable of calculating model error and updating model coefficients. Mini-batch stochastic gradient descent is a common gradient descent implementation used specially in deep learning field. Training process of the CNN network of this research takes place in Tensorflow-TensorBorad. The results of the conducted CNN training is set to store in different forms, such as, confusion matrix, training validation accuracy and training validation error by using skilearn library of Python.

The next section will discuss the identified Key performance Index (KPI) of this proposed New Zealand sign Language translator.

## 3.3 Evaluation and design measures.

Key performance indicators (KPIs) are fundamentals ensuring the efficiency of the system. They act as the bridge between strategies and the obtained results. Following section discusses the identified key performance index for the proposed system of translating New Zealand sign language. Evaluating the machine learning algorithm is vital part of any proposed CNN project. Even though the designed model has provided satisfying results at the evaluation of accuracy score, but a poor result outcome may be generated when evaluated when compared and contrasted to other metrics. Most of the time classification accuracy is used to measure the overall performance of the designed model; however, that is not enough to judge a model's performance. Therefore, evaluating a set of key performance Indicators for the certain model can be an ideal solution for judging the performance status. Fig 3.3 shows the map for Key performance Indicators (KPIs) which have been used to evaluate the proposed New Zealand sign

Language translator. It consists of confusion matrix, F1 score, Training and Validation accuracy curves, Training and validation loss curves.

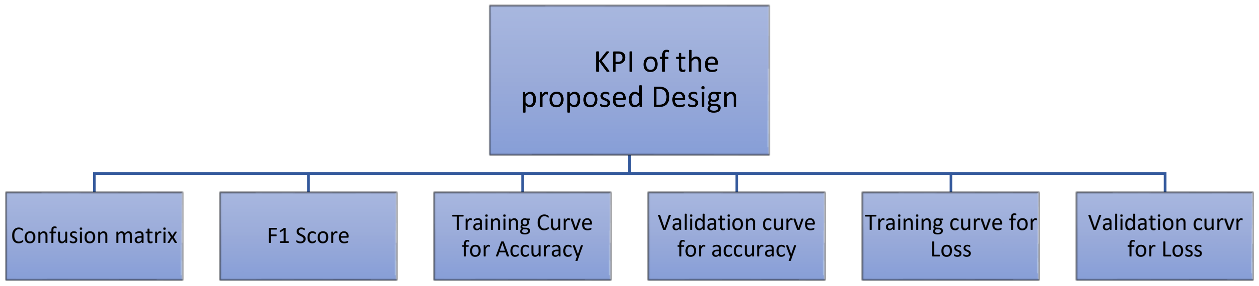

KPI of the proposed Design
Confusion matrix
F1 Score
Training Curve for Accuracy
Validation curve for accuracy
Training curve for Loss
Validation curvr for Loss

**Figure 3.3: Key Performance Indicators for Sign language translator.**

The following section of this chapter describes the identified KPIs in detail. The performance of the designed system has been considered when identifying the relevant Key Performance Indicators (KPI). These measurements are relevant throughout the development process of the research. Chapter 5 is dedicated to a discuss of the performance of the system designed for this research while evaluating the identified KPI for the proposed study.

### 3.3.1 Confusion Matrix

The Confusion Matrix also known as the Error Matrix, is one specific table layout in the machine learning field especially in the statical classification area. This enables the visualization of algorithm's performance usually in the supervised learning category. Every row of the certain matrix signifies the occurrences in the predicted class, while at the same time every column signifies the occurrences in one actual class or else vice versa. The confusion matrix presents as NxN matrix, where the letter "N" is the certain number of each target class. The matrix compares every actual target figure with those predicted by the model in machine learning. This provides the whole view of the performance status of the classification model and what types of errors that can be made. As an example, for the binary classification problem, a 2x2 matrix is shown in Fig 3.4 below.

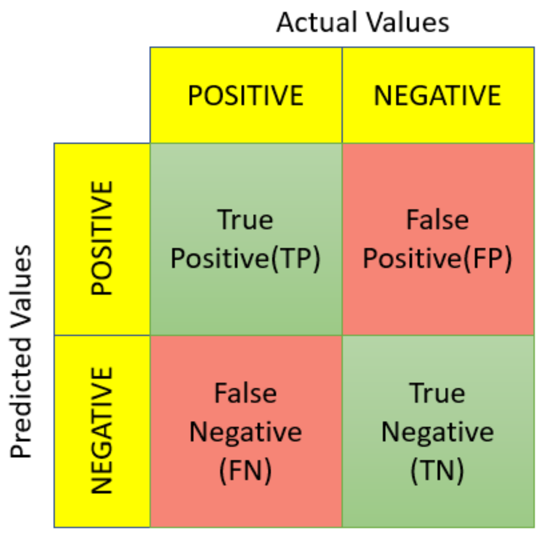


**Figure 3.4:Confucion Matrix for two class classification problem. [78]**

As per the Fig 3.4, the target variable consists with two values namely, Positive and Negative. Actual values of the targeted variables are represented by the columns. Predicted values of targeted variable are represented by row. The figure shows, there are four different outputs, True Positive (TP), False Positive (FP), False Negative (FN) and True Negative (TN).

**True Positive (TP):**

- The predicted value completely matches with the actual value.
- The actual value is positive, and a positive value is predicted by the model.

**True Negative (TN):**

- The predicted value completely matches with the actual value.
- The actual value is negative, and a negative value is predicted by the model.

**False Positive (FP)**: Also known as "**Type 1 error.**"

- The predicted value is incorrectly predicted.
- The actual value is negative, but the model forecast a positive value.

**False Negative (FN)**: Also Known as "**Type 2 error**"

- The predicted value is incorrectly predicted.
- The actual value is positive, but the model forecast a negative value.

Predicted values are described as Positive or Negative, and actual values are described as True or False (refer Fig 3.5).

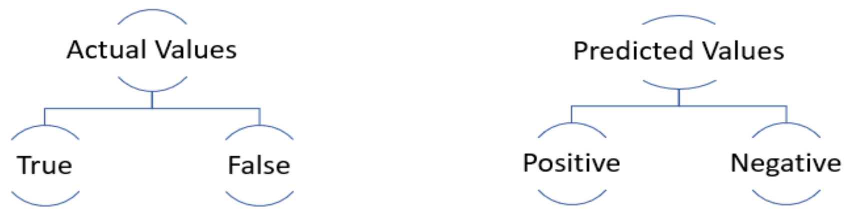


**Figure 3.5:Indication of Actual Values and Predicted Values of Confusion Matrix.**

**Accuracy Paradox:**

Sometimes, selecting a model with a low accuracy rate is more beneficial as it comes with a higher predictive power. As an example, if there is huge class imbalance, the model is able to predict the majority class's value for all predictions, then obtain a high-level of classification accuracy. The issue is that the model is not valuable in that problem domain. So, such models require additional measures or key predictive analytics to evaluate the particular classifiers. Confusion Matrices are normally used to identify key predictive analytics such as accuracy, recall, precision, and specificity. Confusion Matrices are very beneficial as they provide straight comparison of values such as False Positives, True Positives, True Negatives and False Negatives.

**Accuracy:**

Accuracy is just the ratio of the accurate predictions to the sum of the predictions. For binary calculations, accurate predictions include both true positives and true negatives.

Accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$

Accuracy alone does not provide the judgment of the overall performance of the model, especially when dealing with class-imbalanced datasets.

**Precision:**

TP is the sum of true positives while the FP is the sum of false positives. Precision is the sum of the positive predictions divided by sum of the predicted positive classes. This is also known as "Positive predicted value" (PPV).

Precision = $\frac{TP}{TP+FP}$

Precision can reveal the exact number of earlier correctly predicted values determined to be positive.

**Recall:**

Recall can reveal the exact number of positive cases which are able to predict accurately in the designed model. Recall is also known as “True Positive Rate” or “Sensitivity”.

Recall = $\frac{TP}{TP+FN}$

### 3.3.2 F1 Score

The F1 score is one type of weighted average of sensitivity or the recall and precision. The F1 score can be ideal solution when it requires a good balance between Recall and Precision. Practically, when trying to increase the model’s precision value, the recall value automatically goes down and vice-versa. This F1-score denotes the harmonic mean of Recall and Precision. Further, it provides the blended idea about both metrics. The standard equations for calculating F1 score are given below.

F1 Score = $\frac{2}{\frac{1}{Recall}+\frac{1}{Precision}}$

F1 Score = $2\ X\ \frac{Recallx\ Precision}{Recall+Precision}$

### 3.3.3 Training Curve and Validation Curve

Training curve for accuracy, validation curve for accuracy, training curve for loss and validation curve for loss are different types of learning curves in Machine learning Technology. Those learning curves are widely applied diagnostic tools for algorithms in Machine learning. Problems with model learning like overfit model and underfit model can be diagnosed by evaluating learning curves of the relevant models. Usually, learning curves are plotted to show experience or time on the axis x and improvement or learning on axis y. Learning curves(LC) are considered to be effective tools to monitor the performance of the classifications. Those curves denote a mathematical representation for the learning process that takes place during task repetition [79].

LCs are commonly used in algorithms in machine learning that learn incrementally by optimizing its internal parameters, like deep neural networks. Maximizing can be the metric used when evaluating the learning process, such as classification accuracy. When performing the training process of the machine learning model, its status at each step of the training algorithm can be evaluated. It can be assessed on training datasets to evaluate how far the model can "learn". It can be assessed on hold-out dataset(validation) which is not a part of training dataset. Performing the evaluation based on a Validation dataset provides an indication on how well the certain model is adapting properly to novel, unseen data which is called "Generalizing".

- **Training Learning Curve**: This is one type of a learning curve (LC) which is calculated by a training dataset. This provides information on how well a certain model is "Learning".
- **Validation Learning Curve**: This is one type of a learning curve (LC) which is calculated by the validation dataset(hold-out). This provides information on how well the certain model is "Generalizing".

In most of the designed machine learning models, it is very common to generate dual learning curves in their training process for both the validation dataset and the training dataset. In addition, some cases learning curves are produced based on multiple metrics. In such cases model performance and cross-entropy loss are evaluated by using classification accuracy. In that method, two plots can be created: learning curves for each of the metrics, and learning curves for both training dataset and validation dataset.

- **Optimization learning Curves:** This is a type of a learning curve (LC) which is calculated based on the metric in which the model's parameters are being optimized. Example: Loss.
- **Performance Learning Curve:** This is a type of a learning curve (LC) which is calculated based on the metric in which the model is selected and evaluated. Example: Accuracy.

## 3.4 Summary.

To evaluate the research question, a real-time New Zealand sign Language translator has been designed, built, assessed, and compared to the currently available sign language translators. This real time translator is expected to translate New Zealand human hand signs into complete sentences in text format. The KPI's which have been identified in this chapter, directly participate in the evaluation process of this real-time New Zealand sign Language translator.

.

# Chapter 4 : **System Design and Development**

This chapter explains the conceptual design and the development process of this proposed system for translating New Zealand sign language into text mode by capturing human hand gestures. Firstly, selecting criteria of the suitable pre-trained model for the study are discussed. Secondly, data collection and data pre-processing are discussed prior to introducing the training process. Then training of Convolution neural network of this system (CNN) are introduced. Finally, the overall system design including hardware selection and software selection are presented to introduce the system architecture of the entire system.

## 4.1 Pre-trained Model Selection

As per the literature review and the prior section, different methods have been used to implement sign language translators through Convolution Neural Network (CNN). After studying prior research studies, the transfer learning method has been selected to train the Convolution Neural Network (CNN) of this proposed sign language translation system. As detailed in the prior section, there are multiple pre-trained datasets are available to conduct transfer learning. The following sections explains how the best pre-trained data model has been selected.

The pre-trained model, Inception V3 primary aims at burning lower computational power by revising and modifying the initial Inception Architectures [80]. InceptionV3 of GoogleNet has proven and demonstrated to be more a computationally efficient dataset in the term of the economic cost acquired (memory of the system and other resources), and it is also better than other pretrained networks in terms of the number of data or the parameters [81].

When implementing InceptionV3, several techniques have been used to loosen the constraints in order to make the model east to adapt. Those techniques contain regularization, factorized convolutions, parallelized computations and dimension reductions. The model basically functions as multiple numbers of convolution filters which are applied into the same input, including with several pooling. Then the results are concatenated. These phenomena let the model gains the advantage of feature extractions in multi-levels. For instance, the model can extract large filter size, 5X5 and small filter size, 1X1, at the same time. Further applying multiple features from certain multiple features enhances the performance of the network. In addition to the above authentic features, one other fact makes InceptionV3 far better than other models. Most of the pre-trained models prior to Inception, operated convolution in channel wise and spatial wise domains together; but the Inception model performs cross-channel correlations by performing 1X1 convolution while ignoring the spatial wise dimensions. Cross-channel and cross-spatial correlations follow this via 3X3 filters and 5X5 filters.

Since 2014, Inception V3 has become popular amongst AI researchers mainly due to several major accomplishments. The Large-Scale Visual Recognition Challenge 2014 (ILSVRC2014) has ranked GoogleNet's InceptionV3 in 1st place while placing the VGG16 pre-trained model in the 2nd place in the image classification category. The lower error rate (high efficiency) and the less complexity of the architecture of the Inception model have been the major trigger for the above-mentioned achievement. Further, InceptionV3 pre-trained model requires 92MB storage capacity, which is much lower than other popular pre-trained modules such as, ResNet, NASNetLarge and VGG. Nevertheless, IncetionV3 model generates higher accuracy levels than other pre-trained models, except NASNetLarge.

The Inception pre-trained model provides greater benefits to the users compared to other popular models. The web site, Analytical Vidya [82] confirms that the Inception models of GoogleNet can perform convolutions with various filter sizes for the input, carry out Max Pooling and link the result outcome for Inception module which is next inline. The Inception module can reduce the parameters (number of data) drastically by introducing a 1*1 convolution operation. This is one significant point which highlights the high performance of the Inception models. Christian Szegedy,Vincent Vanhoucke, Sergey Ioffe and Jon Shlens have determined the Inception model of GooleNet to be well performing pre-trained model [83]. In their paper, they compared this Inception model with other more popular models, such as VGGNet, AlexNet and selected the Inception model. Further they stated that even though the VGGNet model comes with a simple architectural feature, it comes with a high cost; it requires numerous computations when evaluating the network [84]. Further they have highlighted the higher performance of the Inception model of GoogleNet even under harsh constraints on computational budget and memory [85]. As an example, GoogleNet used around seven million parameters (in Inception V1), which represents 9x reduction compared to its predecessor AlexNet, which employed around sixty million parameters. In addition, VGGNet utilizes 3x more data/parameters than AlexNet [83].

Considering all of the above facts, it is clear that InceptionV3 of GooleNet exihibits higher performance compared to other popular pre-trained models. Most importantly, the Inception V3 model provides more benefits than other pre-trained models. In order to obtain higher performance including higher accuracy, the New Zealand sign Language translator uses in this research is InceptionV3 model and the last layers(pool_3) of the InceptionV3 model has been trained using captured gesture images.

After processing the training for 87,000 numbers of image data through Google InceptionV3 architecture in a Convolution neural network (CNN) for nearly 12 hours, the trained system is able to translate the signer’s hand gesture signs into text mode. This proposed system has been trained to translate the 26 Letters of the English Alphabet into New Zealand sign language, and then has been improved up to translating sentences instead of individual letters.

## 4.2 System Development

As per the Fig 4.1, the entire process of translating New Zealand sign language is based on three main stages. As the initial step, training datasets need to be captured: in this research a “Logi Tech” camera has been used to capture the gesture image sets. Training the datasets is the next step. For that a Jetson Nano developer kit has been used to train the algorithm using the transfer learning technique as this device is capable of participating in the higher computational cost, especially in large scale dataset model training processes. In the classify image stage, with the help of trained model, the output prediction has been displayed as words or sentences.

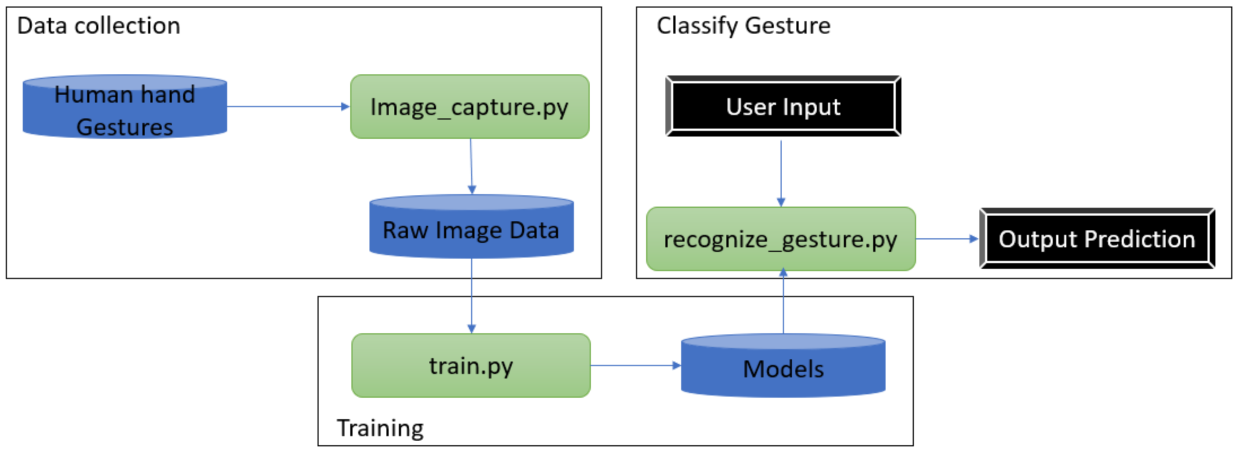


**Figure 4.1: System block diagram for the software.**

### 4.2.1 CNN Structure of This Research

The Convolutional Neural Network structure for this research follows the basic setup of the InceptionV3 pre-trained model. The simplified architecture of Inception V3 convolution neural network (CNN) from TensorBoard is in Fig 4.2. InceptionV3 is a deep neural network model which is difficult to train directly using a low powered machine. TensorFlow offers a tutorial for training the final layer of IncetionV3 for new classes using transfer learning technique [86]. As mentioned in the Methodology chapter, the transfer learning method has been used to maintain the parameters of the former layer while removing the last layer of GoogleNet(InceptionV3) . Finally, train.py retrains the last layer of the InceptionV3 model [72]. The total number of the classes in the dataset is equal to the number of nodes of the output at the last later of the CNN.

A simplified CNN layer structure for this research displayed in Fig 4.2.

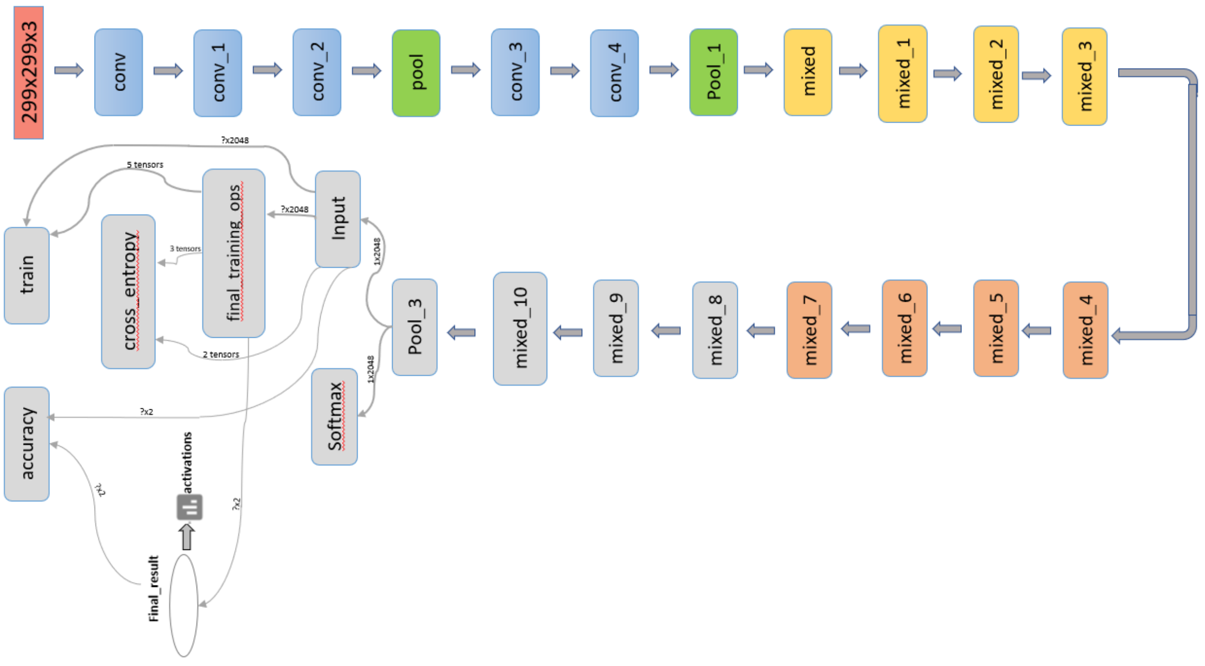


**Figure 4.2:Simplified CNN structure of the research.**

Image sizes of 299X299 are input to the InceptionV3 model in colored format. Then the input images go to first convolution layer which comes with three sets of filters. There are filters in 3x3 size with stride 2(conv), 32 filters in 3x3 size with stride 1(conv_1) and 64 filters in 3x3 size with stride 1(conv_2). Every convolution layer performs batch normalization and ReLu operation in each layer. Then the data is passed to the pool layer(pool), which is 3x3 size and stride 2(pool). After pass by that pool layer, the image passes two convolution layers which are in two filter sizes. They are marked as conv_3 and conv_4 in Fig 4.2. Conv_3 comes with 80 filters in 1x1 size-stride 1 and conv_4 comes with 192 filters in 3x3 size -stride 1. As mentioned before, after every convolution layer, there is batch normalization and ReLu operation. Then the image is passed to another pool layer (pool_1) which comes with the size of 3x3 and stride 2. After this point the data will start to enter to the Inception block of the model and Inception operation will initiate from here. As the first step of the Inception operation, the images enter to three sets of Inception module A; namely, mixed, mixed_1 and mixed_2 in order to lower the computational cost by reducing the parameters. After reducing the parameters, the width, and the height of the gride is changed by forwarding the image to reduction block A, mixed_3. Then it passes to the Inception module B. The task of module B is repeated four times. After completing this step, image directs to a reduction block (Reduction Block C), mixed_8. Then the Inception operation goes through two sets of Inception module C, mixed_9 and mixed_10.

After successfully completing the Inception operation, CNN architecture is retrained using Pool_3 layer. Pool_3 layer is retained for classifying the New Zealand sign alphabet and other several custom symbols. A fully-connected layer with 2048 neurons can be obtained after completing the retraining. Then the classifier layer can be obtained. Softmax fuction is applied to gain 1000 neurons to the classifier. The output Softmax layer in the CNN structure is responsible for returning the probabilities of every patch belonging to every class and the label of patch is determined by the highest probability.

### 4.2.2 Data Collection

This research aims to translate a dual-handed sign language, New Zealand sign language. Although sign language gesture datasets are available for single handed sign languages such as American sign language, Indian sign languages, in many online databases, gesture datasets for dual-handed sign language are not available. Therefore, training a convolution neural network system to translate hand signs of the dual-handed New Zealand sign language requires a high-quality classification dataset. However, the literature review found there is no evidence for any available database relevant to New Zealand sign language image sets. Consequently, required classification dataset has been collected in order to perform the training.

### 4.2.3 Captured Dataset.

The dataset for this research study has been gathered by running image_capture.py, Python program. Images in the dataset are 200 pixels in width and 200 pixels in height from depth images. A Python program for capturing gesture images has been designed to create a Regional of Interest (ROI) having the dimensions of 100mm from top, 150 mm from right, 400mm from bottom and 450 mm from left of the capturing screen. Only the gesture in the ROI area have been saved as an image; therefore, the entire hand gesture needs to be inside the ROI area when performing the gestures in front of the camera. The gestures in the ROI area are constantly tracked until the program achieves the required gesture images. The program itself flips the frame as to avoid the difficulty of mirrored images. The program has set to capture 3000 images per each class. The required number of classes and the number of images can be entered to the running terminal of the Python program. For this proposed study, 29 different classes have been added. This includes 26 different New Zealand signs for English alphabet and three other gestures for “Delete” function, “Space” function and “Nothing” (refer Fig 4.3). Each class consists of 3000 different images. The entire dataset contains 87,000 images.

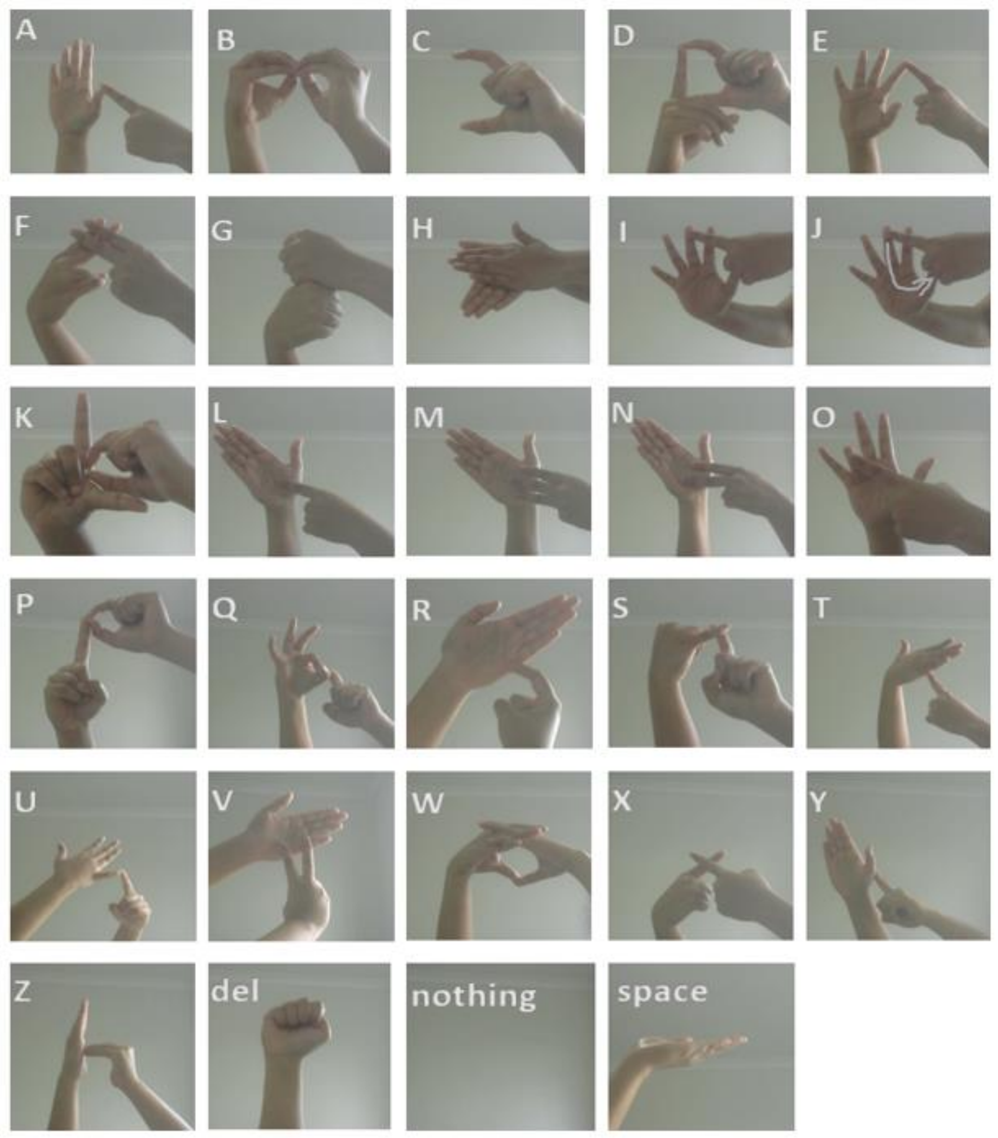


**Figure 4.3: Sample Dataset for this proposed NSL Translator.**

In order to obtain the best performance of the trained system, the input dataset should be gathered by having intra-class variations, such as shape variations, rotation variations and orientation variations [87]. In the dataset, the signers have changed the hand orientations and rotated the gestures when the images in the same data class are being captured. Apart from that, accuracy of the overall performance of the sign language translator can be obtained by varying the background of the sign gesture and lighting the background of the sign gesture. [88]. Therefore, the dataset for this proposed New Zealand sign language translator is also captured and collected under different background lighting conditions, rotating the sign, adding obstacles to the background, and varying the shape of the sign. Some samples of the intra class variation of the dataset are given in the Fig 4.4-4.7 below.

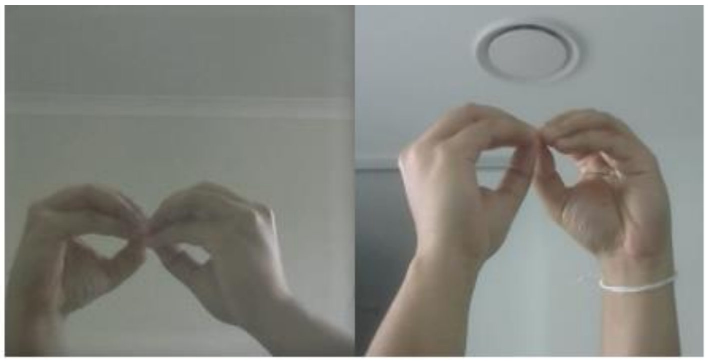


**Figure 4.4:Example for differing the lighting condition when capturing the gesture for Letter "B". (Extracted from the training dataset of this proposed NSL translating system).**

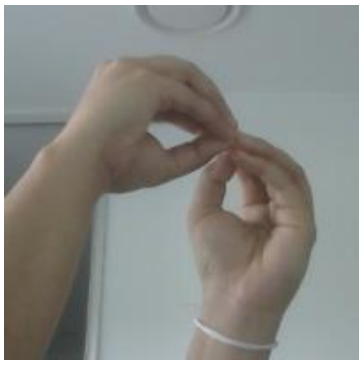


**Figure 4.5:Example for rotating the sign view when capturing the gesture for Letter "B". (Extracted from the training dataset of this proposed NSL translating system).**

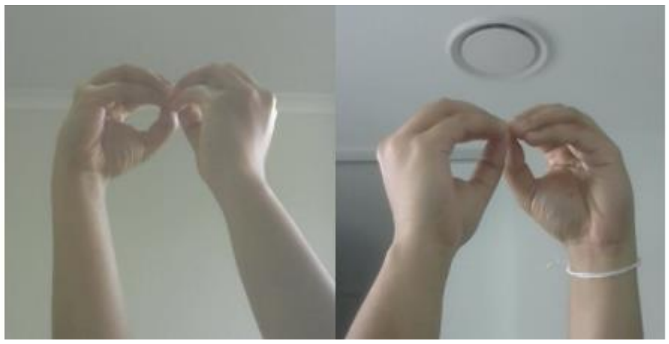


**Figure 4.6:Example for adding obstacles to the background when capturing the gesture for Letter "B". (Extract from the training dataset of this proposed NSL translating system).**

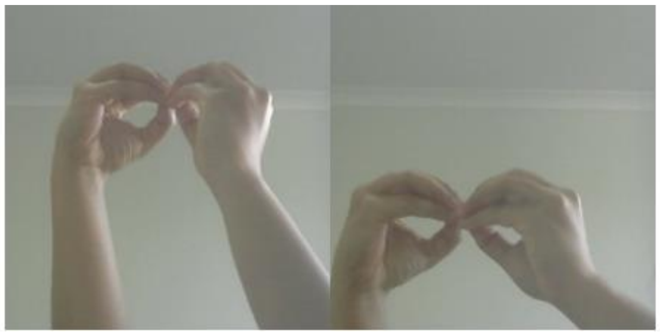


**Figure 4.7:Example for varying the shape of the sign when capturing the gesture for Letter "B". (Extract from the training dataset of this proposed NSL translating system).**

### 4.2.4 Data Pre-Processing and Training.

Both image preprocessing and training the Convolution Neural network (CNN) have been done by train.py, Python program. Preprocessing of images is essential as the images gathered from the camera might contain different formats. However, Image_capture.py, Python program is set to save the images in 200x200 pixels and in JPEG, color format. After collecting the required dataset, preprocessing of the images are conducted by the traing.py, Python program. In order to improve the results, that Python program run the raw data images through several distortions such as scaling, cropping and flipping. These are applied to make the model suitable for the variations expect in the actual world and to train the model to cope more effectively with the natural data.

Cropping of the input images is performed by placing a randomly positioned boundary box in the original image. The cropping parameters control the boundary box size compared to the input image. If the value is zero, the boundary box size will be the same as input. In that case, cropping is not performed. If the detected value is 50%, the cropping box would be the half size of height and width of the input. After performing that cropping task, train.py is programmed to conduct the image flipping and image scaling tasks. Apart from performing the above mention pre-processing data, train.py is coded to build a list for training data from the system and to analyze each sub-folder in the image directory while dividing them into a stable testing set, training set, and validation set. Then it has returned a detailed data structure explaining the image list for each of the labels with their paths.

After completing the image preprocessing, the initial step of the model training has been begun by passing it through the pre trained architecture, InceptionV3 which is trained at ImageNet images. This is one of the major training methods in Convolution Neural network (CNN), and it is called Transfer learning. Transfer learning is applied to an existing model and retrain the last few layers of the pre trained model to classify the new image sets. As the rapid development of the Artificial intelligence field, there is a wide range of image recognition models which come with millions of parameters. The pre-trained model selected for this research is one major model used in modern research. Training pre-trained models from their scratch requires a large amount of labeled data for training and requires high computing power (normally hundreds of great processing power/GPU-hours). This transfer learning technique can bypass the training by extracting a part of the model which has been trained already for the related task and reprocessing it in the new model.

A main advantage of applying this Transfer learning technique stems from Convolution Neural Network's (CNN) capability to learn both features and the weights related to each feature. Similar to other algorithms in Machine Learning, Convolution Neural Networks (CNN) seek to enhance some other objective functions, in particular the loss function. In this research, the loss function (SoftMax-based) given below has been used.

$$Loss = \frac{1}{N}\sum_{i=1}^{N} -\log\left(\frac{e^{f_{i,y_i}}}{\sum_{j=1}^{c} e^{f_{i,j}}}\right)$$

$$f_j(z) = \frac{e^{z_j}}{\sum_{k=1}^{c} e^{z_k}})$$

N= number of system Training examples.
C= number of classes.

$f_j(z)$ denotes the Softmax function. It uses z, the feature vector for a provided training example, and then squashes their values to [0,1] vector (summing the real numbers to 10. The Loss equation, which is given above, collects the mean loss of each training example ($x_i$) in order to provide the total Softmax loss. This Softmax centered classification head provides the output values similar to the probabilities of each of the New Zealand sign language alphabet letter.

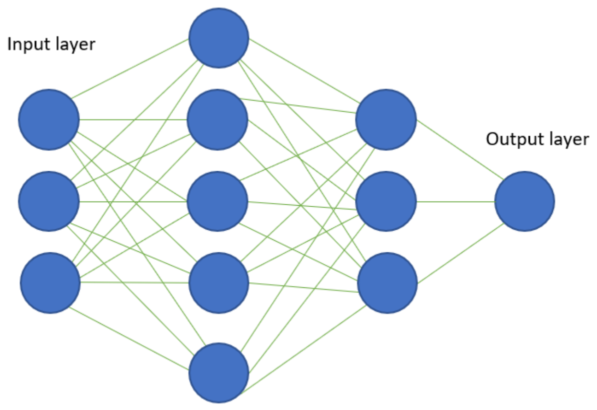


**Figure 4.8:standard convolution neural network model Architecture (green color lines represents the all the weights and biases training.**

As shown in the Fig 4.8, a fully trained convolution neural net receives input parameters in the very first layer and feeds the data forward sequentially while transforming them simultaneously until the 2nd to last layer of the network has built a high-level input representation which is capable to be transformed easily into a final net output. The complete model training involves the weight optimization and the bias terms which are used in every connection. These are labelled in green.

### 4.2.5 Bottleneck Training

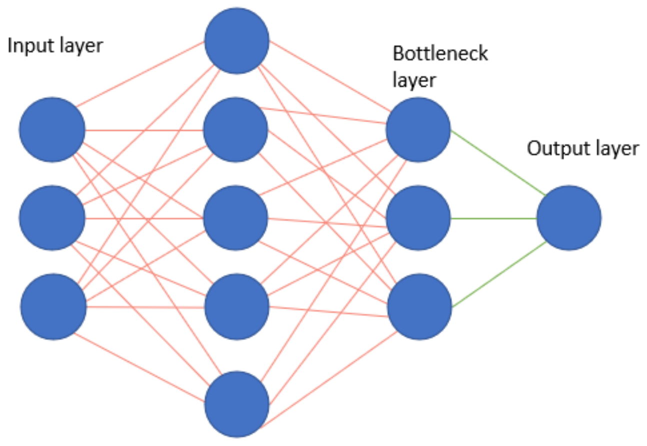


**Figure 4.9:Model Architecture for a neural network model in Transfer Learning.**

As shown in Fig 4.9, the layer just before the final layer is called bottleneck layer and that layer is responsible for pushing the required values of the regression model or the classification model's softmax probabilities to the final network layer.Red lines of the Fig 4.9 indicate the fixed biases and the weights while the green lines indicate the training of weights and biases of the final layer only. Bias is the representation of an additional neuron included with every pre-output layer, and it stores value "1" for every action and in fixed biases while fixing the weights. The training classifiers took more than 12 hours and passed through 2000 numbers of training steps. That training time will depend on the machine speed. Train.py program has been set to analyze all images which are on disk of the computer and then cache the relevant bottleneck values for every single image as the first phase of training the image.

The word, "Bottleneck" is an unofficial term used for introducing the penultimate layer before the output layer. Bottleneck is the layer where the actual classification of the new model has done. As per the TensorFlow hub, this is called "image feature vector". This bottleneck layer or has been trained to provide a set of output values which are sufficient for the classifier to classify all classes it is asked to identify. That suggests that layer has to be compact and meaningful image summary as it needs to compact sufficient information into the classifier to come up with a good choice from a small value set. This transfer learning method provides more benefits to the program. The main benefit is that training a pre trained dataset is faster than starting training from scratch.

### 4.2.6 Googlenet (Inceptionv3) Architecture Model

Train.py program is developed to train the bottleneck layer of the GoogleNet InceptionV3 model to recognize and identify other image classes. This open-source model, InceptionV3, contain more than 25 million parameters pre-trained with high quality hardware. Therefore, this InceptionV3 model has well-fitted parameters along with bottleneck layers and highly optimized input data representation. In order to link the InceptionV3 architecture model to our system, train.py is designed to download the InceptionV3 model to the proposing system through a data URL (uniform resource identifier) which offers the path to input data in external sources. As per the train.py program, which is attached to the appendix of this thesis, the bottleneck tensor size, model input width, model input height and model input depth are used as 2048, 299,299 and 3 respectively. The InceptionV3 model for this research is shown in the Fig 4.10 below (including parameters detailed above).

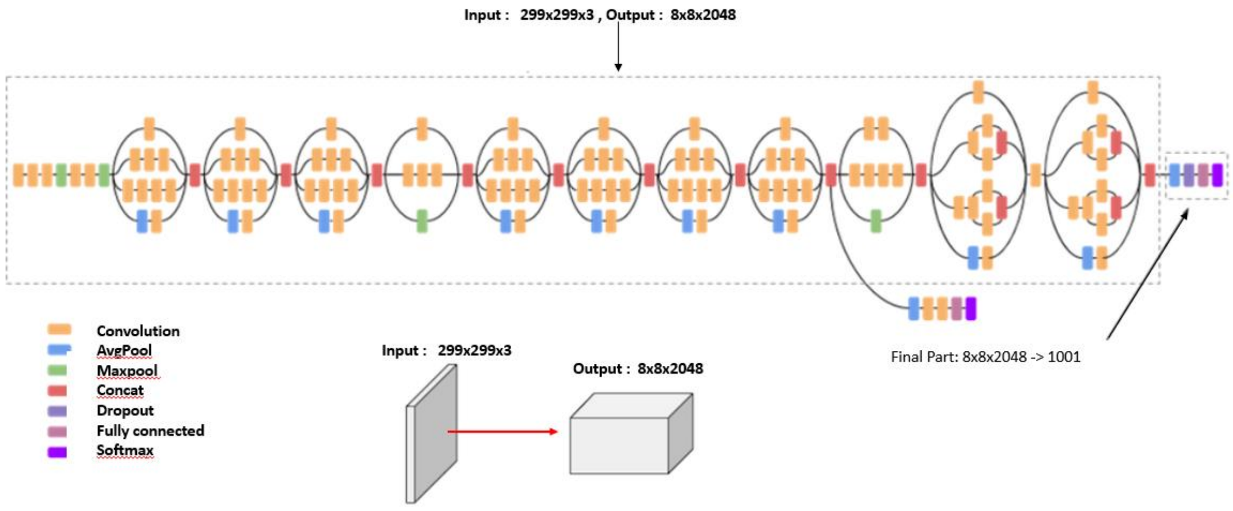
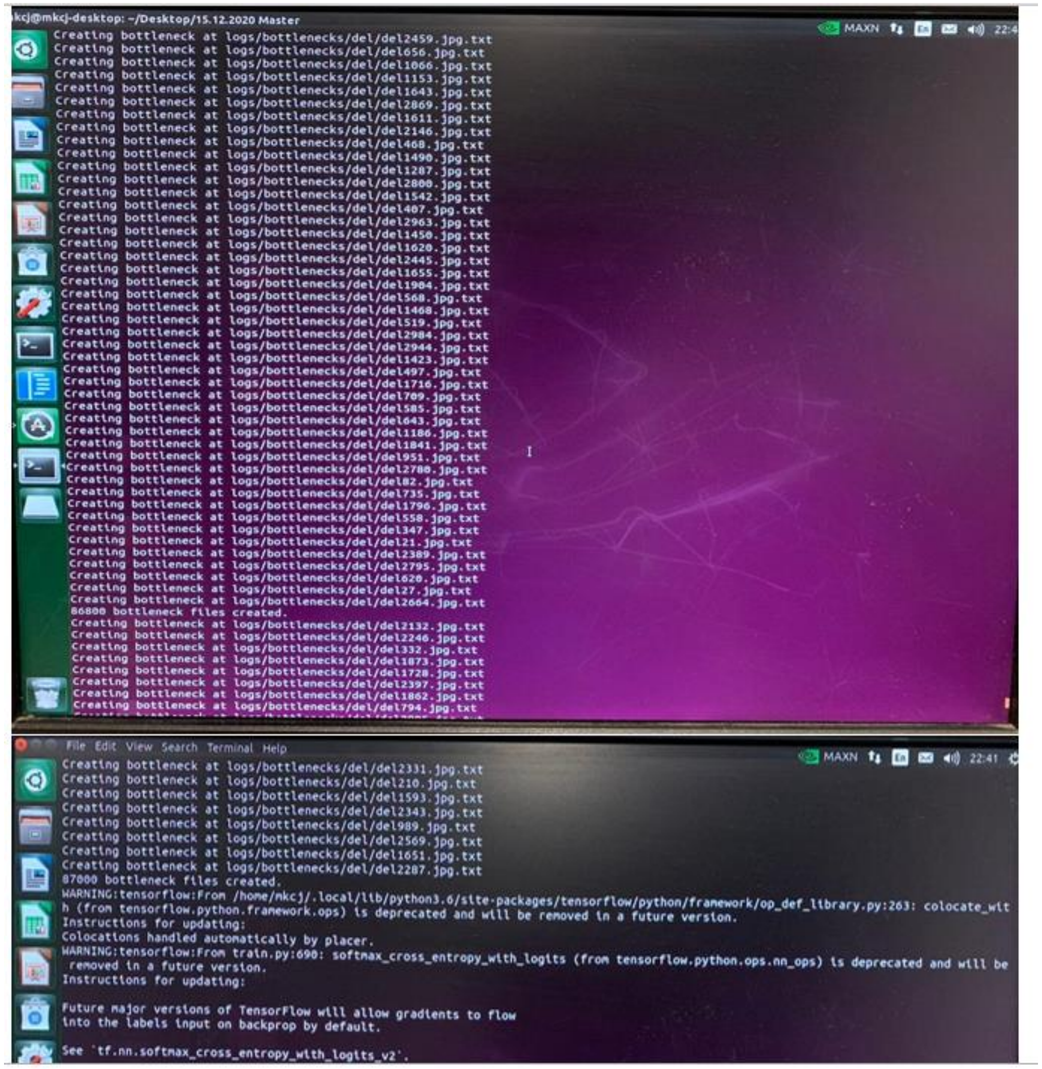


**Figure 4.10: Overview of the InceptionV3 model in the proposed research.**



```
kcj@mkcj-desktop: ~/Desktop/15.12.2020 Master
Creating bottleneck at logs/bottlenecks/del/del2459.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del656.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1066.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1153.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1643.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2869.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1611.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2146.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del468.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1490.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1287.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2800.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1542.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del407.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2963.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1450.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1620.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2445.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1655.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1984.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del568.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1468.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del519.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2984.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2944.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1423.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del497.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1716.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del709.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del585.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del643.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1186.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1841.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del951.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2780.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del82.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del735.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1796.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del558.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del347.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del21.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2389.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2795.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del620.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del27.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2664.jpg.txt
86800 bottleneck files created.
Creating bottleneck at logs/bottlenecks/del/del2132.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2246.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del332.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1873.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1728.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2397.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1862.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del794.jpg.txt
```

```
Creating bottleneck at logs/bottlenecks/del/del2331.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del210.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1593.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2343.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del989.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2569.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1651.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2287.jpg.txt
87000 bottleneck files created.
WARNING:tensorflow:From /home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/op_def_library.py:263: colocate_wit
h (from tensorflow.python.framework.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From train.py:690: softmax_cross_entropy_with_logits (from tensorflow.python.ops.nn_ops) is deprecated and will be
 removed in a future version.
Instructions for updating:

Future major versions of TensorFlow will allow gradients to flow
into the labels input on backprop by default.

See `tf.nn.softmax_cross_entropy_with_logits_v2`.
```

**Figure 4.11:Bottleneck file creation in training Terminal of the proposed reserch.**

The training of this system is based on the technique called, transfer learning of deep learning. That technique is applied to retrain the InceptionV3 pre-trained deep learning model. The features of every data image is extracted and then necessary elements are saved as bottlenecks in order to minimize the computational expenses. Those trained models are used to generate TensorFlow graphs that are, in turn, used in the model's recognition stage. Fig 4.11 illustrates the created bottlenecks for the Sign language translator training which is taken from the training terminal of the program. When conducting the training for this research, 2000 training steps and 100 numbers of batch sizes have been used. Therefore, altogether 87,000 bottlenecks have been created in the system.

### 4.2.7 Algorithm and CNN Model Training

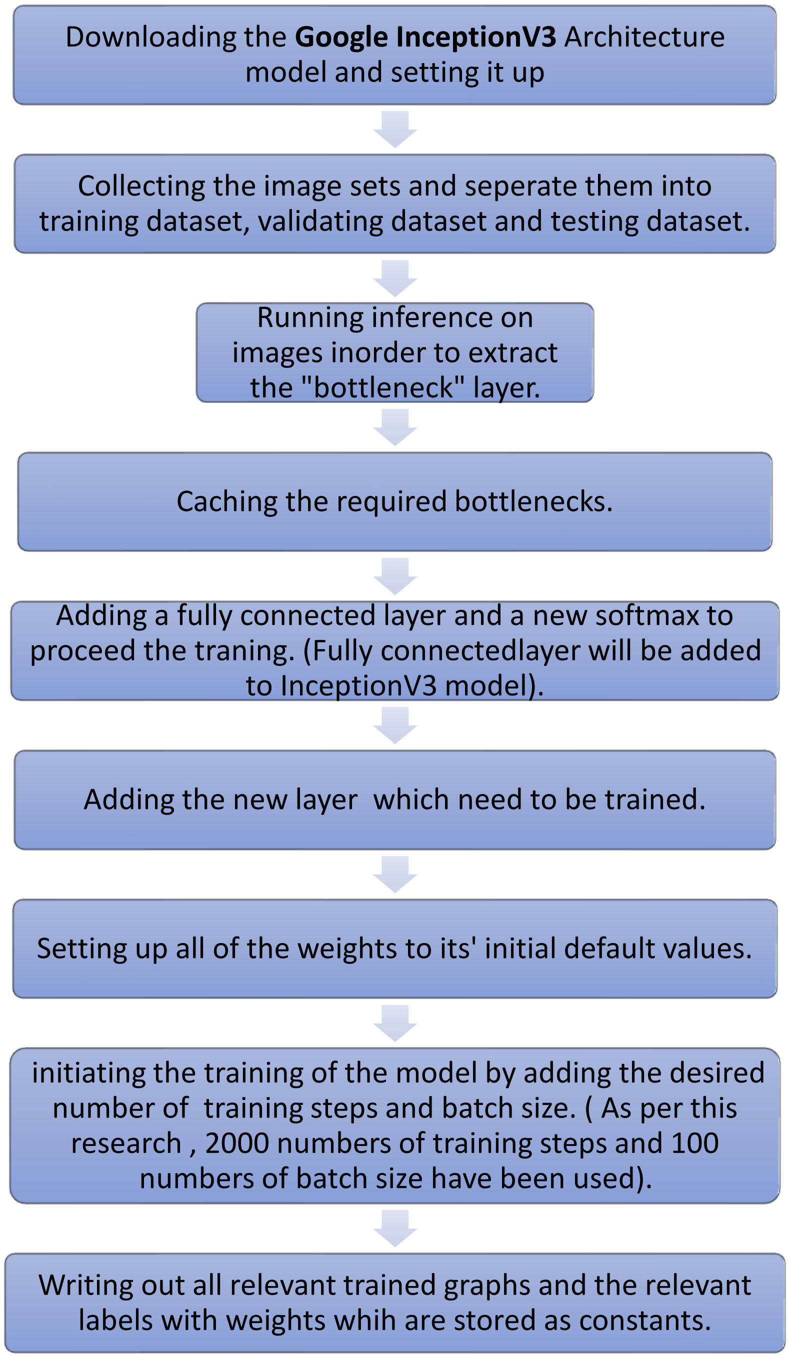

Downloading the **Google InceptionV3** Architecture model and setting it up
Collecting the image sets and seperate them into training dataset, validating dataset and testing dataset.
Running inference on images inorder to extract the "bottleneck" layer.
Caching the required bottlenecks.
Adding a fully connected layer and a new softmax to proceed the traning. (Fully connectedlayer will be added to InceptionV3 model).
Adding the new layer which need to be trained.
Setting up all of the weights to its' initial default values.
initiating the training of the model by adding the desired number of training steps and batch size. ( As per this research , 2000 numbers of training steps and 100 numbers of batch size have been used).
Writing out all relevant trained graphs and the relevant labels with weights whih are stored as constants.

**Figure 4.12: Algorithm for InceptionV3 model training for this proposed New Zealand sign language translator.**

Fig 4.12 represents the complete algorithm which is used to train the InceptionV3 model to obtain the system for translating New Zealand sign language and display it as text mode. As per this algorithm, after the bottlenecks are created, the actual training for the top layer in the network commences. Then the training terminal displays the step output series which are indicating validation accuracy, training accuracy and cross entropy. The validation accuracy denotes the precision of a randomly chosen image group from a different image set. Cross entropy is the loss function which shows how far the learning process has progressed. The overall objective of the training is to minimize the loss.

## 4.3 System design.

In this section, the experimental platform for the proposed New Zealand sign language translator is presented. Specifically, the hardware used and the process of installing all the essential software for the proposed system is discussed.

### 4.3.1 Hardware Selection.

The purpose of this classification is to translate New Zealand signs into text mode through the gesture interpretation of the signer. This includes translating 26 letters in English alphabet and 3 other custom symbols (delete, space, nothing). Most importantly, this proposed system can translate entire sentences and show it in the display (text mode). Table 4.1 below illustrates the specifications of the hardware used for the experiments in this thesis.

**Table 4.1: Used hardware specification for the proposed New Zealand sign language translating system.**

| Property | Specification |
|---|---|
| Graphical Processing Unit-GPU | NVIDIA Tegra X1 (NVGPU)/integrated |
| Processor | ARMv8 Processor rev 1(v8l) x 4 |
| OS type | 64 bit |
| Memory | 3.9GiB |
| Operating System | Ubuntu |
| Image capturing device | Logitech |

### 4.3.2 Graphical Processing Unit-GPU selection.

Generally, the Graphical Processing Unit-GPU consists of a wide range of processing cores. GPUs act as accelerators which are employed to perform fast matrix calculations in parallel. These kinds of devices are normally very affordable since the developments of the GPUs is inspired by the gaming industry [89].The performance of computer vision algorithm is based on not only in large dataset and deep learning technique but also on developed parallel computer architecture which allows sufficient training on multiple layers in neural networks [90]. In addition, a modern GPU come with powerful graphic engine and an extremely parallelized computing processor containing high memory bandwith and high throughput for huge parallel algorithms. GPUs normally operate remarkably well on multiplications which are essential for training in neural networks [91]. Therefore, learning the process can be accelerated dramatically by providing a higher memory bandwidth to the CPUs.

The reason why Graphics Processing Units are well suited for developing algorithms in deep neural network is the similarity between GPUs' image or data manipulation task and the mathematical core of neural network [92]. Considering their advantages, Graphics Processing Units have become the usual choice in hardware implementation. Both NVIDIA Jetson Nano developer kit and NVIDIA Jetson TX2 developer kit are the two of the most efficient computers that can perform image processing in deep neural network, mainly due to low processing time [93].

### 4.3.3 Jetson Nano Developer Kit

In this research, NVIDIA Jetson Nano devices have been used as they come with high computational cost when processing large scale data in algorithms, which applies to deep learning techniques.

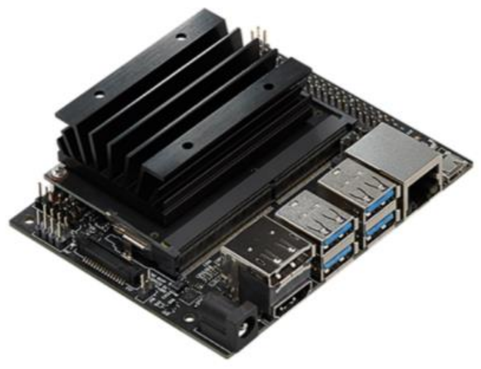


**Figure 4.13: NVIDIA Jetson Nano device.**

NVIDIA Jetson nano developer kit is most recent offering from the GPU manufacturer NVIDIA for Artificial Intelligence applications in the field of edge computing. This manufacturer offers different types of Jetson modules with unique capabilities. Each model comes with a computing system packaged like a plug-in element (System on module) [94]. Fig 4.14 illustrates the features included in NVIDIA Jetpack software development Kit (SDK). The Jetpack SDK contains libraries, OS images, APIs, documentation, samples and developer tools. Further NVIDIA Jetpack SDK offers following modules [89]:

- **TensorRT**- software development Kit for high-performance and high-level deep learning inference.
- **cuDNN**— Graphical Processing accelerated library for models in deep neural networks.
- **CUDA—CUDA** Toolkit offers a complete development environment for C++ and C developers developing GPU-accelerated applications.
- **Multimedia API**—Video decoding and encoding.
- **Developer Tools**—The toolkit contains debugging, Nsight Eclipse Edition, and profiling tools with Nsight Compute, along with a toolchain designed for cross-compiling applications.
- **Computer Vision**—A toolkit for vision processing and computer vision.

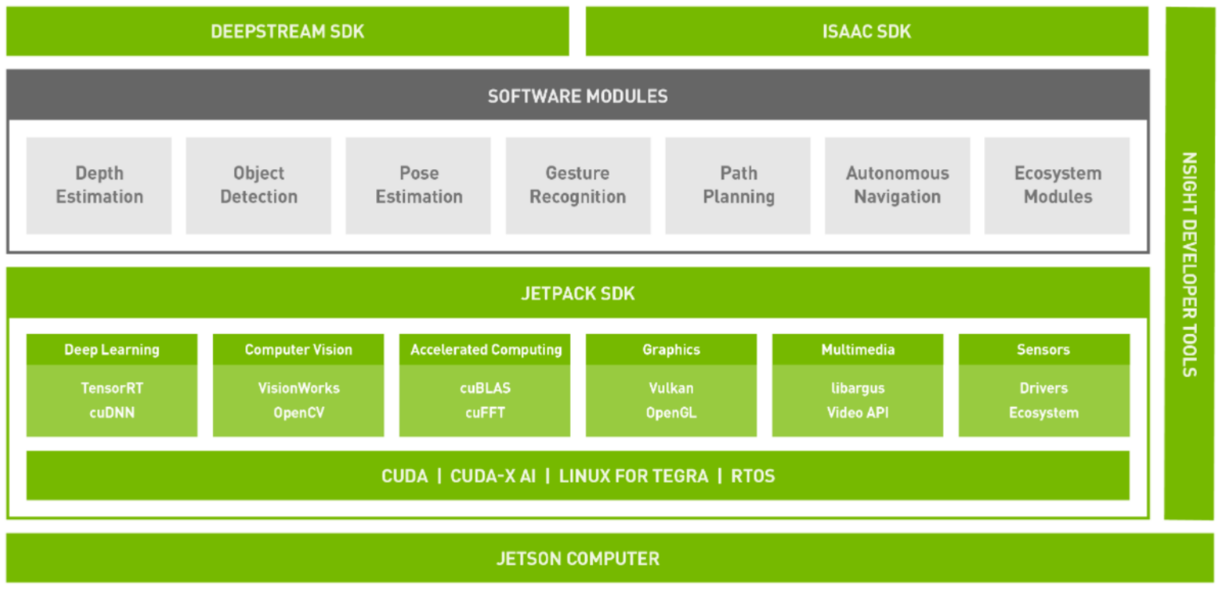


**Figure 4.14:NVIDIA Jetpack SDK [94]**

From this background, a system is developed to translate New Zealand sign language and display it in text mode. This research is completely in the area of image classification which is performed through NVIDIA Jetson Nano device [89], [95], [96]. The Jetson Nano device is a powerful small machine which is capable of executing various parallel neural network algorithms. Object detection, audio detection, speech or audio processing and segmentation can be implemented in addition to image classification by using this powerful device.

### 4.3.4 Software Selection

The training of this proposed New Sign language translating system has been conducted using Google's Tensorflow, one of the best and most popular Deep Learning Frameworks. Specifically, the **tf 1.13.1 GPU** is the compatible TensorFlow version for this system. This machine learning framework makes the process of training models, acquiring data, refining future results, and serving predictions less difficult and daunting. Table 4.2 illustrates the main softwares which have been used to implement the sign language translating system developed in this study.

**Table 4.2: Compatible Software version for the proposed system.**

| Name of the software | Relevant version |
|---|---|
| Ubuntu | 18.04 LTS |
| TensorFlow | 1.13.1 |
| OpenCv | 4.1.1 |
| Numpy | 1.19.2 |
| Matplotlib | 3.3.2 |

### 4.3.5 Summary

As discussed in the Methodology chapter and this system Design and Development chapter, the image capturing, and convolution neural network training of the proposed system has been carried out image_capture.py and train.py programs which are written in Python programming language. (refer the annex). In the next chapter, Experimental Results are discussed.

# Chapter 5 : **Experimental Results**

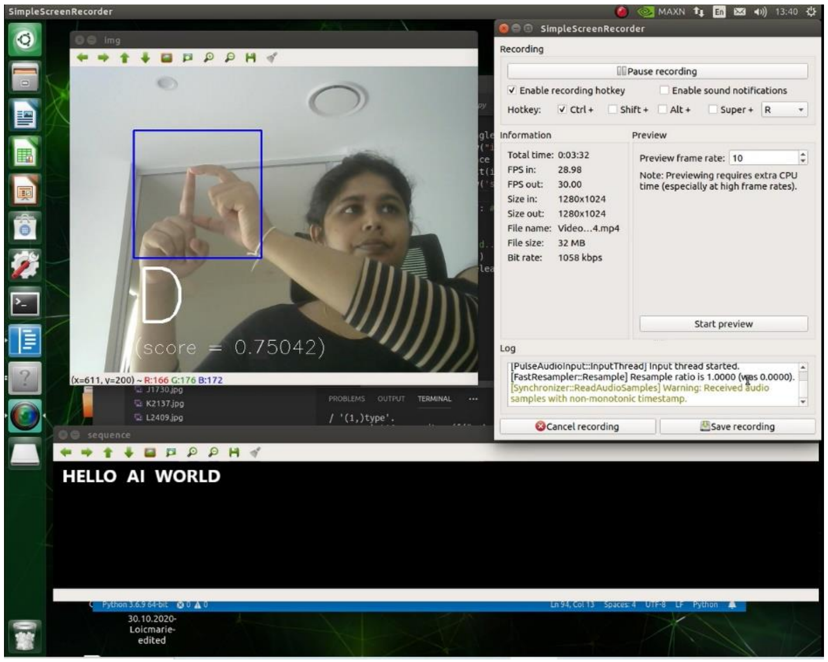
After introducing the architecture and the working principle of the proposed system in the above chapters, it is necessary to convey that how far this research has been successful. Therefore, the purpose of this chapter is to bring up the evidence which fulfill the research goal. For that, as the initial step, research outcome of this study will be presented by exhibiting the performance of the implemented system in real time. Thereafter, research goals will be addressed by presenting the Key performance indicators for the implanted system.

## 5.1 Introduction

The research aim of this study was to find out whether translating New Zealand sign language into text format by training Convolution Neural Network (CNN) with human hand gestures could be successfully accomplished. After training the system through transfer learning technique Convolution Neural Network (CNN), the system was tested in order to evaluate the performance of the study. In the following section, evidence for the performance of the experiment is presented and later analyzed.

### 5.1.1 Experimental Results

Neural network training for this study has been conducted with the learning rate of 0.01 in Code-OSS, open-source software. After successfully completing the training of the system, Recognize_gesture.py program has been initiated to test the trained system. Some of the results of the testing are displayed in Fig 5.1 below.



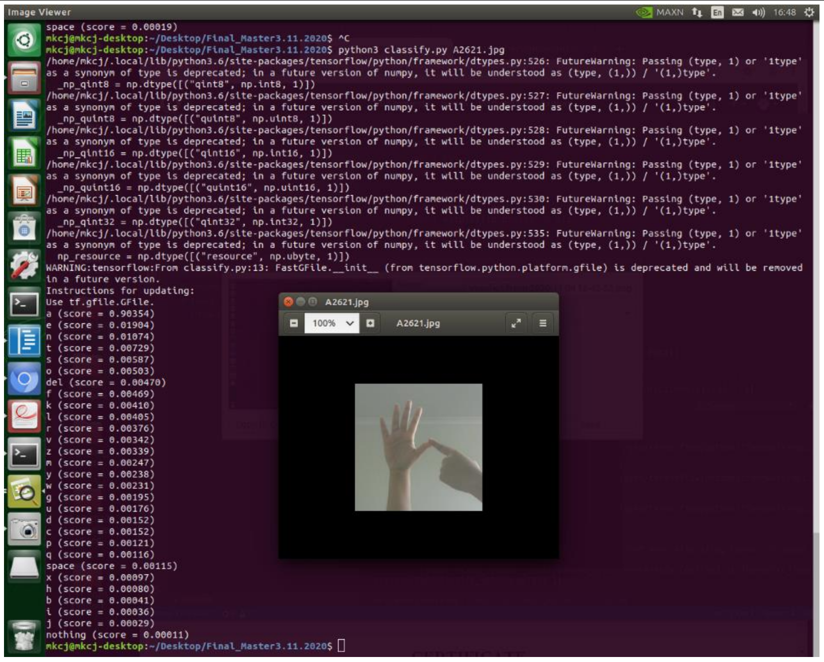
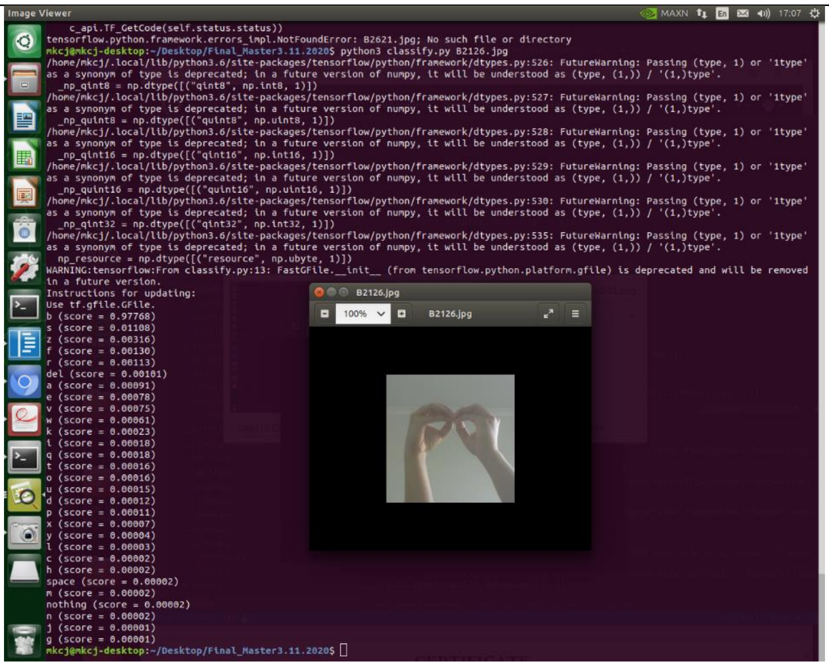
```
Total time: 0:03:32
FPS in:     28.98
FPS out:    30.00
Size in:    1280x1024
Size out:   1280x1024
File name:  Video...4.mp4
File size:  32 MB
Bit rate:   1058 kbps

Preview frame rate: 10
Note: Previewing requires extra CPU time (especially at high frame rates).

Log
[PulseAudioInput::InputThread] Input thread started.
[FastResampler::Resample] Resample ratio is 1.0000 (was 0.0000).
[Synchronizer::ReadAudioSamples] Warning: Received audio samples with non-monotonic timestamp.
```

**Figure 5.1: Demonstrating translation through sign language translator.**

Most of the research that has been conducted so far has been confined to translating only one sign and to classify it as a word. Apart from creating a translator for a dual-handed sign language, the other novelty of this research is to present the gesture prediction that can be expanded to display sentences as well. The recognition of the gesture is based on capturing a picture of the relevant sign and then feeding it to the deep learning neural classifier model. Next, the best-fitted result is generated including their scores. In other words, with the help of this trained network, the signer can perform their signing in front of the camera, which is connected with the system, and then the system can translate the performed signs into text mode in real time. One example for translating the sign for “HELLO AI WORLD” is displayed in the Fig 5.1.

### 5.1.2 Image Classification by using an image as the input.

In order to obtain classifying scores of the gesture signs for each 26 numbers of English Alphabet and three other symbols ("Delete", "Nothing", "Space"), the Python program, score.py has been run in Code-OSS open-source software. Fig 5.2 illustrates some of the collected scores for symbol "A" and "B". All the collected scores for each 29 symbols can be found below from the Fig 5.3.

| | |
|---|---|
| Actual Sign - "A"<br>Predicted Sign- "A"<br>Score - 0.90354 |  |
| Actual Sign - "B"<br>Predicted Sign- "B"<br>Score - 0.97768 |  |

```
space (score = 0.00019)
mkcj@mkcj-desktop:~/Desktop/Final_Master3.11.2020$ ^C
mkcj@mkcj-desktop:~/Desktop/Final_Master3.11.2020$ python3 classify.py A2621.jpg
/home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:526: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
/home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:527: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
/home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:528: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
/home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:529: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
/home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:530: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
/home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:535: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
WARNING:tensorflow:From classify.py:13: FastGFile.__init__ (from tensorflow.python.platform.gfile) is deprecated and will be removed
in a future version.
Instructions for updating:
Use tf.gfile.GFile.
a (score = 0.90354)
e (score = 0.01904)
n (score = 0.01074)
t (score = 0.00729)
s (score = 0.00587)
o (score = 0.00503)
del (score = 0.00470)
f (score = 0.00469)
k (score = 0.00410)
l (score = 0.00405)
r (score = 0.00376)
v (score = 0.00342)
z (score = 0.00339)
m (score = 0.00247)
y (score = 0.00238)
w (score = 0.00231)
g (score = 0.00195)
u (score = 0.00176)
d (score = 0.00152)
c (score = 0.00152)
p (score = 0.00121)
q (score = 0.00116)
space (score = 0.00115)
x (score = 0.00097)
h (score = 0.00080)
b (score = 0.00041)
i (score = 0.00036)
j (score = 0.00029)
nothing (score = 0.00011)
mkcj@mkcj-desktop:~/Desktop/Final_Master3.11.2020$ 
```

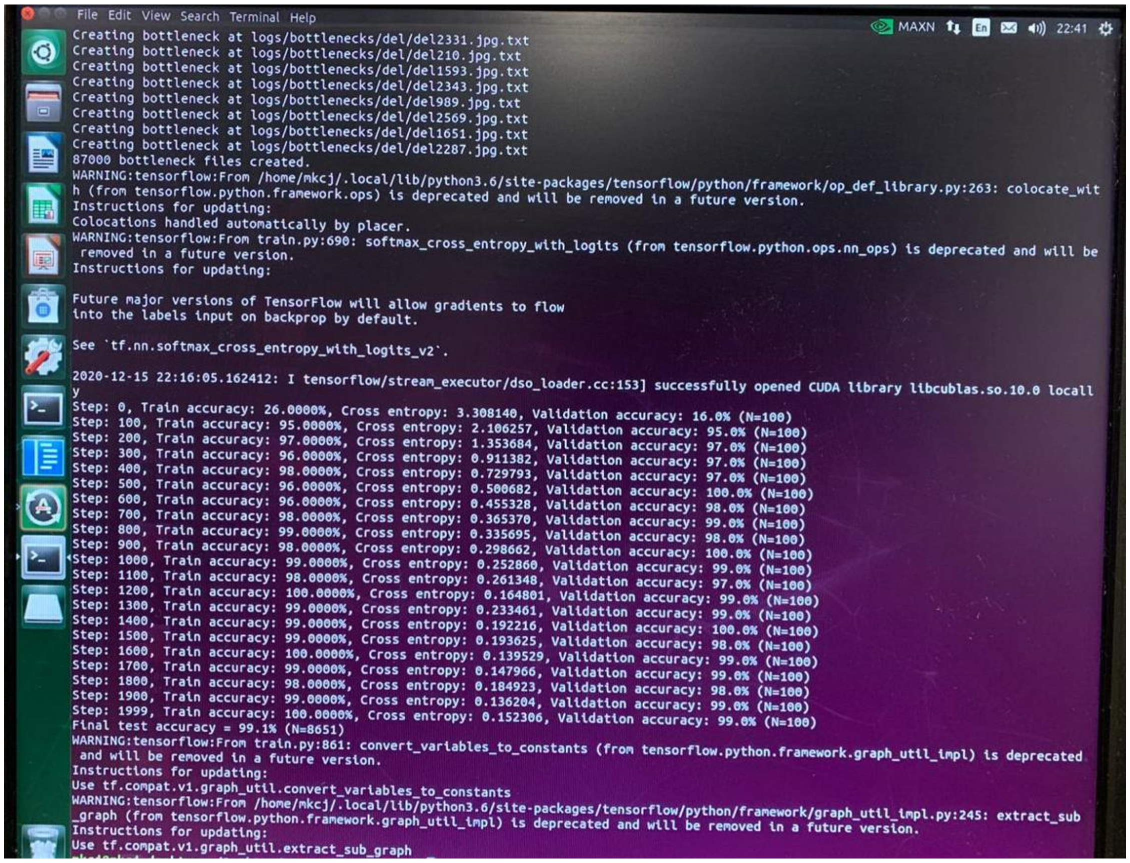
```
    c_api.TF_GetCode(self.status.status))
tensorflow.python.framework.errors_impl.NotFoundError: B2621.jpg; No such file or directory
mkcj@mkcj-desktop:~/Desktop/Final_Master3.11.2020$ python3 classify.py B2126.jpg
/home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:526: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
/home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:527: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
/home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:528: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
/home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:529: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
/home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:530: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
/home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/dtypes.py:535: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
WARNING:tensorflow:From classify.py:13: FastGFile.__init__ (from tensorflow.python.platform.gfile) is deprecated and will be removed
in a future version.
Instructions for updating:
Use tf.gfile.GFile.
b (score = 0.97768)
s (score = 0.01108)
z (score = 0.00316)
f (score = 0.00130)
r (score = 0.00113)
del (score = 0.00101)
a (score = 0.00091)
e (score = 0.00078)
v (score = 0.00075)
w (score = 0.00061)
k (score = 0.00023)
i (score = 0.00018)
q (score = 0.00018)
t (score = 0.00016)
o (score = 0.00016)
u (score = 0.00015)
d (score = 0.00012)
p (score = 0.00011)
x (score = 0.00007)
y (score = 0.00004)
l (score = 0.00003)
c (score = 0.00002)
h (score = 0.00002)
space (score = 0.00002)
m (score = 0.00002)
nothing (score = 0.00002)
n (score = 0.00002)
j (score = 0.00001)
g (score = 0.00001)
mkcj@mkcj-desktop:~/Desktop/Final_Master3.11.2020$ 
```

**Figure 5.2:Examples for Classifying scores in Code-OSS.**

**Table 5.1 Obtained Classification scores for each Letter and symbol.**

| | Name of the Letter or Symbol | Obtained Score |
|---|---|---|
| 1. | Letter 'A' | 0.90354 |
| 2. | Letter 'B' | 0.97768 |
| 3. | Letter 'C' | 0.76610 |
| 4. | Letter 'D' | 0.99358 |
| 5. | Letter 'E' | 0.92208 |
| 6. | Letter 'F' | 0.95220 |
| 7. | Letter 'G' | 0.96902 |
| 8. | Letter 'H' | 0.95675 |
| 9. | Letter 'I' | 0.97657 |
| 10. | Letter 'J' | 0.99086 |
| 11. | Letter 'K' | 0.93127 |
| 12. | Letter 'L' | 0.99502 |
| 13. | Letter 'M' | 0.98151 |
| 14. | Letter 'N' | 0.86590 |
| 15. | Letter 'O' | 0.90021 |
| 16. | Letter 'P' | 0.93939 |
| 17. | Letter 'Q' | 0.98046 |
| 18. | Letter 'R' | 0.97461 |
| 19. | Letter 'S' | 0.87284 |
| 20. | Letter 'T' | 0.86395 |
| 21. | Letter 'U' | 0.81841 |
| 22. | Letter 'V' | 0.93872 |
| 23. | Letter 'W' | 0.87002 |
| 24. | Letter 'X' | 0.91299 |
| 25. | Letter 'Y' | 0.88299 |
| 26. | Letter 'Z' | 0.97700 |
| 27. | Letter 'space' | 0.98462 |
| 28. | Letter 'nothing' | 0.98006 |
| 29. | Letter 'del' | 0.97006 |

As described in the Methodology and Development chapters, the training of this proposed sign language translating system is based on transfer learning technique in Deep Learning, and the GoolgleNet (INceptionV3) model has been used as the pre trained model to conduct the retaining of the system. In the training process, the features from each input data image are extracted, and the necessary elements are saved and stored as Bottlenecks in order to save the computational expenses. Thereafter, those trained models are used to generate the overall accuracy of the system, training accuracy and validation accuracy graphs, and other identified KPI values which are used to identify the performance level of the system. The output result of the train.py that has been displayed in Code-OSS is mentioned in Fig 5.4.



```
Creating bottleneck at logs/bottlenecks/del/del2331.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del210.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1593.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2343.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del989.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2569.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del1651.jpg.txt
Creating bottleneck at logs/bottlenecks/del/del2287.jpg.txt
87000 bottleneck files created.
WARNING:tensorflow:From /home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/op_def_library.py:263: colocate_wit
h (from tensorflow.python.framework.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From train.py:690: softmax_cross_entropy_with_logits (from tensorflow.python.ops.nn_ops) is deprecated and will be
 removed in a future version.
Instructions for updating:

Future major versions of TensorFlow will allow gradients to flow
into the labels input on backprop by default.

See `tf.nn.softmax_cross_entropy_with_logits_v2`.

2020-12-15 22:16:05.162412: I tensorflow/stream_executor/dso_loader.cc:153] successfully opened CUDA library libcublas.so.10.0 locall
y
Step: 0, Train accuracy: 26.0000%, Cross entropy: 3.308140, Validation accuracy: 16.0% (N=100)
Step: 100, Train accuracy: 95.0000%, Cross entropy: 2.106257, Validation accuracy: 95.0% (N=100)
Step: 200, Train accuracy: 97.0000%, Cross entropy: 1.353684, Validation accuracy: 97.0% (N=100)
Step: 300, Train accuracy: 96.0000%, Cross entropy: 0.911382, Validation accuracy: 97.0% (N=100)
Step: 400, Train accuracy: 98.0000%, Cross entropy: 0.729793, Validation accuracy: 97.0% (N=100)
Step: 500, Train accuracy: 96.0000%, Cross entropy: 0.500682, Validation accuracy: 100.0% (N=100)
Step: 600, Train accuracy: 96.0000%, Cross entropy: 0.455328, Validation accuracy: 98.0% (N=100)
Step: 700, Train accuracy: 98.0000%, Cross entropy: 0.365370, Validation accuracy: 99.0% (N=100)
Step: 800, Train accuracy: 99.0000%, Cross entropy: 0.335695, Validation accuracy: 98.0% (N=100)
Step: 900, Train accuracy: 98.0000%, Cross entropy: 0.298662, Validation accuracy: 100.0% (N=100)
Step: 1000, Train accuracy: 99.0000%, Cross entropy: 0.252860, Validation accuracy: 99.0% (N=100)
Step: 1100, Train accuracy: 98.0000%, Cross entropy: 0.261348, Validation accuracy: 97.0% (N=100)
Step: 1200, Train accuracy: 100.0000%, Cross entropy: 0.164801, Validation accuracy: 99.0% (N=100)
Step: 1300, Train accuracy: 99.0000%, Cross entropy: 0.233461, Validation accuracy: 99.0% (N=100)
Step: 1400, Train accuracy: 99.0000%, Cross entropy: 0.192216, Validation accuracy: 100.0% (N=100)
Step: 1500, Train accuracy: 99.0000%, Cross entropy: 0.193625, Validation accuracy: 98.0% (N=100)
Step: 1600, Train accuracy: 100.0000%, Cross entropy: 0.139529, Validation accuracy: 99.0% (N=100)
Step: 1700, Train accuracy: 99.0000%, Cross entropy: 0.147966, Validation accuracy: 99.0% (N=100)
Step: 1800, Train accuracy: 98.0000%, Cross entropy: 0.184923, Validation accuracy: 98.0% (N=100)
Step: 1900, Train accuracy: 99.0000%, Cross entropy: 0.136204, Validation accuracy: 99.0% (N=100)
Step: 1999, Train accuracy: 100.0000%, Cross entropy: 0.152306, Validation accuracy: 99.0% (N=100)
Final test accuracy = 99.1% (N=8651)
WARNING:tensorflow:From train.py:861: convert_variables_to_constants (from tensorflow.python.framework.graph_util_impl) is deprecated
 and will be removed in a future version.
Instructions for updating:
Use tf.compat.v1.graph_util.convert_variables_to_constants
WARNING:tensorflow:From /home/mkcj/.local/lib/python3.6/site-packages/tensorflow/python/framework/graph_util_impl.py:245: extract_sub
_graph (from tensorflow.python.framework.graph_util_impl) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.compat.v1.graph_util.extract_sub_graph
```

**Figure 5.3: Displaying Training Accuracy in Code-OSS.**

As displayed in Fig 5.4, this proposed New Zealand Sign Language translator using convolution neural Network (CNN) of this thesis is able to obtain 99.1 % final test accuracy. This test accuracy for the CNN network has been obtained using a batch size of 100, 2000 training steps and a 0.01 learning rate. When training a model in machine learning, one of the major requirements is to avoid the model overfitting. In order to find out a model's overfitting status, data scientists use a special technique known as cross validation where the data is spitted into two sections: training dataset and validation dataset. That training dataset is used to conduct the training of the model while the validation dataset is used to conduct the evaluation of the model's performance.

Metrics of the training dataset allows to see the progress of the model in terms of training of the model while metrics of the model in the validation dataset allow them to obtain a measure of the model's quality or the model's output. Using this information, the overall test accuracy, training accuracy, validation accuracy and cross entropy at each step have been identified for this study. Each of these values can be obtained from the same figure, Fig 5.4.

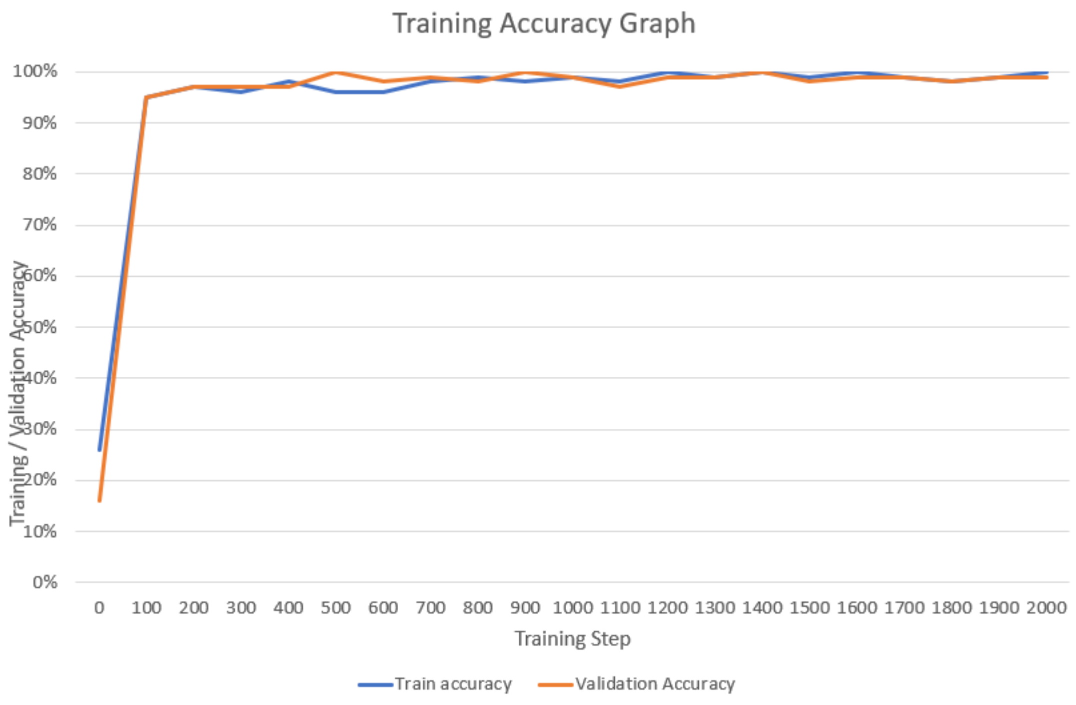


**Figure 5.4: Graph for Training and Validation Accuracy of the NSL Translating system.**

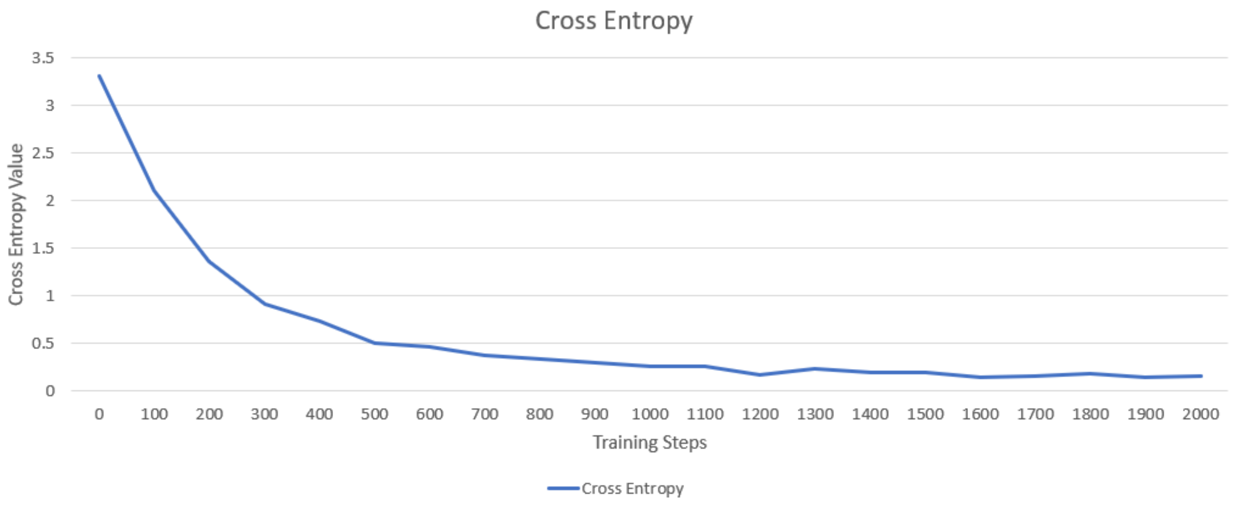


**Figure 5.5:Graph for Cross Entropy Values of the NSL Translating system.**

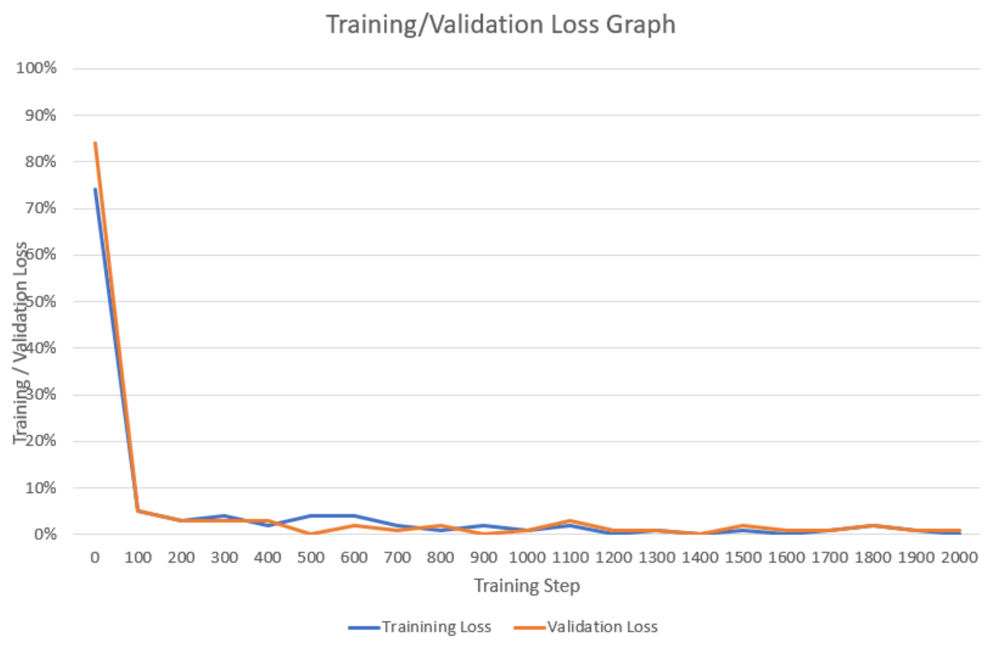
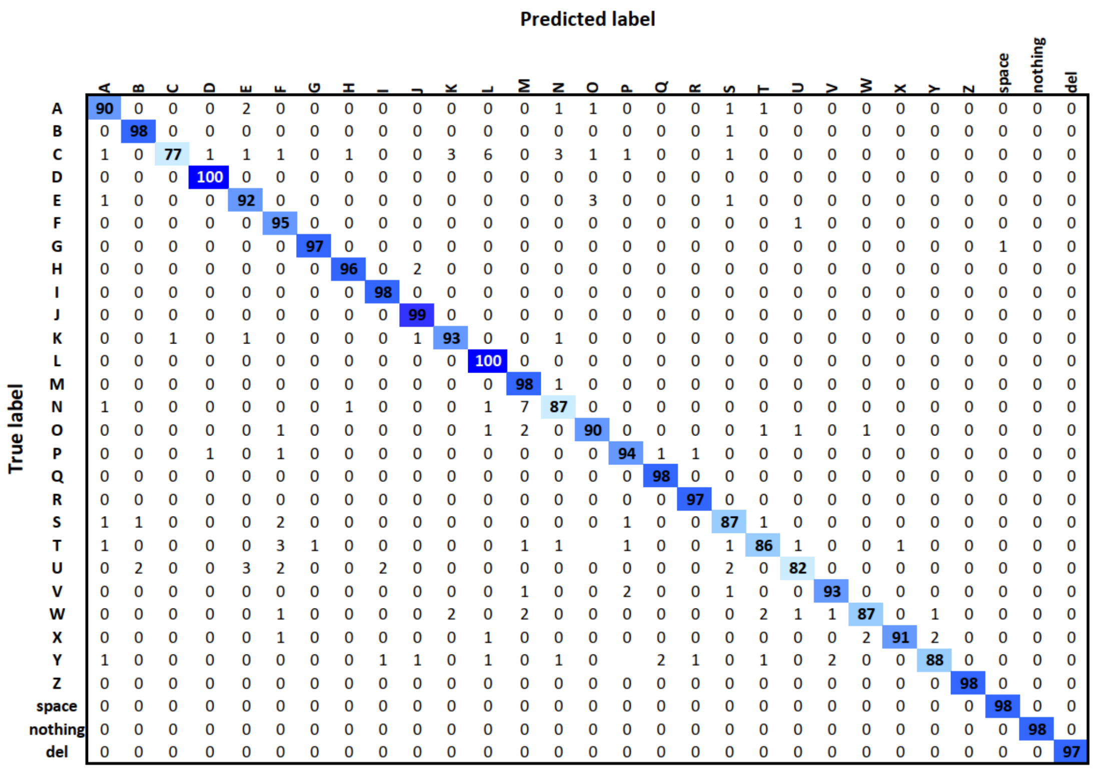


**Figure 5.6:Graph for Training and Validation Loss of the NSL Translating system.**


Predicted label
True label

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | space | nothing | del |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 90 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| B | 0 | 98 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C | 1 | 0 | 77 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 3 | 6 | 0 | 3 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D | 0 | 0 | 0 | 100 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| E | 1 | 0 | 0 | 0 | 92 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| F | 0 | 0 | 0 | 0 | 0 | 95 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| G | 0 | 0 | 0 | 0 | 0 | 0 | 97 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| H | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 96 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| I | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 98 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| J | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 99 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| K | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 93 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| L | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 100 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| M | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 98 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| N | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 7 | 87 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| O | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 0 | 90 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| P | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 94 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 98 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 97 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| S | 1 | 1 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 87 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| T | 1 | 0 | 0 | 0 | 0 | 3 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |  | 1 | 0 | 0 | 1 | 86 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| U | 0 | 2 | 0 | 0 | 3 | 2 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 82 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| V | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 2 | 0 | 0 | 1 | 0 | 0 | 93 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 2 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 1 | 1 | 87 | 0 | 1 | 0 | 0 | 0 | 0 |
| X | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 91 | 2 | 0 | 0 | 0 | 0 |
| Y | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 |  | 2 | 1 | 0 | 1 | 0 | 2 | 0 | 0 | 88 | 0 | 0 | 0 | 0 |
| Z | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 98 | 0 | 0 | 0 |
| space | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 98 | 0 | 0 |
| nothing | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 98 | 0 |
| del | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 97 |

**Figure 5.7: Confusion Matrix of the NSL Translating system.**

| Letter/symbol | Recall | Precision | F1 Score |
|---|---|---|---|
| A | 93.8 | 95.7 | 94.7 |
| B | 97.0 | 99.0 | 98.0 |
| C | 98.7 | 79.4 | 88.0 |
| D | 98.0 | 99.0 | 98.5 |
| E | 94.8 | 94.8 | 94.8 |
| F | 88.0 | 99.0 | 93.1 |
| G | 99.0 | 99.0 | 99.0 |
| H | 99.0 | 98.0 | 98.5 |
| I | 97.0 | 100.0 | 98.5 |
| J | 97.1 | 100.0 | 98.5 |
| K | 94.9 | 96.9 | 95.9 |
| L | 90.9 | 100.0 | 95.2 |
| M | 88.3 | 99.0 | 93.3 |
| N | 91.6 | 90.6 | 91.1 |
| O | 94.7 | 92.8 | 93.8 |
| P | 94.9 | 95.9 | 95.4 |
| Q | 97.0 | 100.0 | 98.5 |
| R | 98.0 | 100.0 | 99.0 |
| S | 97.6 | 93.5 | 95.5 |
| T | 93.5 | 86.9 | 90.1 |
| U | 95.3 | 90.1 | 92.7 |
| V | 96.9 | 95.9 | 96.4 |
| W | 96.7 | 89.7 | 93.0 |
| X | 98.9 | 93.8 | 96.3 |
| Y | 96.7 | 88.9 | 92.6 |
| Z | 100.0 | 100.0 | 100.0 |
| space | 99.0 | 100.0 | 99.5 |
| nothing | 100.0 | 100.0 | 100.0 |
| del | 100.0 | 100.0 | 100.0 |

**Table 5.2: Obtained Recall, Precision and F1 score values of the NSL Translating system.**

Fig 5.5 illustrates the training & validation accuracy curves and Fig 5.6 demonstrates the cross-entropy graph. Training & Validation loss is given in Fig 5.7. The calculated Confusion Matrix for the developed system is shown in Fig 5.8. Another evaluating matrix, the F1 score, is displayed in Table 5.1 for each trained letter and symbol. Another accuracy indicator of Convolution neural network model is the kappa co-efficient, which for this trained model is 0. 957.The kappa co-efficient measures the agreement between truth values and classification. It can be a value from 0 to 1. If the kappa co-efficient value equals to 0, there is not any agreement between truth values (truth image) and classified images. If the Kappa value equals to 1, truth values and classified images are totally identical. Therefore, the higher the kappa value, the more precise the classification.

# Chapter 6 : Discussion

In this discussion chapter, the focus is on identifying how research findings are related to research problems. This will provide more clearer view of the study and how far proposed hypothesis could be verified by interpreting the results. The second part of the study limitations and challenges encountered are discussed. The final section of the chapter discusses discussing recommendations to overcome challenges faced during the research study.

## 6.1 Discussion

New Zealand sign language gestures are based on dual-hand operations and are complicated to interpret compared to single handed sign languages. This research focuses on testing and verifying the hypothesis “**Dual-handed sign Languages, especially New Zealand sign language can be interpreted into text mode based on human hand gestures”.** Findings have been obtained using AI based image processing system developed using CNN architecture and Jetson Nano kit as hardware. The Python programming language has been used to program the system.

After completing the InceptionV3 model retraining, 99.1% of test accuracy has been obtained (result is displayed in Fig 5.4). Test accuracy is presented using sensitivity and specificity of the trained system. Most importantly, the test method can be justified when obtaining a higher test accuracy. That means the system is fully capable of performing the assigned task it is trained for. This test accuracy directs the user to realize how reliable are the obtained results. Having a higher test score value is necessary to validate the hypothesis of the research.

In real time application, it is required to achieve a higher classifying score to evaluate the performance of the classification ability of the system. Each letter or a symbol has achieved its individual classification scores based on metrics and algorithms used. In the experimental stage of this research, a classifying score for each letter from A-Z and three other symbols have been identified using gesture images as inputs. Those identified scores for 29 letters and symbols are varied between 98% and 76% (Refer Table 5.3). However, the classifying scores are not enough to measure the model’s performance. Therefore, apart from these measurements, some other Key Performances Indicators have been evaluated in this study.

Evaluating and measuring different metrics is useful for judging the model performance of a CNN system. Therefore, a set of KPIs, such as confusion matrix, F1 score, training and validation accuracy curves, training and validation loss curves have been calculated and presented in chapter 5 Experimental Results. As per Fig 5.8, confusion matrix which is a way to summarize the performance for a classification algorithm, has been presented for this research. By adhering to the confusion matrix table Recall scores and Precision scores have been identified individually to obtain the F1 score which is another technique to evaluate the model performance. The F1 score of this research has varied between 88% and 100% (calculated for each 26 letters and 3 symbols). The dynamics and the shape of learning curve is useful for evaluating the behavior of any machine learning algorithm. As per the Fig 5.7, which presents the Training and validation loss curve, it is clear that the learning curve reveals a good fit for the following reasons:

- The plot for training loss has been lowered to the stability point.
- The plot for validation loss has been lowered to the stability point and maintained a lower gap with training curve. [97]

The training and validation accuracy learning curves for this research, which are shown in Fig 5.5, display a good fit learning process of the trained system.

This result demonstrates higher accuracy values in the overall performance of the model and the individual performance of each letter or symbol. Those higher accuracy values are able to verify the research hypothesis of this study. That means, it is proven that gesture signs of dual-hand New Zealand sign language, can be translated into text format and displayed as entire sentences with high accuracy.

## 6.2 Limitations and Challenges Along with Their Acknowledgments.

This research study might contain some limitations. This research has been surveyed and analyzed research articles published mostly between 2002-2020, using word phrases such as “Sign Language translator” and “Image recognition”. Research articles which may have been done for dual-handed sign languages and research articles which are focused on dynamic signs could not be extracted. Apart from that, this research has limited the search of research articles and publications to online database. There could be other printed academic articles which may have been able to offer more comprehensive details related to sign language translators in real time; but due to the Covid-19 pandemic, there was limited access to the library. Although conducting research on Sign language recognition began several years back, it is still in its infancy, because no such system has been able to deploy on a scale that is capable of interpreting signs of large vocabularies in real time. The difficulties to implementing a strong system still exist including database scalability, hand occlusion, high computational cost, and illumination environment changes.

Apart from the above-mentioned limitations, there has been plenty of challenges when conducting this research study. The main one was to identify the suitable and compatible software to the Jetson Nano developer kit. It is essential to install an efficient framework to perform effective computations at higher speed. TensorFlow with GPU (Graphics Processor Unit) was the ideal solution for that but, finding the compatible TensorFlow version with Jetson Nano -Jetpack which includes OS images specially for JETSON products with APIs and libraries, developer tools samples and documents, proved to be challenging. As per the official website of NIVIDIA developers, TensorFlow GPU can work with only the older versions of the Jetpack pack [98]. This is a crucial selection as the final performance of the model keenly relies on TensorFlow performance. As an example, the TensorFlow GPU version can complete a training overnight while TensorFlow CPU might take weeks to do the same training. The modern jetpack versions are only compatible with TensorFlow CPU versions. The best versions of TensorFlow and jetpack for this research model have been selected after doing couple of trial trainings. TensorFlow-GPU version 1.13.1 and Jetson Nano-Jetpack version 4.2.2 have been selected and installed into the system. Even though setting up the TensorFlow GPU was somewhat complicated, the overall performance obtained was well worth it. In this specific study, TensorFlow-GPU convolution neural network (CNN) training is much faster than when using TensorFlow-CPU. For this research study it took only around 12 hours to complete the training using TensorFlow-GPU 1.13.1.

Moreover, even though TensorFlow GPU is the best performance framework, GPU training requires huge memory of which available memory is the main limiting factor. In order to overcome that issue, different solutions have been analyzed. Among them, adding an additional swap memory has been the best practical solution for my specific model. Swap memory, which is a sperate space on the disk has been able to continue the training when the physical Rando-

Access Memory (RAM) memory is full. For the Ubuntu system of this study, 6 GB Swap memory has been added. When the Ubuntu system run out of RAM, inactive pages which are in Ram has been moved to swap space in order to vacant the RAM memory. Consequently, the research study has been completed after overcoming the challenges and reducing the effects of the limitations.

# Chapter 7 : **Conclusion and Future Directions**

In this chapter 6, the last chapter of the thesis, initially the overall discussion for the proposed New Zealand Sign language translator is presented, and later, limitations and challenges that were experienced when conducting this research study will be discussed. Finally, some ideas for implanting this research study into future applications is reviewed.

## 7.1 Conclusion

Implementing an efficient method to translate dual-handed sign language recognition using image data is still considered a challenge. Similarity of gestures, signs with several meanings, user context, and accent lead to ambiguity. Initially, most of the literature and web articles were reviewed to identify the research gap for implementing sign language translators. In that step, the difficulty of translating dual-handed sign language into written language has been identified as a vital existing problem for deaf community. The deaf people in New Zealand adhere to a dual-handed sign language called, New Zealand Sign language (NZSL) and that has been selected as the main sign language addressed in the study. After identifying the research gap related to gesture-based sign language translators, goals for this research have been identified. The main goal of this research study is to find out NZSL could be successfully translated into text format. A literature review has been then conducted to clarify the research problem, the importance of addressing the identified problem, and the existing methods for implementing the system. After a thorough review of the online literature, the Convolution neural network (CNN) based transfer learning method has been used to implement the NZSL translator. Even though implementing systems for recognizing sign language systems go back to the late 70s' [19], there is no record of implementing translators for dual-handed sign languages. Consequently, it took considerable time to gather the relevant information and device with a design to train the system.

In order to train the neural network, it is essential to gather an accurate dataset that is relevant to New Zealand sign language. Even though there are several online databases to download single-handed sign language datasets, there is no resource for a double-handed sign language. Therefore, this research study has been begun from scratch. In order to increase the accuracy of the dataset, images have been collected under differing the background lighting conditions, rotating the sign, adding obstacles to the background, and varying the shape of the sign. Gathering 87,000 images is time consuming work and a challenging task. The pre-trained model called GoogleNet(InceptionV3) has been used as the pre-trained model for this network, and the training was conducted by using CNN algorithms. As per the algorithm, after the bottlenecks are created, the actual training for the top layer in the network begins. Then the training terminal displays the step output series, which indicate validation accuracy, training accuracy and cross entropy. It has taken more than 12 hours to complete the training of the Convolution neural network (CNN) for this proposed system.

Compared to the research outcomes found in the published research, this prototype sign language translating system has achieved a higher recognition rate of 99.1%. Further, to verify the higher model performance rate, graphs for training accuracy, validation accuracy, training loss and validation loss has been plotted. Apart from those basic graphs, confusion matrix and other relevant values, recall, precision, F1 Score and kappa value have been calculated to validate the classification performance of the proposed system. Those graphs and values are able to validate this implemented system for translating NZSL. As the final step, the trained system has been tested with using few different signers. The trained system is able to identify the gesture signs which were performed in front of the camera and translated them into text in the display. Some example images of the testing are displayed in the results and discussion chapter above. As per that Fig 5.1, the system is able to translate the sign for “Hello AI WORLD”.

So far, the results of the research study are promising enough to open the doors to various phases of practical applications. Further, InceptionV3 can be seen as the most suitable architectural pre-trained model to be implemented into appliances that are having low processing units. This gathered image data for English Alphabet and other symbols in NZSL will be saved in an online database to be shared with scientific community. It is hoped that this proposed system can be developed up to a program that allows people to communicate with each other regardless of their hearing ability.

## 7.2 Future Directions

This research study has identified a communication barrier between deaf people and haring-unimpaired people and has developed a prototype to identify and translate New Zealand gesture signs into English texts to bridge that communication gap. In the future, a fully operational system can be developed through this proposed NZSL translator to translate New Zealand signs into both text and speech, and to make that framework supportive for several platforms that use mobile based and web-based technologies. In order to determine the feasibility of the prototype in the educational field, interviews and surveys can be conducted through teachers who teach deaf students in separate classrooms or in mixed classes where hearing-unimpaired students and deaf students study together. Apart from that, the reverse version of this research outcome is also possible. In future, this proposed framework could be implemented to translate English sentences, which are in text mode or speech mode, into New Zealand gesture signs and that output can be integrated into multiple platforms, such as mobile based or web-based applications. Further this research work can be expanded to handling commands in video games through hand signs and more importantly, in the applications related to robot controlling.

# Appendix 1

[99]

**Image_capture.py Python program:**

```
import cv2
import os
import numpy as np


IMG_SIZE=200


#  coordinates of Region of Interest
top, right, bottom, left = 100, 150, 400, 450

exit_con='**'

a=''

dir0=input('enter the  name of the directory : ')

try:
  os.mkdir(dir0)
except:
  print('exist folder in the same name')

# referencing to the webcam
camera = cv2.VideoCapture(0,cv2.CAP_V4L)

while(True):

  a=input('exit: ** or Please enter the name of the label  : ')

  if a==exit_con:
    break

  dir1=str(dir0)+'/'+str(a)
  print(dir1)

  try:
    os.mkdir(dir1)
  except:
    print('EExcisting folder')

  i=0
  while(True):
    (t, frame) = camera.read()



    # Frame is being flipped
    frame = cv2.flip(frame, 1)

    # Obtaining the
    Roi = frame[top:bottom, right:left]

    # converting the Region of Interest into grayscale and then blur it
    Gray = cv2.cvtColor(Roi, cv2.COLOR_BGR2GRAY)
    Gray = cv2.GaussianBlur(gray, (7, 7), 0)

    #resize img
    Gray = cv2.resize(Roi, (IMG_SIZE,IMG_SIZE))

    cv2.imwrite("%s/%s/%s%s.jpg"%(dir0,a,a,i),gray)

    i+=1
    print(i)
    if i>3000:
      break

    cv2.rectangle(frame, (left, top), (right, bottom), (0,255,0), 2)

    cv2.imshow("Feeding Vedio 1", gray)

    cv2.imshow("Feeding Vedio", frame)

    keypress = cv2.waitKey(1)

    #Use "Esc", to stop rhe loop
    if keypress == 27:
      break

camera.release()
cv2.destroyAllWindows()
```

**Recognize_gesture.py Python program:**

```
import cv2
import os
import sys
import matplotlib
import matplotlib.pyplot as plt
import copy
import numpy as np
os.environ['TF_CPP_MIN_LOG_LEVEL']='2'
import tensorflow as tf

def predict(image_data):
   predictions = sess.run(softmax_tensor, \
          {'DecodeJpeg/contents:0': image_data})
   top_k = predictions[0].argsort()[-len(predictions[0]):][::-1]

   max_score = 0.0
   res = ''
   for node_id in top_k:
      human_string = label_lines[node_id]
      score = predictions[0][node_id]
      if score > max_score:
         max_score = score
         res = human_string
   return res, max_score

label_lines = [line.rstrip() for line
               in tf.gfile.GFile("logs/trained_labels.txt")]

with tf.gfile.FastGFile("logs/trained_graph.pb", 'rb') as f:
   graph_def = tf.GraphDef()
   graph_def.ParseFromString(f.read())
   _ = tf.import_graph_def(graph_def, name='')

with tf.Session() as sess:
    softmax_tensor = sess.graph.get_tensor_by_name('final_result:0')

   c = 0
   cap = cv2.VideoCapture(0,cv2.CAP_V4L)
   res, score = '', 0.0
   i = 0
   mem = ''
   consecutive = 0
```

```
  sequence = ''
    while True:
    ret, img = cap.read()
    img = cv2.flip(img, 1)

    if ret:
      x1, y1, x2, y2 = 100, 100, 300, 300
      img_cropped = img[y1:y2, x1:x2]

      c += 1
      image_data = cv2.imencode('.jpg', img_cropped)[1].tostring()
       a = cv2.waitKey(1) # waits to see if `esc` is pressed
      if i == 4:
        res_tmp, score = predict(image_data)
        res = res_tmp
        i = 0
        if mem == res:
          consecutive += 1
        else:
          consecutive = 0
        if consecutive == 2 and res not in ['nothing']:
          if res == 'space':
            sequence += ' '
          elif res == 'del':
            sequence = sequence[:-1]
          else:
            sequence += res
          consecutive = 0
      i += 1
      cv2.putText(img, '%s' % (res.upper()), (100,400), cv2.FONT_HERSHEY_SIMPLEX, 4, (255,255,255), 4)
      cv2.putText(img, '(score = %.5f)' % (float(score)), (100,450), cv2.FONT_HERSHEY_SIMPLEX, 1, (255,255,255))
      mem = res
      cv2.rectangle(img, (x1, y1), (x2, y2), (255,0,0), 2)
      cv2.imshow("img", img)
      img_sequence = np.zeros((200,1200,3), np.uint8)
      cv2.putText(img_sequence, '%s' % (sequence.upper()), (30,30), cv2.FONT_HERSHEY_SIMPLEX, 1, (255,255,255), 2)
      cv2.imshow('sequence', img_sequence)
      if a == 27:
        break
cv2.destroyAllWindows()
cv2.VideoCapture(0).release()
```

# Appendix 2

**A video demonstration of identifying New Zealand signs by the implemented system of this research, has been attached with this document.**

# References

[1] D. F.Armstrong, "The gestural theory of language origins," *Sign Language Studies,* vol. 8, no. 3, p. 289–314, 1999.

[2] N. Z. s. L. Board, "New Zealand sign Language Board," 2020. [Online]. Available: https://www.odi.govt.nz/nzsl/about/. [Accessed 10 11 2020].

[3] N. Z. Parliment, "New Zealand Parliment official website," 2006. [Online]. Available: https://www.parliament.nz/en/pb/bills-and-laws/bills-proposed-laws/document/00DBHOH_BILL6011_1/new-zealand-sign-language-bill. [Accessed 10 11 2020].

[4] Rachel McKee, " 'New Zealand Sign Language', Te Ara - the Encyclopedia of New Zealand," 2020. [Online]. Available: https://teara.govt.nz/en/new-zealand-sign-language. [Accessed 10 11 2020].

[5] T. Johnston, " BSL, Auslan and NZSL: three signed languages or one? in A Baker, B van den Bogaerde & O Crasborn (eds), Cross-linguistic perspectives in sign language research: selected papers from TISLR 2000," Amesterdam, 2000.

[6] fingeralphabet.org, "fingeralphabet.org," 2020. [Online]. Available: https://www.fingeralphabet.org/. [Accessed 10 11 2020].

[7] . K. P. Kour and L. Mathew, "LITERATURE SURVEY ON HAND GESTURE TECHNIQUES FOR SIGN LANGUAGE RECOGNITION," *International Journal of Technical Research & Science,* vol. 2, no. 7, pp. 431-433, 2017.

[8] "AI Media," 2020. [Online]. Available: https://www.ai-media.tv/sign-language-alphabets-from-around-the-world/. [Accessed 2 12 2020].

[9] S. Pivac Alexander, M. Vale and R. McKee, "E-learning of New Zealand Sign Language: Evaluating learners' perceptions and practical achievements.," *New Zeal. Stud. Appl. Linguist.,,* vol. 23, pp. 60-79, Nov. 2017.

[10] D. Jaffe, "Jaffe DL. Evolution of mechanical fingerspelling hands for people who are deaf-blind.," *Journal of Rehabilitation Research and Development. ,* vol. 31, no. 3, pp. 236-244, Aug 1994.

[11] B. Parton, "Sign Language Recognition and Translation: A Multidisciplined Approach From the Field of Artificial Intelligence.," *Journal of Deaf Studies and Deaf Education.,* pp. 94-101, Sep 2005.

[12] R. Alzohairi, R. Alghonaim, W. Alshehri, S. Aloqeely,, M. Alzaidan and O. Bchir, "Image based Arabic Sign Language Recognition," *(IJACSA) International Journal of Advanced Computer Science and Applications,* vol. 9, no. 3, pp. 185-194, 2018.

[13] V. Nwadinobi, "Chapter Eight Hearing Impairment," 25 09 2019. [Online]. Available: https://www.researchgate.net/publication/336025368_CHAPTER_EIGHT_HEARING_IMPAIRMENT. [Accessed 30 11 2020].

[14] WHO, "Deafness and hearing loss," 2020. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss. [Accessed 1 12 2020].

[15] N. Z. S. L. Board, "New Zealand Sign Language Board," 2018. [Online]. Available: https://www.odi.govt.nz/nzsl/about/. [Accessed 1 12 2020].

[16] “. v. 1. n. 3. p. 3. 2. M. Fernanda et al., "Main difficulties and obstacles faced by the deaf community in health access: an integrative literature review.," *Instituto Cefac São Paulo, Brasil,* vol. 19, no. 3, pp. 395-405, 2017.

[17] R. McKee, "Encyclopedia," 2020. [Online]. Available: https://teara.govt.nz/en/new-zealand-sign-language. [Accessed 1 12 2020].

[18] R. Ruben, "Sign language: Its history and contribution to the understanding of the biological nature of language," *Acta Oto-Laryngologica,* vol. 125, no. 5, pp. 464-467, 2005.

[19] P. Badhe and V. Kulkarni , "Indian Sign Language translator using gesture recognition algorithm.," *In: Proceedings of IEEE international conference on computer graphics on vision and information security (CGVIS), Bhubaneshwar, India,* pp. 195-200, 2015.

[20] A. Wadhawan and P. Kumar, "Sign Language Recognition Systems: A Decade Systematic Literature Review.," *Archives of Computational Methods in Engineering .,* 2019.

[21] C. Amrutha, N. Davis, K. Samrutha, N. Shilpa and J. Chunkath, "Improving Language Acquisition in Sensory Deficit Individuals with Mobile Application,," *International Conference on Emerging Trends in Engineering, Science and Technology (ICETEST-2015),* vol. 24, pp. 1068-1073, 2016.

[22] T. Dasgupta , S. Shukla , S. Kumar , S. Diwakar and A. Basu , "A multilingual multimedia Indian Sign Language dictionary tool.," *In: Proceedings of international joint conference on natural language processing, Hyderabad, India.,* pp. 57-64, 2008.

[23] M. F. Tolba and A. S. Elons, "Recent developments in sign language recognition systems, 2013 8th Int. Conf. Comput. Eng. Syst. ICCES," Cairo, Nov 2013.

[24] V. I. Pavlovic, R. Sharma and . T. S. Huang, "Visual interpretation of hand gestures for human-computer interaction: A review," *IEEE Trans Pattern Anal. Mach. Intell.,* vol. 19, no. 7, p. 677–695, Jul. 1997.

[25] S. S. Fels and G. E. Hinton, "Glove-talk: A neural network interface between a data-glove and a speech synthesizer," *IEEE Trans. Neural Netw.,* vol. 4, no. 1, pp. 2-8, 1993.

[26] D. J. Sturman and D. Zeltzer, "A survey of glove-based input," *IEEE Comput. Graph. Appl.,* vol. 14, no. 1, pp. 30-39, Jan. 1994.

[27] D. L. Quam, "Gesture Recognition with a Dataglove," *Proc. IEEE Natl.Aerosp. Electron. Conf. NAECON 90,* vol. 2, p. 755–760, May 1990.

[28] J. Eisenstein, S. Ghandeharizadeh, L. Huang, C. Shahabi, G. Shanbhag and R. Zimmermann, "Analysis of clustering techniques to detect hand signs," *Proc. 2001 Int. Symp. Intell. Multimed. Video Speech Process. ISIMP 2001 IEEE Cat No01EX489,* p. 259–262, May 2001.

[29] H. Brashear, V. Henderson, K. Park, H. Hamilton, S. Lee and T. Starner, "r, "American Sign Language Recognition in Game Development for Deaf Children," Portland, Oregon, USA, Oct. 2006.

[30] J. J. Stephan and S. Khudayer, "Gesture Recognition for Human-Computer Interaction (HCI)," *Int. J. Adv. Comp. Techn,* vol. 2, pp. 30-35, 2010.

[31] K. P. Kour and L. Mathew, "Literature Survey on Hand Gesture Techniques for Sign," *International Journal of Technical Research & Science,* vol. 2, no. 7, pp. 431-433, 2017.

[32] M. A. Ahmed, B. B. Zaidan, A. A. Zaidan, M. M. Salih and M. M. B. Lakulu, "A review on systems-based sensory gloves for sign language recognition state of the art between 2007 and 2017," *Sensors (Basel, Switzerland),* vol. 18, no. 7, 2018.

[33] A. Z. Shukor, M. F. Miskon and M. H. Jamalud, "ScienceDirect A New Data Glove Approach for Malaysian Sign Language Detection.," *Procedia - Procedia Comput. Sci.,* vol. 76, pp. 60-67, 2015.

[34] Y. Zheng, Y. Peng, G. Wang, X. Liu and X. Dong, "Development and evaluation of a sensor glove for hand function assessment and preliminary attempts at assessing hand coordination.," *J. Meas.,* vol. 93, pp. 1-12, 2016.

[35] Z. Shen, J. Yi, X. Li, M. Lo and . M. Chen, "A soft stretchable bending sensor and data glove applications.," *Robot. Biomim.,* vol. 3, no. 22, 2016.

[36] M. J. Cheok, Z. Omar and M. H. Jaward, "A review of hand gesture and sign language recognition techniques.," *Int. J. Mach. Learn. Cybern.,* vol. 10, no. 1, pp. 131-153, 2019.

[37] B. P. P. Kumar1 and M. B. Manjunatha, "A Hybrid Gesture Recognition Method for American sign Language.," *Indian Journal of Science and Technology.,* vol. 10, no. 1, Jan 2017.

[38] R. Jayaprakash and S. Majumder, "Hand Gesture Recognition for Sign Language: A New Hybrid Approach,.," *Conference: International Conference on Advanced Computing, Networking and Security, ADCONS 2011,* vol. 1, 2011.

[39] H. Stern, M. Shmueli and S. Berman, "Classifiers for hand gesture recognition," Herzelia, Israel, 2010.

[40] J. Wachs, A. Pablo, P. Juan, M. Wachs and H. Kölsch, "Vision-based hand-gesture applications.," *Commun. ACM,* vol. 54, p. 60–71, 2011.

[41] K. Bengler and W. Hahn, "Gesture Control for Use in Automobiles.," Tokyo, Japan, January 2000.

[42] S. Apoorv, S. Bhowmick and R. Prabha, "Indian sign language interpreter using image processing and machine learning.," *IOP Conf. Ser. Mater. Sci. Eng.,* p. 12026, Jun. 2020.

[43] D. Mandloi, "Implementation of image processing approach to translation of ASL finger-spelling to digital text.," Rochester Institute of Technology-Thesis, Rochester, 2006.

[44] J. Ravikiran, K. Mahesh, S. Mahishi, D. Dheeraj, S. Sudheender and N. Pujari, "Automatic Recognition of Sign Language Images," vol. 52, p. 321–332., 2010.

[45] M. Fagiani, E. Principi and S. Squartin, "A New Italian Sign Language Database. In: Zhang H., Hussain A., Liu D., Wang Z. (eds) Advances in Brain Inspired Cognitive Systems. BICS 2012. Lecture Notes in Computer Science," Berlin, 2012.

[46] O. Koller, J. Forster and H. Ney, "Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers.," *Comput. Vis. Image Underst.,* vol. 141, p. 108–125, Dec. 2015.

[47] P. Trigueiros, F. Ribeiro and L. P. Reis, "Hand Gesture Recognition System Based in Computer Vision and Machine Learning," in *Developments in Medical Image Processing and Computational Vision*, January 2015.

[48] G. R. S. Murthy and R. .. Jadon, "A review of vision based hand gesture recognition.," *Int. J. Inf. Technol. Knowl. Manag.,* vol. 2, p. 405–410, August 2009.

[49] M. d. L. Gorce, D. J. Fleet and N. Paragios, "Model-Based 3D Hand Pose Estimation from Monocular Video.," *IEEE Trans. Pattern Anal. Mach. Intell,* vol. 33, no. 9, p. 1793–1805, September 2011.

[50] C. Zhang, P. P. and . H. Haddadi,, "Deep Learning in Mobile and Wireless Networking," *A Survey," IEEE Commun. Surv. Tutorials.,* vol. 21, no. 3, p. 2224–2287, 2019.

[51] P. Anjankar and S. Borkar, "Artificial Intelligence Based Robot Control Using Face and Hand Gesture Recognition.," *A Review," IJRIT,* vol. 3, p. 120–126, February 2015.

[52] J. Frankenfield, "Investopedia," 28 08 2020. [Online]. Available: https://www.investopedia.com/terms/a/artificial-neural-networks-ann.asp#:~:text=An%20artificial%20neural%20network%20(ANN)%20is%20the%20piece%20of%20a,by%20human%20or%20statistical%20standards.. [Accessed 21 12 2020].

[53] S. University., " CS231n Convolutional Neural Networks for Visual Recognition.," 2016. [Online]. Available: http : / / cs231n . github . io / neural - networks - 1. [Accessed 21 12 2020].

[54] M. Islam, M. Hossain, R. Islam and K. Andersson, "Static Hand Gesture Recognition using Convolutional Neural Network with Data Augmentation.," Washington, USA, April 2019.

[55] G. Halvardsson, J. Peterson, C. Soto-Valero and B. Baudry, "Interpretation of Swedish Sign Language using Convolutional Neural Networks and Transfer Learning.," Stockholm, Sweden, October, 2020.

[56] E. Mathe, A. Mitsou, E. Spyrou and P. Mylonas, "Arm Gesture Recognition using a Convolutional Neural Network.," September 2018.

[57] R. Su, . X. Liu and L. Wang, "Convolutional neural network bottleneck features for bi-directional generalized variable parameter HMMs.," August 2016.

[58] T. Shanthia, R. Sabeenian and R. Anandc, "Automatic diagnosis of skin diseases using convolution neural network," *Microprocessors and Microsystems.,* vol. 76, pp. 1-8, July 2020.

[59] . H. Fujiyoshi, T. Hirakawa and T. Yamashita, "Deep learningbased image recognition for autonomous driving.," *IATSS Research ,* vol. 43, p. 244–252, 2019.

[60] H. Fujiyoshi, T. Hirakawa and T. Yamashita, "Deep learningbased image recognition for autonomous driving.," *IATSS Research.,* vol. 43, p. 244–252, 2019.

[61] W. Xu, X. Zhang, L. Yao, W. Xue and B. Wei, " A multi-view cnn-based acoustic classification system for automatic animal species identification," *Ad Hoc Networks ,* 2020.

[62] P. Ahmadvand, R. Ebrahimpour and P. Ahmadvand, "How popular cnns perform in real applications of face recognition.," 2016.

[63] S. Shen, M. Sadoughi, . M. Li, Z. Wang and C. Hu, "Deep convolutional neural networks with ensemble learning and transfer learning for capacity estimation of lithium-ion batteries.," 2020.

[64] G. Halvardsson, J. Peterson, C. Soto-Valero and B. Baudry, "Interpretation of Swedish Sign Language using Convolutional Neural Networks and Transfer Learning.," Elsvier, Stockholm, Sweden, 2020.

[65] S. University, "CS231n Convolutional Neural Networks for Visual Recognition - Transfer Learning.," 2020. [Online]. Available: url: http://cs231n.github.io/transfer-learning/.. [Accessed 09 12 2020].

[66] E. B. Ylla, "Synthesizing Images to Recognize Natural Images with Transfer Learning in Convolutional Neural Networks.," København, 2017.

[67] Y. LeCun, "Backpropagation Applied to Handwritten Zip Code Recognition," *In: Neural Computation,* vol. 1, no. 4, p. 541–551, 1989.

[68] B. LeCun, Y. Huang and F. , ". “Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting.," 2004.

[69] J. Brownlee, "MAchine Learning Mastery," May 2019. [Online]. Available: https://machinelearningmastery.com/how-to-use-transfer-learning-when-developing-convolutional-neural-network-models/. [Accessed 09 12 2020].

[70] K. Developer, "Keras," 2020. [Online]. Available: https://keras.io/api/applications/. [Accessed 09 12 2020].

[71] B. Raj, "Towards Data Science.," 30 05 2018. [Online]. Available: https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202. [Accessed 10 12 2020].

[72] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, "Rethinking the Inception Architecture for Computer Vision," Boston, 2015.

[73] . S. Sheshadri, "Introducing the Ultimate Starter AI Computer, the NVIDIA Jetson Nano 2GB Developer Kit | NVIDIA Developer Blog.," 2020. [Online]. Available: https://developer.nvidia.com/blog/ultimate-starter-ai-computer-jetson-. [Accessed 22 12 2020].

[74] L. Barba-Guaman, J. E. Naranjo and A. Ortiz, "Deep learning framework for vehicle and pedestrian detection in rural roads on an embedded GPU.," *Electron,* vol. 9, no. 4, pp. 1-17, 2020.

[75] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovic, "Going deeper with convolutions," *In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition.,* pp. 1-9, June 2014.

[76] Y. Shu, Y. Chen and C. Xiong, "Application of image recognition technology based on embedded technology in environmental pollution detection.," *Microprocessors and Microsystems .,* vol. 75, pp. 1-5, 2020.

[77] G. Halvardsson, J. Peterson, C. Soto-Valero and B. Baudry, "Interpretation of Swedish Sign Language using Convolutional Neural Networks and Transfer Learning.," Stockholm, Sweden, 2020.

[78] A. Bhandari, "Analytics Vidhya," 2020. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/. [Accessed 11 12 2020].

[79] M. J. Anzanello and F. S. Fogliatto , "Learning curve models and applications: Literature review and research directions," *International Journal of Industrial Ergonomics,* vol. 41, pp. 573-583, 2011.

[80] C. Szegedy, V. Vanhoucke, S. Ioffe and J. Shlens, "Rethinking the Inception Architecture for Computer Vision," 2015.

[81] V. Kumarama, "Paper Space Blog-A Review of Popular Deep Learning Architectures: ResNet, InceptionV3, and SqueezeNet," 05 06 2020. [Online]. Available: https://blog.paperspace.com/popular-deep-learning-architectures-resnet-inceptionv3-squeezenet/. [Accessed 09 12 2020].

[82] P. Huilgol, "Analytical Vidya," 18 08 2020. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/08/top-4-pre-trained-models-for-image-classification-with-python-code/. [Accessed 10 12 2020].

[83] C. Szegedy, V. Vanhoucke, S. Ioffe and J. Shlens, "Rethinking the Inception Architecture for Computer Vision," *CVPR- Computer Vision Foundation.,* pp. 2818-2826, 2015.

[84] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition.," 2014.

[85] C. Szegedy, W. Liu, Y. Jia, P. Sermanet and S. Reed, "Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 1-9, 2015.

[86] A. A. e. a. Martin Abadi, "TensorFlow: Large-ScaleMachine Learning on Heterogeneous Distributed Systems.," 2016.

[87] Z. A. Ansari and G. Harit, "Nearest neighbour classification of Indian sign language gestures using kinect camera.," *Indian Academy of Sciences.,* vol. 41, no. 2, p. 161–182, February 2016.

[88] A. Deza and D. Hasan , "Hasan," December 2018.

[89] L. Barba-Guaman, J. E. u. Naranjo and A. Ortiz, "Deep Learning Framework for Vehicle and Pedestrian Detection in Rural Roads on an Embedded GPU.," *Electronocs 589,* vol. 9, no. 4, pp. 1-17, March 2020.

[90] X. Feng, Y. Jiang, X. Yang, M. Du and X. Li, "Computer vision algorithms and hardware implementations.," *A survey. Integr. VLSI J,* vol. 69, p. 309–320, 2019.

[91] C. Zhang, P. Patras and H. Haddadi, "Deep learning in mobile and wireless networking: A survey.," *IEEE Commun.Surv. Tutor. ,* vol. 21, p. 2224–2287, 2019.

[92] A. HajiRassouliha, A. Taberner, M. Nash and P. Niel, "Suitability of recent hardware accelerators (DSPs,FPGAs, and GPUs) for computer vision and image processing algorithms.," *Signal Process. Image Commun.,* vol. 68, p. 101–119, 2018.

[93] A. Basulto-Lantsova, J. Padilla-Medina, F. Perez-Pi and A. Barranco-Gutierrez, "Performance comparative of OpenCV Template Matching method on Jetson TX2 and Jetson Nano developer kits.In Proceedings of the 2020 10th Annual Computing and Communication Workshop and Conference (CCWC),Las Vegas, NV,USA," p. 0812–0816, January 2020.

[94] NVIDIA, "NVIDIA JetPack," 2020. [Online]. Available: https://developer.nvidia.com/embedded/jetpack. [Accessed 20 12 2020].

[95] V. Mazzia, A. Khaliq, F. Salvetti and M. Chiaberge, "Real-Time Apple Detection System Using Embedded Systems With Hardware Accelerators: An Edge AI Application," *IEEE Access,* vol. 8, pp. 9102-9114, 2020.

[96] R. Febbo, B. Flood, J. Halloy, P. Lau, K. Wong and A. Ayala, "Autonomous Vehicle Control Using a Deep Neural Network and Jetson Nano Network and Jetson Nano.," New York NY, United States, 2020.

[97] J. Brownlee, "Machine Learning Mastery.," February 2019. [Online]. Available: https://machinelearningmastery.com/learning-curves-for-diagnosing-machine-learning-model-performance/. [Accessed 10 1 2021].

[98] NVIDIA, "NVIDIA Accelarated Computing," 2020. [Online]. Available: https://docs.nvidia.com/deeplearning/frameworks/install-tf-jetson-platform-release-notes/tf-jetson-rel.html. [Accessed 01 01 2021].

[99] W. Xin, "CSDN," 30 05 2017. [Online]. Available: https://blog.csdn.net/l18930738887/article/details/72812689. [Accessed 02 02 2021].

[100] J. L. Raheja, R. Shyam, U. K. and P. . B. Prasad, "Real-Time Robotic Hand Control using Hand Gestures," in *Second International Conference on Machine Learning and Computing*, Pilani – 333031, Rajasthan, 2010 .

[101] M. University, "Gesture Dataset 2012," 2012. [Online]. Available: https://www.massey.ac.nz/~albarcza/gesture_dataset2012.html. [Accessed 08 12 2020].

[102] I. L.-S. V. R. C. ILSVRC2014, "Results of Imagenet Large-Scale Visual Recognition Challenge 2014 -ILSVRC2014.," Stanford Vision Lab, Stanford, 2014.

[103] B. Garcia and S. A. Viesca, "Real-time American Sign Language Recognition with Convolutional Neural Networks.," pp. 225-232, 2016.

[104] A. Rosebrock, " Deep Learning for Computer Vision with Python: ImageNet Bundle.," New York, 2017.

[105] S. Chetlur, C. Woolley, P. Vandermersch, B. Catanzaro and E. Shelhamer, "cudnn: Efficient primitives for deep learning.," December 2014.

[106] J. Nickolls, . I. Buck, M. Garland and K. Skadron, "Scalable parallel programming with CUDA.," *ACM Queue ,* vol. 6, pp. 40-53, 2008.

[107] "Jetson Nano Brings AI Computing to Everyone.," 2020. [Online]. Available: https://devblogs.nvidia.com/jetsonnano-ai-computing/ . [Accessed 20 12 2020].

[108] S. Yegulalp, "InfoWorld," 2019. [Online]. Available: https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-

library-explained.html#:~:text=Created%20by%20the%20Google%20Brain,way%20of%20a%20common%20metaphor.. [Accessed 21 12 2020].

[109] O. Library, "Intel acquires Itseez:OpenCv.," 27 05 2016. [Online]. Available: https://opencv.org/intel-acquires-itseez.html. [Accessed 21 12 2020].