2023

A Model for sign language recognition for Kenyan sign language.

Wanjala, Geoffrey Kasembeli
School of Computing and Engineering Sciences
Strathmore University

A Model for Sign Language Recognition for Kenyan Sign Language

By

Geoffrey Kasembeli Wanjala

147823

Master of Science in Information Technology

2023

A Model for Sign Language Recognition for Kenyan Sign Language

By

Geoffrey Kasembeli Wanjala

147823

Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Information Technology at Strathmore University

School of Computing & Engineering Sciences

Strathmore University

Nairobi, Kenya

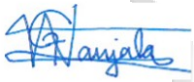
July, 2023

Declaration and Approval

Declaration

I declare that this work has not been previously submitted and approval for the award of a degree by this or any other University. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due references is made in the thesis itself.

Student's Name: Geoffrey Kasembeli Wanjala

Signature  Date May 25th 2023

Approval

The thesis of Geoffrey Kasembeli Wanjala was reviewed and approved for examination by the following:

Dr. Henry Muchiri, PhD
School of Computing and Engineering Sciences,
Strathmore University

Dr. Julius Butime,
Dean, School of Computing & Engineering Sciences,
Strathmore University

Dr. Bernard Shibwabo,
Director of Graduate Studies,
Strathmore University

Abstract

Computer vision aids in increasing tech accessible for communities that are underserved, such as the disabled community. This study demonstrates how artificial intelligence via computer vision helps to bridge the communication gap between those with hearing problems and the general population. The purpose of this paper is to bring forth an artificial intelligence solution to cater to this targeted group to aid in communication. Artificial intelligence has come a long way to solve this problem of enabling sign language notations to be translated into readable form that can be easily understood. This is in accordance with the fact that there is a collective duty to ensure that the deaf can be part of our society on an equal basis with others, free from discrimination even when it comes to speech and communication. There is a great need for this interpretation so that communication is sped up through translation. Understanding between the deaf and hearing people can be fostered as well as costs associated with training individuals in sign language communication in sign language training centers are minimized.

To answer this question, the research work collected and analyzed photos and videos in a quasi-experiment consisting of target photos of Kenyan sign language notations. The model is trained on 9100 Kenyan sign language (KSL) notations of varied gestures spanning from health and wellness to common day to day basic notations such as greetings, expressing feelings among others. Transfer learning through TensorFlow object detection model, OpenCV framework for image processing and python was used to actualize this sign language translation model in this research work. A trained machine learning model organizes the input photos and videos, analyses them and produces text that maps to the corresponding sign language notation used. Individual users can use the model to translate Kenyan sign language notations into readable English text.

The model gave performance levels of 85% accuracy on a 20,000 training steps for 40 epochs. This gave a perfect balance of training duration and accuracy levels on the dataset given. One of the notable findings was that notations that involved movement of the hands and other body parts to express gestures were harder to detect and translate due to the motions involved. A lot of training data on such notations is needed to train the model further in detecting them.

Keywords: *Artificial Intelligence, computer vision, machine learning, sign language, disability*

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AE	-	Autoencoder-based deep neural network
AI	-	Artificial Intelligence
ANN	-	Artificial Neural Network
ASL	-	American Sign Language
ASLR	-	Automatic Sign Language Recognition
CNN	-	Convolutional Neural Network
DBN	-	Dynamic Bayesian Network
HMMs	-	Hidden Markov Model
KNAD	-	Kenya National Association of the Deaf
KNCHR	-	Kenya National Commission on Human Rights
KSL	-	Kenyan Sign Language
ML	-	Machine Learning
NCPWD	-	National Council for Persons with Disabilities
NMT	-	Neural Machine Translation
RNN	-	Recurrent Neural Network
SLID	-	Sign Language Identification
SLP	-	Sign Language Production
SLR	-	Sign Language Recognition
SSD	-	Single Shot Detector

Acknowledgements

Sincere thanks to the Lord God Almighty for the strength, determination and grace to undertake this study. Lots of gratitude to my supervisor, Dr. Henry Muchiri, for his hand holding and knowledge sharing towards the success of this study. Strathmore university as an institution deserves my sincere gratitude for giving me the environment conducive through the facilities and skilled faculty to impart me the necessary tools and skills needed to accomplish this study. Prof. Ismail Ateya, with all his unique guidance on the thesis seminars enabled me to make this key milestone of completing this study. I would also be doing a disservice by not recognizing Dr. Vincent Omwenga at Strathmore University who guided us on the itinerary and facilitating our smooth stay in the International Module at the Czech Republic.

Definition of Terms

Artificial Intelligence	This refers to the ability and various technologies that allow machines to mimic human intelligence and expertise. This involves various levels such as natural language, cognitive thinking and decision analysis. (Neapolitan & Jiang, 2018)
Automatic Sign Language Recognition	Sign Language Recognition (SLR) is the process of automatically recognizing the meaning of the movements and postures, which may be translated into one or multiple words in the spoken language. (Bantupalli & Xie, 2019)
Deep Learning	Is a subset of AI, which is basically a brain network with at least three layers. These brain networks endeavor to recreate the way of behaving of the human mind permitting it to "learn" from a lot of information. (Charniak & Eugene, 2018)
Hidden Markov Model	It is a measurable model that can be used to showcase the development of recognizable occasions that rely upon interior variables, which are not straightforwardly perceptible. The noticed event is called an 'image' and the invisible component fundamental to the perception a 'state'.(Mor et al., 2021)
Neural Network	This comprises of thousands or even great many straightforward handling hubs that are thickly interconnected. They are a method for AI where a PC figures out how to play out an undertaking by examining explicit models. (Aggarwal, 2018)
Sign Language Production	The automatic conversion of spoken language sentences into sign language sequences. SLP is the reverse of sign language translation, which translates from text to sign. (Quinto-Pozos & Cooley, 2020)

Chapter 1 Introduction

1.1 Background

One of the key issues that has always existed is how to make the unfortunate to also enjoy lives like others in a societal setting. People living with disability need to be accommodated in every way possible even in normal communication with their peers. The deaf and dumb are the individuals falling in this group. They can neither speak nor talk effectively. Such shortcomings could be as a result of sickness or just simply by birth. Various solutions have been invented to bridge this shortcoming. Braille has been used to cater for the blind. Ear piece amplifiers have been used to aid listening for the partially deaf. Being deaf is a global issue that various technological advancement aims to improve on it (Papastratis et al., 2021).

There exist various sign languages in different jurisdiction of the world. Americans use American sign language which is much different from Indian sign language in India. China has a distinct sign language as well as Kenya. Each part of the world has differing sign languages consisting of differing notations. These variation in notations come as a result of the cultural influences, environmental factors as well as the day-to-day beliefs. Linguistics keeps evolving as well to different languages and forms of expression through speech and non-verbal communication to convey one's ideas (Lucas & Bayley, 2011).

There are several challenges that exists affecting the deaf community and their interaction with the rest of the population. Normal people have difficulties comprehending Deaf and Mute people's language since they communicate using hand gestures. It is challenging to assess how the educational system accommodates the local deaf population in the same learning environment and curriculum. Kenya is a multinational country that has more than one language that is spoken by its natives. The deaf community speaks only one language while the hearing population speaks three languages. All these challenges have a footing on how the disabled population live their daily lives (Dhake et al., 2020). Furthermore, studies have found that parents' failure to learn sign language contributes to some of the communication challenges they face with their hearing-impaired children (Luchivya et al., 2020).

Technology has come a long way to aid in sign language communication as well as break the barrier in communication using visual images, braille and other tools used for this matter. However much technology has been evolving, researchers have also been using emerging technologies to make life better for these special groups. Artificial intelligence, machine

learning and computer vision are technologies that have been adopted to aid in sign language communication (Patel, 2022).

Object detection has been one of the key algorithms and models that has been used by computer vision to single out objects within a field of view on the images and video inputs. Most applications of object detection tracking of vehicles and traffic safety with the aims to control careless traffic on the roads, vehicles identification and monitoring. This type of application has aided traffic officers and law enforcement in modifying residents' perceptions of irresponsible driving on the highways, marking the first step toward changing transportation behavior (Hammoudeh et al., 2022). Object detection has been used in software quality assurance testing as an automation tool to test window-based application and window-based application hidden objects by comparing images of application windows (Anam et al., 2015).

Understanding sign language needs proper conventional training both for the deaf individuals and also anyone who wishes to talk and interact with them in a meaningful way. This means that time for learning, costs and accommodation are just few of the things that are be needed to achieve the end goal of elevating this communication barriers with the deaf (Kusters & Hou, 2020). Technology through artificial intelligence has come to reduce the training and time constraint that has existed. Training the deaf to speak sign language and also any individual interacting with them needed a lot of time. Artificial intelligence has been an emerging technology that has some of its applications in assisting people living with disability (PWD), especially the deaf to be able to communicate easily with individuals around them in order to form associations and relationships easily as well as carry out their day-to-day activities with ease. Communication is the basic foundation for understanding and sharing of ideas. These ideas when communicated well can foster understanding, interactions and normal living. (Holmer et al., 2017)

The number of Kenyan people with hearing impairment (deaf) are around 150,000 based from the recent 2019 census report (Development Initiatives, 2020). This is definitely a large number and thus shows criticality of the issue at hand. Technology has a part to play in enabling the deaf to live fruitfully. Mentioned the use of assistive technologies like cochlear implants and other support services that are used in interpretation and real-time text in the classroom. These technologies have proven to have some certain level of efficiency towards the cause but not sufficient enough to address the aforementioned challenge. (Crowe et al., 2017)

1.2 Problem Statement

Sign language is a form of communication used by people with impaired hearing and speech. Having the capabilities of translation between spoken and sign language can link individuals that otherwise may not have been able to clearly share their ideas, thoughts and feelings in a more effective way. A door is opened to a more socially enriching and a well-rounded form of interaction fostered by this communication when neither party are having struggles engaging in communication so as to make their wants, needs and ideas open to the rest. Hearing still falls short even though there are governance systems and structures in the form of law that are tailored to facilitate accessibility. The hearing-impaired face lots of shortcomings on a day-to-day basis which can in many cases be discouraging and uncomforting.

Existing techniques have been leveraged to cater for this issue but challenges still exist. Human Sign language interpreters and teachers have been used to bridge this gap but it is still not sufficient enough to cater for the entire population and facilitate communication between the two groups. Technology has intervened to present current sign language approaches that are sensor-based which uses gloves worn on the hands or Kinect based that reads hand trajectories and hand shapes in determining sign language notations (Bulugu, 2021). Others are vision based using machine learning and computer vision techniques that used algorithms to detect and recognize sign language notations. Hybrid methods involve a combination of both to achieve sign language recognition (Papastratis et al., 2021). The reason for translating Kenyan Sign Language is because its the main communication mode for individuals with hearing and speech impaired needs in kenya and should be translated to spoken language to the hearing population (Dhake et al., 2020).The problem is that sensor and hybrid-based methods are purely expensive and resource intensive to implement than computer vision techniques and also there is a need to have a system that translates Kenyan Sign Language (KSL) into a readable format that can aid in interacting with normal hearing people (Bantupalli & Xie, 2019).

1.3 Main Objective

To analyse and develop a model for Kenyan Sign Language (KSL) translation using computer vision to allow signers to communicate with the hearing population.

1.4 Specific Objectives

i. To investigate the challenges in Kenyan Sign Language translation.
ii. To review existing computer vision algorithms and models used in Sign Language recognition.

iii. To design a computer vision model that recognizes Kenyan Sign Language.
iv. To validate the computer vision model.

1.5 Research Questions

i. What challenges exist in Kenyan Sign Language translation.
ii. What computer vision algorithms and models exist for this research?
iii. How can a computer vision model to translate Kenyan Sign Language be developed?
iv. What is the performance of the developed model?

1.6 Justification

This research is aimed for the benefit of the public including individuals engaging with the deaf, parent with deaf children, employees at work who are deaf and need to be understood with the people they are interacting with. Computer vision is the driving engine of the solution. The reason why computer-based solutions rank the best is because they involve faster and simpler processes due to the use of algorithms and not mechanical devices. They are also able to carry out repetitive and monotonous tasks in analyzing and classifying images and videos. They can also be trained very well and commit zero mistakes. In summary, they exhibit speed of execution, reliability, accuracy and versatile allowing it to be used for various uses.

The research work solves the problems for the hearing impaired by providing a seamless technology that interprets sign language into human readable text in English. It fosters clear communication. It minimises the learning curve for parents and colleagues for having to spend quite a lot of hours to learn sign language symbols. This research provides definite ways of improving solutions for the people living with disability in Kenya. Computer vision can also be adapted on other areas of disability that can leveraged by the findings of this research.

1.7 Scope and Limitations

The study geographical area of the research work is based in Kenya. It focuses on the problems that the deaf face in their day-to-day activities and the proposed solution to this problem. This makes sure that the issues can be understood based on the context of the citizen of the country. The study also focuses on Kenyan Sign Language and not American or Chinese Sign Language (These languages are significantly different). The proposed solution is limited to sign language gestures in health and wellbeing. The images and videos collected of the notations consists of gestures that express hygiene, happiness, satisfaction, food etc. which forms a huge portion of the Kenyan Sign Language.

The deaf are the target group. They are the main group for this solution. The governing institutions mentioned in the target groups are the key players in information gathering for the research. (Kenya national association of the deaf, national council for persons living with disability and Kenya society for deaf children). The study does not focus on all signs in Kenyan Sign Language. It focuses on a few key signs first. This limitation is based on the availability of the data in the data collection phase and two, the study might be very large if the focus is very wide.

Chapter 2 Literature Review

2.1 Introduction

Computer vision, being a subset of machine learning aims to provide computer systems the ability to see and process vision. It models the human vision into the computing devices and systems. This technique has enabled researchers to use it in solving real world solutions that has helped the human population to live and operate in a more fulfilling way. Solutions such as traffics flow analysis, self-driving cars, cancer detection just to name a few. the focus of this paper is on Kenyan sign language translation where computer vision is still playing a key part in the solution. The discussion and review of the various knowledge areas of this literature serves as a foundation and basis for the research topic.

2.2 Theoretical Literature

2.2.1 Kenyan Sign Language

Sign languages are those that use visual and manual modalities instead of verbal spoken words. It can be stated that sign languages are a fully-fledged language on its own that can be used to express an individual's thoughts and ideas. The deaf community in Kenya and Somalia primarily uses Kenyan Sign Language. One of the distinguishing features of the KSL like in most sign language is the hand gestures, the motions and movement of the gestures and the location of the movement as well as the orientation of the palm (Mweri, 2018).

The KSL is acknowledged as one of the official languages of the Kenyan parliament (Article 120), and it is expressly stated in Article 7(3)(b) that the state would support the growth of the KSL. This ensures its development and maintenance as well as conducting all initiatives to

2.2.2 Challenges faced in the translation of Kenyan Sign Language

Every living person has access to a certain set of fundamental freedoms that enable them to lead fulfilled lives. Rights to justice as well as those to education, transportation, healthcare, employment, and information. The Kenya National Commission on Human Rights has outlined the essential areas to which individuals with disabilities are entitled, as well as the different levels of access to these amenities without jeopardizing their safety or worse, endangering their health because of their physical problems. The different types of disabilities and the different economic sectors are outlined, along with suggestions for how each could be improved to better serve those who are disabled (Kenya National Commission on Human Rights (KNCHR), 2014).

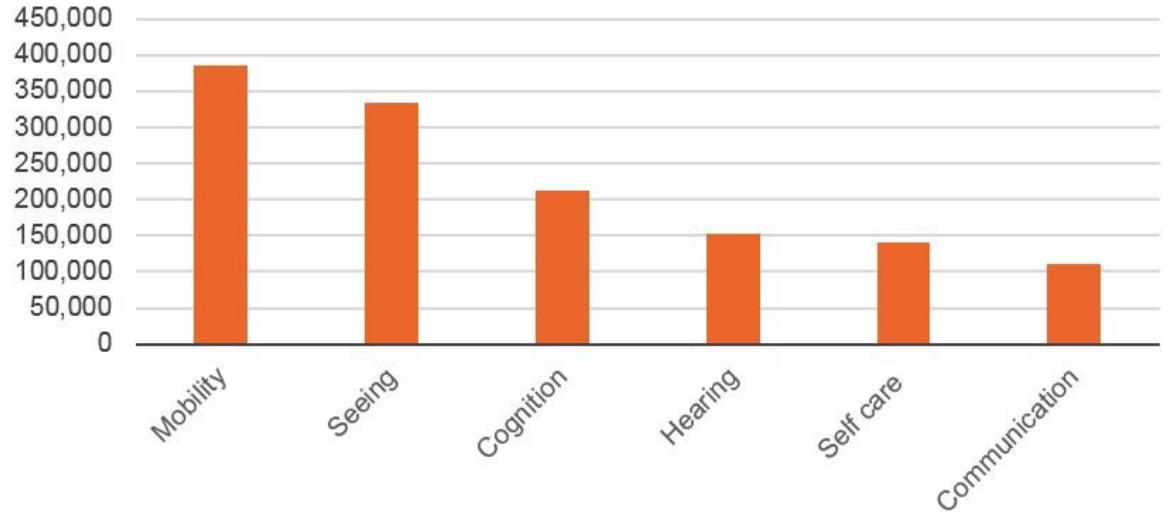


Figure 2.1 Status of disability in Kenya (Census Statistics 2019)

Kenya's National Council for People with Disabilities is a government organization entrusted with integrating the aforementioned group into all facets of the country's economy. It enacts laws and regulations that different important national players can incorporate into their businesses, facilities, and industries. The IT industry has a part to play in this endeavor as well. To support this mainstreaming, many technology tools and solutions can be deployed. The main objective is for persons with disabilities to enjoy and lead regular lives, just like the rest of society (Parida & Sinha, 2021).

Sign languages have their own distinct syntax in that they are not an exact mapping of spoken languages. For example, the right eyebrow position must be used with a well-constructed statement. It is normal for the eyebrows to be in a particular posture while someone is asking questions about who, where, what, why, and when. The eyebrows are typically anticipated to rise in a certain way if the question is a yes-or-no situation. SL involves body posture, facial expressions, hand locations, and movements in addition to hand gestures while expressing ideas. The vocal cords are used to produce speech, while the ears are used to listen for sounds. Even though participants of spoken languages are able to produce and understand audible sources and written language, spoken languages also require visible communicating gestures. In contrast, sign languages only require visible actions and visual perception. Additionally, sign languages use simultaneous configuration and movement of multiple bodily parts, including the hands, head, face, eyes, and torso, unlike speech, which produces utterances through phonation, which is a single vibrating column of air. Translation is difficult because

the meaning might vary with a change in any of these aspects or with the passage of time. (Wolfe, 2021).

Additionally, studies have indicated that parents' refusal to learn sign language is a contributing factor in some of the communication difficulties they experience with their hearing impaired children. The children experience cognitive development delays because they lack a home language of communication with their parents and siblings when their parents do not learn how to sign (Luchivya et al., 2020).

2.2.3 Difficulties faced by the deaf and dumb

Interacting with those who have hearing loss is really difficult. Since Deaf and Mute people utilize hand gestures to communicate, normal people have difficulty understanding their language from the signs they make. Systems that can recognize various signs and communicate information to common people are thus necessary. Over the years, communication barriers between hearing and deaf people have created several challenges in both the labor market and the educational system. It is important to carefully assess how the educational system accommodates the local deaf population. Kenya is a multinational country that has more than one language that is spoken by its natives. These languages are English, Swahili and other mother tongues while there is a propensity to lump up all Kenyan sign language users together. The deaf community speaks only one language while the hearing population speaks three languages (Dhake et al., 2020).

2.2.4 Technology Intervention through Sign Language Translation

The ability to recognize sign languages has been made possible and improved by technology, particularly computer vision. Researchers have made significant progress in replacing device-based methods with vision-based techniques employing artificial intelligence (AI) and deep learning, in order to not only remove any barrier to communication between hearing-impaired and non-impaired people. Researchers have separated the various tasks involved in processing the language. We have sign language identification (SLI) and sign language recognition (SLR). Identification of signer languages is the goal of sign language identification. The goal of sign language recognition is to convert the signer's movements into tokens or signs (Sultan et al., 2022).

Sign language recognition focuses on translating any hand gestures and body positions used in sign language, sign gesture, and also text generation to the hearing people to understand the deaf people. The feature extraction stage of the recognition system is essential for detecting any sign. It is crucial for recognizing signs. They must be unique, normalized, and pre-

processed. Identification of sign language is the process of putting a language to a set of hand gestures, postures, movements, and facial expressions of movements. Researchers are still exploring for several effective models for sign language recognition that can aid in the understanding of sign language (Sultan et al., 2022).

2.2.5 Computer Vision

The recognition of sign languages has benefited greatly from advancements in computer vision technology. It replaces human eyes and brains with input and peripheral devices like cameras to acquire video information, and it employs computers to analyze video content in place of human brains to achieve objective perception and comprehension of the three-dimensional world. A lot of effort has been paid to computer vision-based human behavior recognition in recent years from a variety of study domains. Computer vision is utilized in virtual reality, security systems, human-computer interface, sports support, and clinical correction. Recognition of gestures comprises two steps that are essential to the vision process. These range from high level analysis to low level processing. The low-level procedure entails trajectory prediction, tracking, and detection. Extraction of characteristics, modeling of gestures, and scene relationship are all included in the high-level analysis. This is in line with what Sultan said about recognizing and identifying sign language (Zhao et al., 2021).

Computer vision is a field of artificial intelligence (AI) that allows systems to extract information from digital photos, videos, and other visual inputs and act on it or make recommendations. The addition of Deep Learning represents a significant change in this. It's true that the field of computer vision and pattern recognition has seen a significant change as a result (Davies, 2018).

2.2.6 Deep learning

Convolutional Neural Networks (CNN) and deep learning are used in computer vision to perform high-speed, high-volume unsupervised learning on visual input in order to train machine learning systems to interpret data in a manner somewhat similar to how a human eye functions. The fundamental advantage of deep learning is that the network automatically learns the data; there is no need to manually extract features. Deep learning can extract characteristics from huge data that typically have great migration and generalization, making it more suitable for target and behavior identification. This learning corresponds to how human neurons interpret their surroundings (Hassaballah & Awad, 2020).

Deep learning-based algorithms typically employ networks such as supervised convolutional neural network (CNN), supervised recurrent neural network (RNN), and autoencoder-based

deep neural network (AE). The most common form of neural network has two characteristics. The local connection is the first feature that may be reduced in fully connected neural networks to reduce the impact of overfitting. The second strategy is weight sharing, which can further reduce network parameters and integrate local elements of images retrieved from local perception into higher-level global features (Zhao et al., 2021).

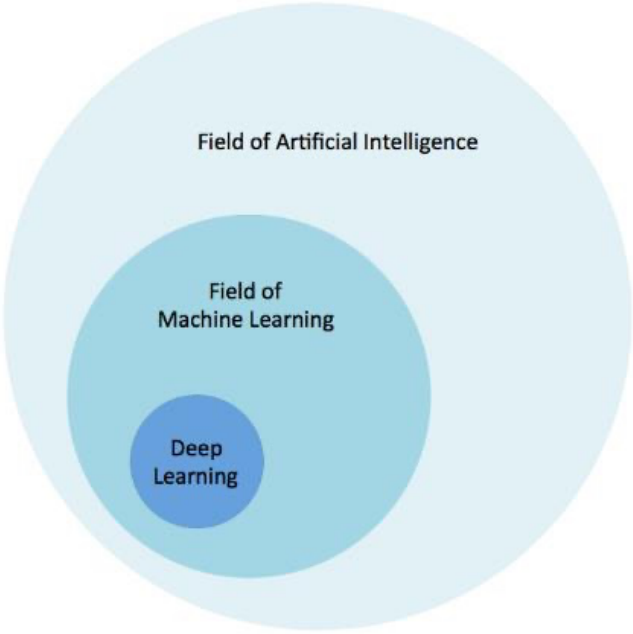


Figure 2.2 The relationship between AI, deep learning and Machine learning. (Gibson and Patterson 2017)

2.2.7 Sign language recognition pre-processing steps

The processes for pre-processing differ amongst sign language models. They do, however, have certain commonalities. The most common tasks are segmentation and filtering. These tasks are divided into the following subtasks: gesture recognition, handshape identification, feature extraction, picture and video segmentation, and skin detection. Each model frequently starts with the signer's picture and transforms it to a different color space. Non-skin photographs are removed, and the remaining images are subjected to image morphology (erosion and dilation) to decrease noise. Each image is checked to determine whether there is a hand present. If this is the case, certain approaches are employed to divide fingers from hand mask pictures and determine the Region-of-Interest (ROI). Edge detection may be accomplished using a variety of image enhancement techniques, including image filtering, data augmentation, and other algorithms (Neethu et al., 2020).

Skin detection is the method of recognizing between skin and non-skin colors. Given that human skin differs from person to person, it is impossible to establish a consistent method for its detection and segmentation. Although RGB is a commonly used color mode, its chrominance, brightness, and other non-uniform features make it unsuitable for skin identification. On YCbCr and HSV (HUE, and Saturation Values) pictures, skin detection is applied. The area of interest is concentrated on identifying hand motions and obtaining the most pertinent information. Skin-detection from the initial image is used to identify the hand region using some predetermined masks and filters, as illustrated.

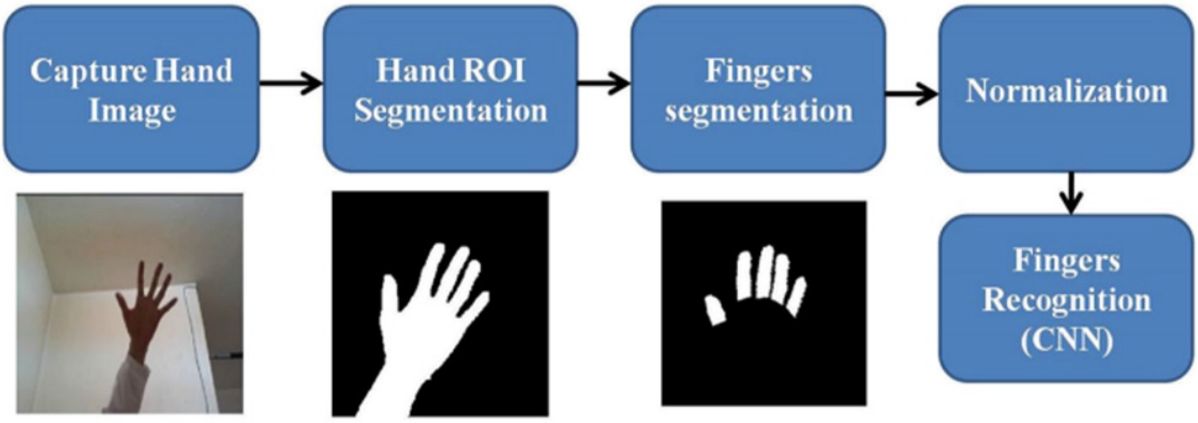


Figure 2.3 A proposed hand gesture recognition system using Rregion-of-Interest (Neethu et al., 2020)

Image resizes is a pre-processing step that involves the resizing of images by either expanding or decreasing image size. An interpolation algorithm exists that changes the image accuracy from one to another.

2.3 Models and Frameworks

2.3.1 TensorFlow

TensorFlow is a machine learning and artificial intelligence software library that is free and open source. Although it may be used for a variety of applications, deep neural network training and inference are given specific consideration. It is a neural network symbolic math library that works well with dataflow programming for a number of uses. It offers a variety of abstraction levels for model development and training. TensorFlow, like other deep learning frameworks, is built with a Python API rather than a C/C++ engine to improve speed. Because there is experimental support for a Java API that is not yet considered stable, it is not yet a solution for the Java and Scala communities (*TensorFlow*, 2022).

Tensorflow has the benefit over other frameworks in that it leverages both the Python and NumPy libraries. It compiles more quickly than Theano, a framework used for similar computing tasks as Tensorflow. Tensorflow can handle data and model parallelism and offers a tensorboard for display. However, Tensorflow moves a little more slowly than other frameworks.

2.3.2 Keras

Keras is a deep learning API written in Python that works on top of the TensorFlow machine learning framework. It was designed to enable rapid experimenting with machine learning tasks. When conducting research, it is vital to be able to move swiftly from idea to outcome. Keras is straightforward, which decreases the cognitive strain and allows you to concentrate on the most crucial aspects of the problem. Its flexibility originates from the notion of progressive complexity disclosure, which states that simple processes should be quick and straightforward, while arbitrarily sophisticated workflows should be feasible via a clear route that builds on what you've already learned (*Keras*, 2022).

Keras' core data structures are organized into layers and models. The sequential model is the most basic type of model and is made up of a linear stack of layers. More complex architectures can be obtained using the Keras functional API, which allows for the creation of arbitrary graphs of layers as well as the creation of models entirely from scratch via subclassing. Keras works with the Python, Ubuntu, Windows, and macOS operating systems. It benefits from being a rapidly growing framework and is likely to become a standard Python API for neural networks.

2.3.3 Caffe

Caffe is a deep learning framework that prioritizes expressiveness, performance, and modularity. It is being worked on by Berkeley AI Research (BAIR) and community contributions. As part of his PhD studies at UC Berkeley, Yngqing (2014) created the project. The BSD 2-Clause license governs the distribution of Caffe. The expressive architecture of Caffe promotes experimentation and innovation. Rather than being hard-coded, models and optimization are specified. You may train on a GPU system and then deploy to commodity clusters or mobile devices by setting a single flag. Its flexible coding promotes active development. Caffe's speed makes it suited for research and industrial applications. On a single NVIDIA K40 GPU*, Caffe can handle more than 60 million photos each day. That's 1 ms for inference and 4 ms for learning each image, and later library versions and hardware are even

quicker. Caffe has already been used to power university research projects, start-up prototypes, and large-scale industrial vision, speech, and multimedia applications (Yangqing, 2014).

2.3.4 OpenCV

OpenCV is a programming function package focused on real-time computer vision. The Apache 2 Open-Source Software License makes the library cross-platform and freely useable. The library includes approximately 2500 optimized algorithms that span a wide range of traditional and cutting-edge computer vision and machine learning methodologies. These algorithms can be used to, among other things, find related images in an image database, remove red eyes from flash images, identify objects, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to create a high-resolution image of an entire scene, and track eye movements. It identifies the environment and, among other things, creates markers to overlay with augmented reality. It supports Windows, Linux, Android, and Mac OS, and it has C++, Python, Java, and MATLAB interfaces (OpenCV, 2022).

The advantage of OpenCV is that it provides access to almost 2500 traditional algorithms. Many well-known companies, like IBM, Google, and Toyota, are heavy users of this software. Additionally, it offers effective algorithms, primarily for processing real-time programming. It has also been designed in a way that makes hardware acceleration and deployment on multi-core systems possible.

2.4 Architectures and Designs

Techniques based on sign language recognition fall into three basic categories: vision-sensor-based SLR systems, flex-sensor-based SLR systems, and hybrid SLR systems, which combine both types of systems. A high-definition camera, often a Kinect camera, is used to capture input data in vision-sensor based systems for input processing. By comparing it to sign language images that have been stored in the database, the captured input image is processed and recognized. The fundamental benefit of the vision-sensor-based approach is the elimination of sensors. The vision-based model simply needs an HD camera, while the sensor-based model is more expensive (Amin et al., 2022).

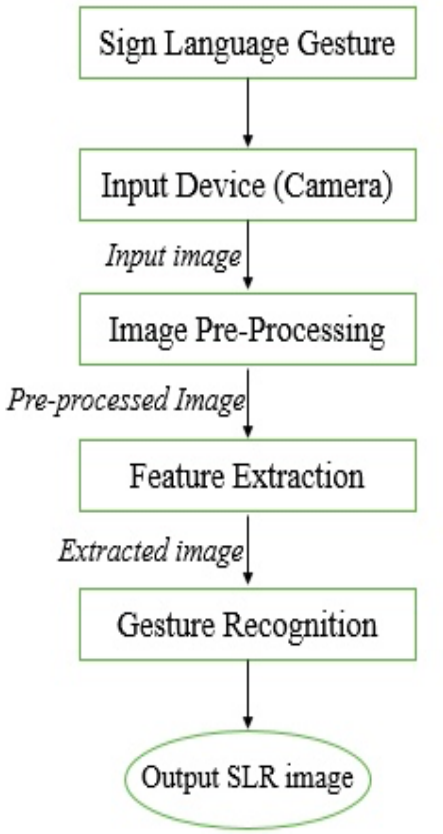


Figure 2.4 Processing steps for the vision sensor-based SLR system. (Amin et al., 2022)

Gesture recognition is achieved utilizing a smart glove equipped with numerous sensors in the sensor-based paradigm. These sensors include flex sensors for sensing figure bending, accelerometers for measuring angle and degree movement, and abduction and proximity sensors. The length of the sensor varies with finger length since these sensors are mounted to the user's fingers and the top side of the palm. Analog output is produced by the sensors. The degree of finger bending determines value variation.

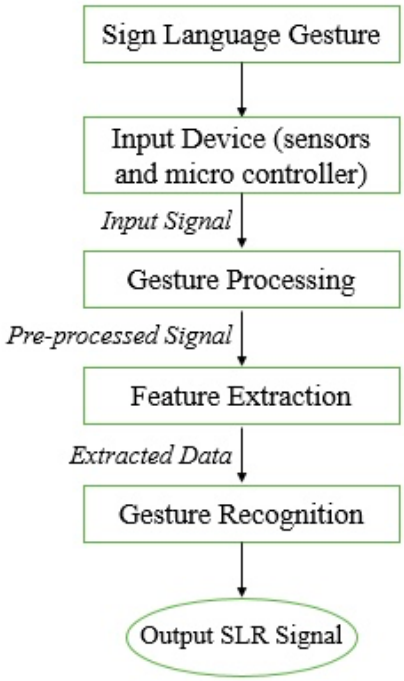


Figure 2.5 Processing steps for the sensor-based SLR system. (Amin et al., 2022)

A hybrid system for sign language combines models based on various sensors as well as vision-sensor technology. Therefore, raw gesture data is collected using both glove and camera-based orientation. For data accuracy, this technique eliminates errors. This form of data collection is uncommon due to the extensive and sophisticated computations required as well as the associated costs.

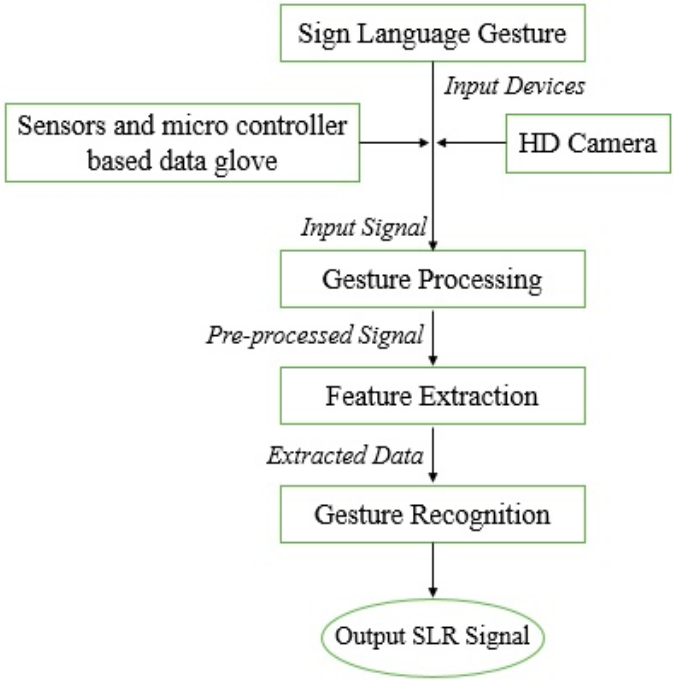


Figure 2.6 Processing steps for the hybrid SLR system. (Amin et al., 2022)

2.5 Algorithms

Many attempts have been explored in the past to overcome the challenge of sign language recognition. The methods range from employing specialized hardware furnished with sensors to detect hand motions to entirely software-based solutions based on Image Processing and Convolutional Neural Networks. Every solution has weaknesses, therefore select the approach that best matches the particular use case. There are several algorithms available.

2.5.1 Hidden Markov Model

This model is named after Andrey Andreyevich Markov, a Russian mathematician who was influential in the development of statistical theory. Since the late 1980s, this algorithm has been successfully applied in the analysis of biological sequences. It was first used in speech recognition. To track and model gestures, the Markov model is used (Starner, 1995). The basic idea behind HMM-based gesture recognition is to use multidimensional HMM to represent the defined gestures. According to Yang and Xu (1994), the parameters of the model are determined by the training data. The HMM-based gesture recognition approach is as follows:

a) **Define meaningful gestures** - Define meaningful gestures - Before using gestures to communicate, meaningful gestures must be defined. For example, if the gestures are to be used to edit text files, a specific vocabulary must be specified, and certain editor symbols must be provided in advance.

b) **Describe each gesture in terms of an HMM** - Each gesture is modelled using a multi-dimensional HMM. A gesture is defined by a set of N distinct hidden states and M distinct observable symbols in r dimensions. A transition matrix A and r discrete output distribution matrices B characterize an HMM;, I = 1,..., r. It is important to note that only the structures of A and B are determined in this step; the values of elements in A and B are estimated during the training process.

c) **Collect training data** - The training data is used to specify gestures in the HMM-based approach. It is critical that the training data be represented in a concise and consistent manner. Before training the HMMs, the raw input data is pre-processed. Because of the assumption of independence, each dimensional signal can be pre-processed separately. Pre-processing techniques such as the short-time Fourier transform and vector quantization are used in the prototype system discussed later.

d) **Train the HMMs through training data** - One of the most important procedures in an HMM-based approach is training. The model parameters are adjusted so that the likelihood P (0 |) for the given training data is maximized. So far, no analytic solution to the problem has been discovered. The Baum-Welch algorithm, on the other hand, can be used to iteratively re-estimate model parameters in order to achieve the local maximum.

e) **Evaluate gestures with the trained model** - To classify the incoming gestures, the trained model can be used. To classify isolated gestures, the Forward-Backward algorithm or the Viterbi algorithm can be used. Continuous gestures can also be decoded using the Viterbi algorithm.

2.5.2 Convolutional Neural Networks

One of the most well-known types of neural networks is the convolutional neural network. It is applied to high-dimensional data such as photos and movies. They are employed in image processing and recognition since they were specifically created to process pixel data. The fact that each unit in a CNN layer is a two- (or high-) dimensional filter that is convolved with that layer's input marks a significant difference. This is crucial when trying to learn patterns from high-dimensional input data, such as pictures or movies. According to Khan et al. (2018), CNN filters use parameter sharing to drastically cut down on the number of learnable variables and add spatial context by having a spatial form that is similar to but smaller than the input media (Khan et al., 2018).

A convolutional neural network contains three layers which are input, hidden, and output. Concealed layers are any intermediary layers in a feed-forward neural network that have their inputs and outputs hidden by the activation function and final convolution. Convolutional layers are found in the hidden layers of a convolutional neural network. Typically, this consists of a layer that performs a dot product of the convolution kernel and the layer's input matrix (Venkatesan & Li, 2018).

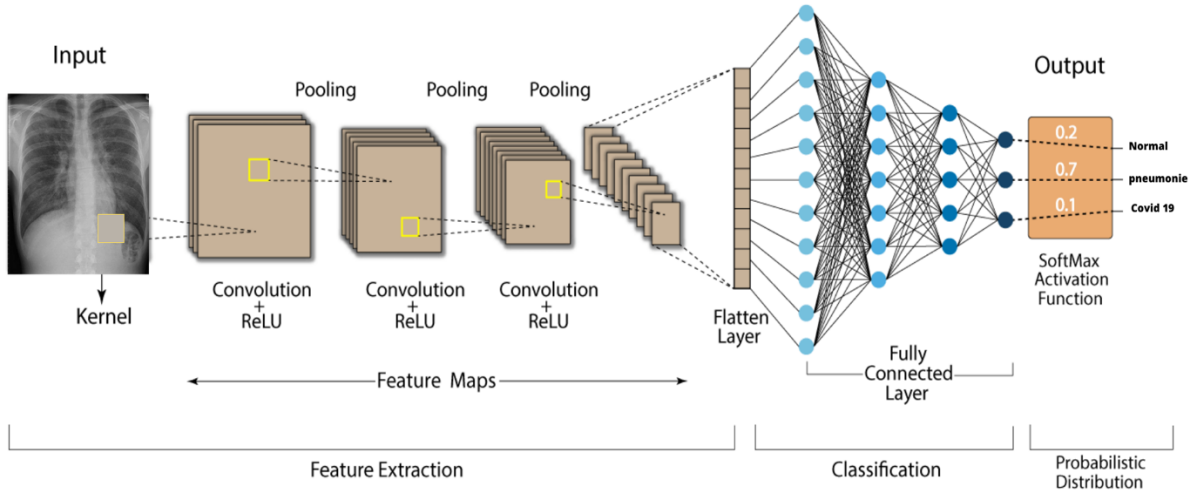


Figure 2.7 Convolutional Neural Network (Rguibi et al., 2022)

2.5.3 Bayesian Networks for Gesture Recognition

A Bayesian network also known as a belief network, or judgment network) is a probabilistic graphical model that represents a collection of variables and their conditional relationships using a directed acyclic graph (DAG). A set of directed edges links pairs of vertices, reflecting the direct interdependence of the variables. Cabaas De Paz et al. (2011) define the collection of nodes referring to X as its parents, represented by pa (X). To quantify the link between variables, conditional probabilities, generally tables (CPTs), are attached with each node, notably P (X | pa(X)). When the CPTs are concatenated, they depict the full joint distribution in a compact manner. A moist grass might result from either an operating sprinkler or rain. Rain has a direct impact on sprinkler usage (namely that when it rains, the sprinkler usually is not active). This problem may be modeled using a Bayesian network. Each variable has two possible values: T (for true) and F (for false) (for false).

The BNs framework has the ability to capture static behavior. Bayesian networks, on the other hand, may be applied in cases where the environment is intrinsically dynamic, such as a video sequence, which is our domain. Dynamic Bayesian networks (DBNs) are a well-known Bayesian network variation that may be utilized to represent the dynamics in this sequential scenario. Perhaps the most major problem of a Bayesian Networks approach is the lack of a broadly agreed mechanism for creating a network from data. There are several initiatives underway to build such approaches, but the literature shows that no clear winner has emerged to far. This weakness has two separate consequences, the first of which is the comparably high

effort required by the Bayesian Net's architecture. The fact that Bayesian Networks can only use causal influences that are identified by the person programming it leads to an issue.

On the other hand, neural networks have the benefit of being able to learn any patterns and aren't constrained by the person who programs them. However, this drawback is also a benefit compared to a neural network technique. There is no assurance that all domain-specific data is utilized during the training of neural networks, and it is also extremely difficult to look for specific features. But using Bayesian networks, it is simple to check if a specific aspect of the data is being taken into account and, if not, to force the inclusion of that aspect. As a result, Bayesian Networks may be relied upon to utilize every known aspect of a situation (Cabañas De Paz et al., 2011).

2.6 Empirical Literature

2.6.1 Sign language identification and recognition

A review of the literature reveals that several approaches have been taken to address the issue of sign language recognition using computer vision techniques and other methods. Sultan et al., (2022) distinguished and discussed the significance of sign language recognition and detection. According to the authors, sign language recognition simply entails all efforts made to the target with the goal of identifying the signer language. The goal of sign language recognition is to convert the signer conversion into tokens or signs. As a result, the two occur at different stages of the recognition process, with one feeding into the other to provide a solution. The paper emphasizes the importance of machine learning and deep learning algorithms in the recognition and identification processes (Sultan et al., 2022).

2.6.2 Human Posture Recognition in a Video Sequence

Human Posture Recognition in a Video Sequence Using Methods Based on 2D and 3D Appearance was published by Zhao et al. (2021). The paper describes utilizing PCA to detect silhouettes from a static camera and then using 3D to build posture for recognition. A silhouette is a sketch or cutout of an object's outline filled in with a solid color, generally black. The downside of this strategy is that it requires intermediary gestures, which might lead to ambiguity in training and, as a result, reduced prediction accuracy. The author also examined several ways of human gesture recognition, such as statistical, template-based, and deep learning-based methods, which served as the study's knowledge foundation. As you can see,

deep learning methods has joined the ranks in human gesture recognition as well (Zhao et al., 2021).

2.6.3 Text2Sign Translation

Text is an useful communication method in which information is encoded in letters that are formed into words and phrases using alphabets. Bantupalli and Xie (2019) worked together to create a vision-based application that translates sign language to text, allowing communication between signers and non-signers. From video sequences, the model extracts temporal and spatial data. The author employed Inception, a CNN, to recognize spatial elements (Convolutional Neural Network). Following that, the author trained a recurrent neural network (RNN) using temporal characteristics. The dataset was the American Sign Language Dataset.

There were three problems in the model presented in the study. One of the challenges the model had was with facial characteristics and skin tones. When tested against multiple skin tones, the model's accuracy decreased if it had not been trained and forecasted on a certain skin tone. The model also lost accuracy when faces were added; because signers' appearances fluctuate, the model ends up training inaccurate characteristics from the videos. As a result, the recordings had to be trimmed to show only motions that reached the neck. When the model's outfit was changed, she likewise performed poorly.

2.6.4 Usability in Sign Language Gesture Recognition through Computer Vision

When it comes to designing and implementing information systems, usability is critical. It enables users to benefit from system usage. Njagi Nyaga & Diko Wario (2018) tested an existing gesture recognition system for usability. The goals of this study were to determine the effectiveness of the system, its efficiency, and the satisfaction of the deaf participants with the system's use. The usability of the gesture recognition system was evaluated by 7 deaf participants in the study. The researcher used an observation data collection technique, which was followed by an interview facilitated by a sign language teacher.

The system was deemed effective and efficient by the participants. All of the participants appeared to be interested, eager to participate, and motivated by the system. The users' testing task was to finger spell the numbers one, two, three, and four. The data was then analyzed. One of the study's flaws was that it was difficult for most participants to place their hand on the rectangular box, which was considered to be the region of recognition. As a result, it is

appropriate to recommend a larger area so that participants do not have to struggle to place their hand in the region before beginning to perform the gestures.The system must be improved in order to recognize all of South Africa's alphabet and numerical characters. Additionally, for the system to be effective in a controlled environment such as a deaf classroom, a white background can be placed behind each computer user's seat to reduce the challenge of complex background noise and improve background subtraction (Njagi Nyaga & Diko Wario, 2018).

2.6.5 Sign Language Gesture Recognition through Computer Vision

In computer vision, motions are recognized using Microsoft Kinect, convolutional neural networks (CNNs), and GPU acceleration. Rather of generating intricate handmade features, CNNs may automate feature development. The author correctly identified 20 Italian motions. The prediction model was able to generalize on people and environments that did not exist during training, with a cross-validation accuracy of 91.7%. The study proved that even when users and their surroundings are not included in the training set, convolutional neural networks can reliably distinguish distinct signals of a sign language. The capacity of CNNs to generalize in spatiotemporal data can assist progress the field of automated sign language recognition research (Bronstein et al., 2015).

2.6.6 Real-Time Communication System for Speech and Hearing Impaired

Real-time communication systems have also taken advantage of advances in image processing, deep learning, and computer vision to provide text-to-sign language translation. Bohra et al. (2019) created a software-based solution that can be installed on any computer with adequate specs. It is also a two-way communication system, allowing not only speech and hearing-impaired people to communicate with normal people, but also normal people to communicate with speech and hearing-impaired people.

The system, on the other hand, was able to predict 17600 test images in 14 seconds, with an average prediction time of 0.000805 seconds and a 99% accuracy. A reliable recognition of sign language gestures can be performed using image processing techniques and a CNN model. The model demonstrates its robustness in a variety of lighting conditions and backgrounds because effective image processing techniques are used before the image is fed into the model (Bohra et al., 2019).

Ndungi and Karuga (2021) used convolutional neural networks to solve their problem of sign language recognition during their research. They captured hand gesture images naturally, converted them to grayscale, and used them to train a classification model that could recognize the English alphabets from A to Z. The letters that had been identified were then used to construct sentences. Using data mining techniques, this study revealed the most accurate sign language to English language prediction. This study can serve as a foundation or starting point for those conducting research with machine learning and convolutional neural networks (Ndungi & Karuga, 2021).

2.6.7 Architectures for Real-Time Automatic Sign Language Recognition on Resource-Constrained Device

Blair (2018) suggested a real-time automated sign language recognition architecture on a resource-constrained device, claiming that the major techniques of obtaining hand motion data for Automatic Sign Language Recognition are computer vision and direct capture employing gloves (ASLR). He says that computer vision is a better technique since it does not require specific equipment or the user to wear a separate gadget in order to utilize the system. Hand tracking and feature extraction are the two key challenges with vision techniques for ASLR.

Hand tracking requires that the signer's complete upper body be visible to the camera. When using 2D video from a normal camera, various limits on the background and the signer's attire are generally required. Many of these constraints can be solved by employing three-dimensional stereo cameras, albeit at a higher computational expense. Hand location (in relation to the body), form, and direction are all important factors in perceiving complete indications. Hand motion trajectories may also be beneficial for some classifications (Blair, 2018).

2.6.8 Summary of Empirical Literature Review

Table 2.1 Summary of Empirical Literature Review

NO	AUTHORS	RESEARCH TOPIC	APPROACH	FINDINGS	WEAKNESSES
1.	Ndungi, R., & Karuga, S. (2021)	A Sign Language Prediction Model using Convolution Neural Network	CNN Machine learning Algorithm using tensor flow	Accuracy score of a 99% value when run on epoch of 10.	The study did not use any Kenyan Sign Language dataset. It only used Modified National Institute of Standards and Technology data set (MNIST) from Kaggle.
2.	Stoll, S., Camgoz, N. C., Hadfield, S., & Bowden, R. (2020)	Text2Sign: Towards Sign Language Production Using Neural MachineTranslation and Generative Adversarial Networks	Neural Machine Translation with a Motion Graph (MG) to produce human pose sequences.	The system is capable of producing sign videos from spoken language sentences. Does not compete well with motion capture and avatar-based approaches.	It used low resolution of translation data. It suffered low output resolution and expressiveness obtained by motion capture and avatar-based approaches.
3.	Bantupalli, K., & Xie, Y. (2019)	American Sign Language Recognition using Deep Learning and Computer Vision	Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN).	In terms of CNN improve ments, using Capsule Netw orks instead of Inception may yield better results than Inception. One of the potential improvements would be to experimentwith different RNN architectures for the output of the poollayer.	The model also suffered from loss of accuracy with the inclusion of faces, as faces of signers vary, the model endsup training incorrect features from the videos. The model also performed poorly when there was variation in clothing.

NO	AUTHORS	RESEARCH TOPIC	APPROACH	FINDINGS	WEAKNESSES
					Maybe using a ROI to isolate hand gestures from the images would help accuracy.
4.	Bohra et al., (2019)	Real-Time Two-Way Communication System for Speech and Hearing-Impaired Using Computer Vision and Deep Learning	Convolutional Neural Network (CNN).	Predict 17600 test images in 14 seconds with an average prediction time of 0.000805 seconds with an accuracy of 99%.	Delays and classification errors experienced with letters m, and n due to its similarity.
5.	Njagi Nyaga & Diko Wario. (2018)	Sign Language Gesture Recognition through Computer Vision	OpenCV, Python	The objective of this study was to determine the effectiveness, efficiency and satisfaction of the deaf participants on the use of the system.	Challenges were experienced when placing users' hands in the region of recognition. It required a bigger area so that participants will not have to struggle to place the hand in the region before they start performing the gestures.
6.	Sultan, A., Makram, W., Kayed, M., & Ali, A. A. (2022)	Sign language identification and recognition: A comparative study.	Vision-based techniques, device-based techniques	Does a thorough comparative study on the algorithms, models, datasets involved in sign language translation.	The electronic-based systems were not giving users any comfort during the usage and communication with others. Gloves, sensors and other electronic devices restricted interaction with the system and had less accuracy from the many trials done.

2.7 Research Gaps

Despite the fact that several SLR models have been created, none of them, to our knowledge, can distinguish multiple Sign Languages. Simultaneously, in recent decades, there has been a tremendous demand for a trustworthy system that can engage and converse with individuals from other nations and SLs. COVID-19 Coronavirus is a worldwide pandemic that has caused many people to work and communicate remotely. Deaf persons must connect with and attend online meetings using a variety of platforms, including Zoom, Microsoft Team, and Google Meeting rooms. We must define and globalize a separate Sign Language since excluding deaf people and ignoring their presence would affect their psyche and have a detrimental influence on overall job progress, underlining the notion of "nothing about us without us.

Furthermore, Sign Language occupies a large space in all daily life activities such as TV sign translators, local conference sign translators, and international sign translators, making it difficult to translate all conference points to all deaf people from different nations, as each deaf person requires a translator of their own SL to translate and communicate with him. Many deaf athletes were invited to compete in the Deaflympics in 2010. They must engage and communicate with one another, as well as with everyone else in their home.

The current approaches that use hardware-based devices to map out sign language notations into inputs into models come with a lot of designing, fabrication and purchase requirements which adds up to the cost of implementation. Glove based devices are expensive to both designs, acquire the sensors and also interfacing the hardware to the respective model that does the pre-processing. This is contrary to the computer vision and Artificial intelligence-based models that are used to input images and videos with sign language notations. In the computer vision-based designs, the model just simply needs to be trained in classifying the different sign language notations and mapping their meanings to produce valuable outputs for those in need of such a solution.

It is also important to note that, as of the time this research was conducted, there had not been a study specifically designed to translate Kenyan sign language into a more readable friendly manner to promote another clear aspect of communication, namely the conversion of notations into readable English text with respect to the Kenyan context and all of its social, cultural, and technological landscape.

2.8 Conceptual Framework

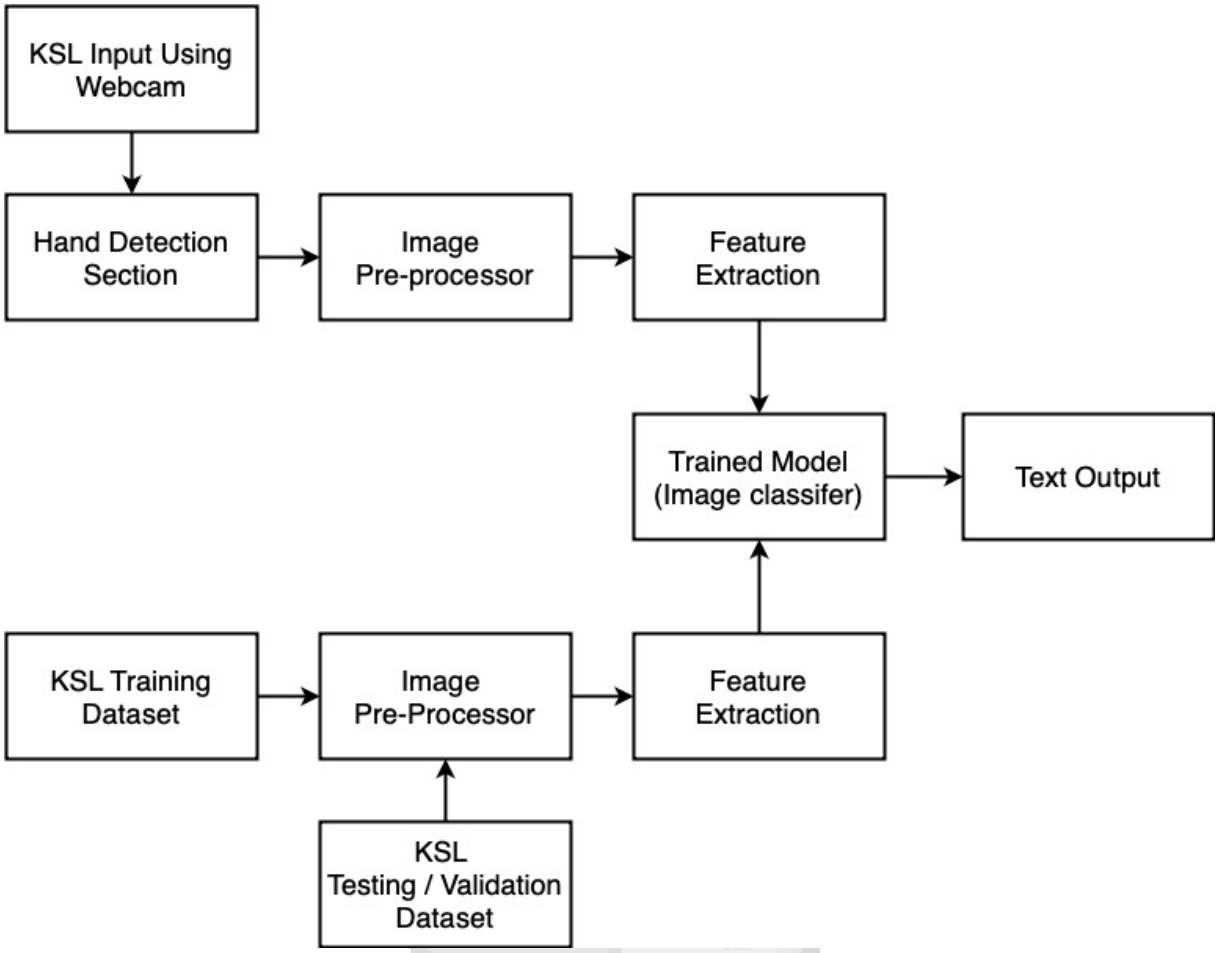


Figure 2.8 Conceptual Framework of the sign language recognition system

Figure 2.8 is A conceptual framework of the sign language recognition model with all the steps indicated to show the mappings to the model.

Chapter 3 Research Methodology

3.1 Introduction

According to Kothari (2004), "Research methodology" is defined as the scientific method for systematically solving a research problem. It includes all methods, techniques, and procedures used in conducting research. Computer vision research is primarily concerned with algorithms and models that are used to input and process visual images and videos in order to derive meaning from them, which can then be used as inputs to subsequent processes. The research methodology provides a description of the methods employed to perform the study. In essence, it describes the method of data collection, details on the range of participants, and the method of study inclusion. The research design for the study is also described, along with a detailed justification of the decision made. It's worth noting that it highlights the location, target demographic, and sample population of the study. Additionally, it outlines the system approach employed in the study for the article and highlights its advantages over other techniques already in use. The content also examines the research's ethical considerations.

3.2 Research Design and Philosophy

The study's goal is to create a model that can translate Kenyan Sign Language notations and symbols done by the deaf into English readable text. This entails capturing the deaf individual's notations using web cams, images or video inputs and then using computer vision algorithms to output English text that aids in communication. A computer vision algorithm analyzes the images and performs actual translations of the visual inputs into text. The study's qualities make it quantitative in nature (Kumar Bhayyalal Dubey & Kothari, 2022). The quantitative research design is used in research to portray the outcome by quantifying the data acquired from the photos captured. Similarly, the correlation research design to be utilized in the study may identify and predict a result with no supervision or control by the researcher.

3.3 Population and Sampling

3.3.1 Study Site

The study is going to focus on entirely secondary data sources. The data source is the KSL dataset available in Kaggle. Kaggle is nothing but an online site which is a subsidiary of Google. It allows users to find datasets they can use in building AI models, publish datasets, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges. The KSL Dataset in Kaggle provides images of signers which is

normally used in the training and classification for models. The Kaggle KSL repository has a total of 8930 Images containing users doing KSL hand notations that provides a source of useful training dataset for the model. The image collection was done by researchers who collected images to be used for their studies and also as a repository for sharing data collected by other researchers. It provides readily available secondary data for studies needed in Kenyan Sign Languages and other useful dataset needed.

200 signer photos are acquired from the Kenya National Association for the Deaf (KNAD) to supplement other notations for this project. KNAD is a non-governmental organization founded and run by Deaf people. KNAD was founded in 1986 and registered under the Societies Act in 1987; it is also an ordinary member of the World Federation of the Deaf. KNAD provides a good source due to the necessary environment, resources and the availability of key data and publication literature and books on fingerspelling, numbers and general Kenya Sign language notations. A good example is the 2011 edition of the basic words and phrases in Kenyan Sign Language publication booklet that contains images of KSL notations.

3.3.2 Target Population

The target population consists of all Kenyan Sign Language notations including finger spellings, numbers and sign language notations. Kaggle has a lot of datasets with quality images for this study. The dataset was fit for this study as it consisted of images taken in a controlled environment and rich in features used for the sign language detection. The KSL dataset contains notations on Church, Enough/Satisfied, Friend, Love, Me, Mosque, Seat, Temple and You. These are the common day to day notations used for communication. The image features include variations in skin tones of signers, garments worn for variation in color, lighting conditions from outdoor to room lighting and body size variations. The KSL dataset has a total of 8930 images files which are used for the model. The images collected from the Kenya National Association for the Deaf (KNAD) consist of notations in health and wellbeing like hospital, washing hands, feel, water and good among others.

3.3.3 Sampling

The study concentrates on using probability sampling to avoid biasing the samples to meet the goals of this research. The investigation employs simple random sampling of the images. This is due to the rule of statistical regularity, which indicates that if a random sample is taken, on average, it can have the same features as the population of the same pictures or sign language notation. Probability sampling supports this law (Kothari, 2004).

3.4 Dataset Acquisition

Secondary data obtained from the KSL dataset library from Kaggle contains images spanning from notations that communicate church, enough, friend, love, me, mosque, seat, temple and you. The images are downloaded as a zip file from the Kaggle repository. The data consists of original quality images of 720 by 1280 pixels in dimensions taken in a controlled environment. It contains images files of 8930 in total. The images were extracted to an output folder. The other data source is sourced from Kenya National Association for the deaf (KNAD) to supplement the data from Kaggle. The output folder storing the images is used as the source for the image data to the model being built. Three datasets were created from the photographs that were gathered. The model was trained on 80% of the photographs, tested on 10%, and validated on 10% of the images.

Table 3.1 Summary of the Image Dataset

Data Source Category	**Notation Class / Label**	**Number of Images**
Kenya National Association for the Deaf	Health and Wellbeing	170
Kaggle Site	Church, Enough/Satisfied, Friend, Love, Me, Mosque, Seat, Temple, You	8930
Total		**9100**

3.5 Model Construction

Model construction involves all the steps and activities involved in the design, development and testing of the model being used in this study. It highlights what key steps are necessary for the processing of the image datasets that is used to train the model in order to provide a fully trained KSL classifier. The model construction procedure is critical for providing a quick study of the algorithm structure, which aids in the modeling process. It also provides early responses to inquiries about the aspects of interest in the modeling process. In contrast to manual feature extraction strategies, deep learning models may automatically learn/extract features for picture categorization during training. The collected images are collected from an image path defined for the researcher.

3.5.1 Pre-processing

a) **Image Augmentation** – This refers to the regeneration and transformation of an image to inflate the dataset into more images to be used for the training of the model. It involves several augmentation forms such as flipping which flips the image either horizontally or vertically. Rotation involves rotating each image into different angles of degrees. Zooming is also a form of augmentation that tries to focus on the particular region of interest of the image by bringing it closer or further away to provide variations of the image.

b) **Image Standardization** – This process involves providing uniformity to the input images in the aspects of image resolution, dimensions and sizes. This uniformity allows the model to be able to provide consistent results and outputs that the users can appreciate. The image size expected for the model in 320 by 320 pixels. This is critical in limiting variability within the dataset and allowing the photos to suit the model's input form.

3.5.2 Feature extraction

Feature extraction is usually used when the original raw data is very different and we cannot use it for machine learning modelling when the raw data is being transformed into the desired form. It is a method for creating a new and smaller set of features that captures mods of the useful information of raw data. In real world machine learning problems, there rarely is refined data in the form of xml or csv format. So key features need to be extracted from the raw data to either text, images, geospatial data, date and time and sensor data among others. This section involves the following subprocesses.

a) LabelImg is a lightweight and easy-to-use image annotation tool to label object bounding boxes in images. The images are annotated by the software to indicate their respective translation of the notations portrayed in the image. If an image contains a signer doing a thank you notation, the area of the image with the hand notation is mapped out and labeled as thank you with the labelImg library. After this, the labelmap is then updated for each Kenyan Sign Language notation.

b) OpenCV (Open-Source Computer Vision Library) - It is primarily concerned with image processing, video recording, and analysis, including features like as face detection and object detection. The OpenCV engine analyzes images to detect sign language notations. It scans each input image and video from the live input and detect for the key areas of

interest in the input, in this case the Kenyan sign language notation being performed by the signer.

3.5.3 Model development and training

The CNN model being built and trained to classify Kenyan Sign Language notation uses TensorFlow for the training of the model. TensorFlow's Object Detection API is an open-source framework built on top of TensorFlow that makes it simple to build, train, and deploy object identification models. Convolutions in the CNN model uses batch normalization to reduce overfitting and increase the generalizability of the model. In addition to considerable efficiency improvements, the approaches have been exploiting enormous picture databases to lessen the demand for large datasets. The model development, training and deployment is carried out in Jupyter notebook platform which is suitable for the model development as provides a powerful integrated environment for the model development. The figure 3.2 shows the execution of the model at a glance.

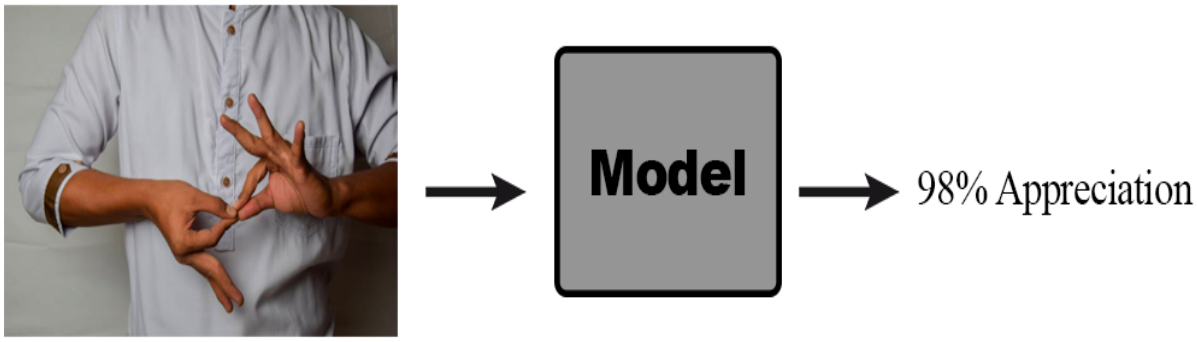


Figure 3.1 Image Classification Process (Kiyegga 2020)

3.5.4 Model Testing

To evaluate the model based on the fundamental functioning characteristics, functional testing is done on it. This helps in determine the operation of the model based on the set requirements and guidelines and the objective of this research. 10% of the images are used for the testing of the model with this regard.

3.5.5 Model Validation

The validation of the model is carried out by 10% of the image dataset. A confusion matrix is used in the implementation phase from which several metrics are derived for the evaluation of the model's performance.

3.6 System Development Methodology

The software methodology used in this study is a Prototyping based methodology. This methodology is chosen to build the Convolutional Neural Network (CNN) model. Evolutionary prototyping guides the different phases of the model's development. Prototyping is needed in the development of the CNN since some parameters, such as the number of training steps, is experimental, necessitating iterative improvement. This style enables for the gradual addition or removal of features, as well as real-time development through experimentation.

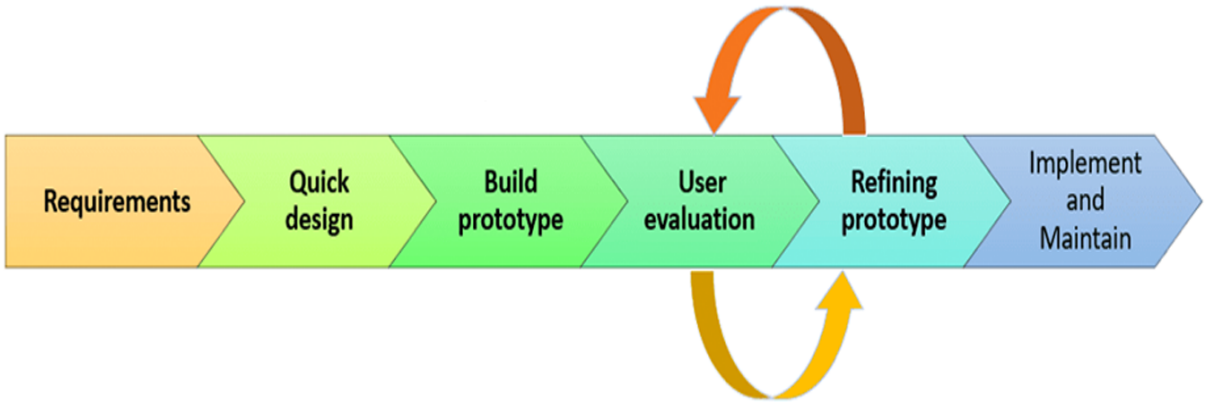


Figure 3.2 Prototyping Model Phases (Martin 2022)

a) Requirements gathering and analysis

The requirements for the Kenyan Sign Language recognition model is developed using several hardware, software and programming tools. It also involves the solicitation of stakeholder stories for this study. Both hardware and software components are part of the system architecture as well as tools that are used in the development of the model.

i) **Hardware components** - Consist of the camera for capturing images of quality sign language notations done by an individual. A laptop for developing the model for this study. The laptop has 16 Gigabytes of memory and Corei7 processor to handle the processing power needed to train the model once complete.

ii) **Software components -** This consists of the windows 10 operating system for hosting the applications and programs needed for this model development. The object detection algorithm is utilized to identify sign language notations. The OpenCV is an opensource application that is used for capturing and structuring images in the model. The images captured by the camera are organised and pre-

processed by the OpenCV to be fed into the model as both training data and validation data. The labelImg is an opensource application that is used to label images with their respective translation in text.

iii) **Programming Tools -** Jupyter Notebook is an interactive computing tool that runs on the web. Live code, mathematics, narrative prose, visualizations, and other coding activities are all combined in the notebook. Github is used to manage version control and code releases. It is a Git-based Internet hosting service for software development and version control. It includes access control, bug tracking, software feature requests, task management, continuous integration, and wikis for each project, in addition to Git's distributed version control. Python programming language is predominantly used in the development of this model.

b) Quick design

Object Oriented Analysis and Design is used in this research. This ensures ensure proper capturing of the requirements. Unified Modelling Language (UML) diagrams are used in the system design since they provide a clear model that lessens system complexity. These models include:

i. Entity Relationship Diagrams (ERD) that are applied in the study during database design to show entities, attributes, and relationships.
ii. Use-case diagrams are used to show how the actors and system processes interact.
iii. Sequence and collaboration diagrams are used to show object interactions arranged in time sequence.
iv. System Sequence Diagram (SSD) helps to track how use cases functions are performed inside the system. Also, to model the software in concern with how the system interacts with the events.

These diagrams are of necessity in showing the process, information flow and relationships between the entities in the model.

c) Initial user assessment

In this phase, are involved in the to provide the assessment for the product or prototype to be made. It is normally needed to provide the strength and shortcoming of the functioning model. All the remarks and idea are gathered to inform on the development approach as well. The design of the model and the algorithm used are evaluated to determine if its effective enough

for the detection. This initial evaluation involves using 10 sign language notations that are recognised into text and see the model's performance in detection accuracy.

d) Refining prototype

This phase involves looking at the output of the prototype and refining it according to the various feedback and suggestions. These adjustments involves finetuning the overfitting and underfitting of the machine learning model to make sure that the detection is exactly as it is expected. The sign language notations in the images should be accurately detected and correlated to the textual meaning of the label mapping. the quality of the images taken can be adjusted for clarity especially by taking photos of the subjects doing the sign language notation in a clear background.

e) Implement product and maintain

The Kenyan Sign Language recognition model is built in this phase. The development environment uses Jupyter Notebook, an online integrated development environment that enables Python script execution. The phases of the implementation process is as follows:

i. Clone and setup the object detection repository and configure it properly into the working laptop.
ii. Collect images using code and OpenCV
iii. Setup the labelling of images using labelImg.
iv. Label the images accordingly by mapping the respective sign language notations to the text meanings.
v. Train the model.
vi. Update checkpoints of the project.
vii. Run the program to detect Kenyan Sign Language.

3.7 Quality and Dependability of Research

Quality alludes to the degree to which the information precisely gauges what it was planned for. Dependability centers around the degree to which the information assortment strategy yields similar discoveries whenever repeated by different analysts in the particular field. Proper system design and analysis is done to carefully provide this reliability. The requirements of the study are gathered correctly and thus facilitate the reliability criteria. Quality images are collected for the model for clear sign language detection by the algorithm. Proper labelling of the images are done for accuracy of the sign language notation to text mappings. Proper coding

techniques and code labelling are employed to enhance quality as well. Finally, the project is also subjected to an ethical review for quality purposes.

3.8 Utilization and Dissemination of Research Results

The research results are utilized by any stakeholder of the country who may need to use the model to translate Kenyan Sing Language into english text for to foster communication with the deaf. The target audience are any member of the public spanning from ‘mama mboga’ in the kiosk to officials both in the public and private sector.

The setting for the use of the model is any engagement involving the deaf in the community. This clearly facilitates the utilization of the model and the research results in interacting with the deaf. This research also helps in the formulation of policies that may include the use of technological models and gadgets that facilitates communication with the deaf e.g., not to use improper and incorrect mappings of sign language to text notations in order to hoax individuals. This can be considered an offense.

3.9 Ethical Considerations and Issues

This study supports honesty and veracity and forgoes all instances of plagiarism, data fabrication, and falsification. In terms of technical responsibility, the topic relates to the technical requirement of having a sign language translation tool to facilitate communication with the deaf. The study's findings are technically significant and are anticipated to help both deaf people and scholars in the realm of contemporary technology. Images from high-end cameras and cellphones are used as the primary source of data for this project in order to record signs that people make in sign language. Books on sign language that have been published and include the signs' respective definitions and explanations are the key source of secondary data for this study. The data collection process is done in accordance with the permission for educational research to ensure ethical conduct. Before beginning the study, the Strathmore University ethics review board is consulted to make sure that ethics is safeguarded throughout the duration of its implementation. All previous authors' research work has been appropriately cited to acknowledge their contributions. A research permit has been obtained from the appropriate authorities, as well as a document proving the originality of this study.

Chapter 4 System architecture and design

4.1 Introduction

System design plays a critical role in the implementation of systems and applications. It's used to provide a high-level blue print of the various elements and components that hold the entire system together to achieve its end solution to provide a holistic product. System architecture portrays the information flow, interaction, transformation, wireframes, use cases and as well as collaboration of the elements. The common notation used in system design and analysis is the Unified Modelling Language (UML). UML is the standard language used in software engineering to showcase an abstract high-level view of the different entities and relations and their environment in the real world. This chapter touches on the requirement analysis then to the designs and architecture of the sign language model to be implemented.

4.2 Requirement analysis

This research develops a model to translate sign language notations into readable text to the hearing population. Based on the objective of this research, this section outlines the various requirements to be fulfilled in the proposed model.

4.2.1 Functional Requirements

i. The system should accept series of images from the camera.
ii. The system should transform each of the images and extract relevant features.
iii. The system should identify the sign language notations of the people in the image.
iv. The system should classify the notation from the image accordingly.
v. The system should provide an output on text recommending the meaning of the sign language notation.
vi. The system should display the confidence level of the sign language notation and the image on the digital screen.

4.2.2 Non-functional requirements

i. Usability

The application is used by the hearing population in decoding sign language notations to clearly understand the hearing disabled people. The application is designed for easy usage and interaction. This allows users to fully benefit from the applications capabilities and features.

ii. Scalability

The application should keep improving on the accuracy levels of recognition of the sign language notations. This can guarantee that the program responds quickly to fresh data.

iii. Reliability

The model performs its intended functions adequately for a specified period of time a defined environment without failure. Reliability aims to make sure that the model operates and give consistent results without any form of shortcomings.

iv. Availability

The availability is one of the critical requirements since without it, the model seizes to be of benefit to the users of the model. The sign language translation model should be able to be used at all times when it is needed.

v. Data integrity

One of the main concerns it to make sure that the correct Kenyan sign language notations are mapped to their respective meanings and text. This is to avoid any form of confusion and misinterpretation during the translation process of the model. This model should also do proper transformation to the input images without distorting it or losing its clarity.

vi. Adaptability

The model should be able to adapt to fresh datasets from various environmental situations. These conditions could be background lighting conditions, clothes variations worn by the signer, distance of the image capture from the signer to the camera. This allows the model to adapt to the various conditions of the image capture.

vii. Configurable

The system implements image processing using object detection through a convolution neural network. It may require reconfiguration of the weights to attain higher accuracy levels in the translation of sign language notations. Therefore, the system should have an easier method to enable quick reconfiguration and achieve better results.

viii. Security

The system constantly captures images of people in a specific location which intrudes personal privacy. Information captured needs to be stored behind a system with strong security measures to maintain the privacy of such data. In addition, reconfiguration of the system requires authorized access to increase the security of the system.

4.3 System Architecture

4.3.1 System Model Architecture

The system architecture showcases the entire layout of the sign language translation model including the main components that constitute it. The model uses an input camera or webcam as a source of input for images. The images undergo the acquisition step that acquires the images successively in an organized and orderly manner sequentially. This is critical so that the successive model is able to translate each sign notation into a syntactical and readable text. The image is then pre-processed. This step involves resizing, noise removal, segmentation and morphological adjustments. The image segmentation step involves selecting and marking portions of the images with areas on interests. The feature extraction aims at extracting the hand gestures. The classification component recognizes the gestures and compares it to the text dataset and outputs the recognized sign in text. Figure 4.1 shows the architecture.

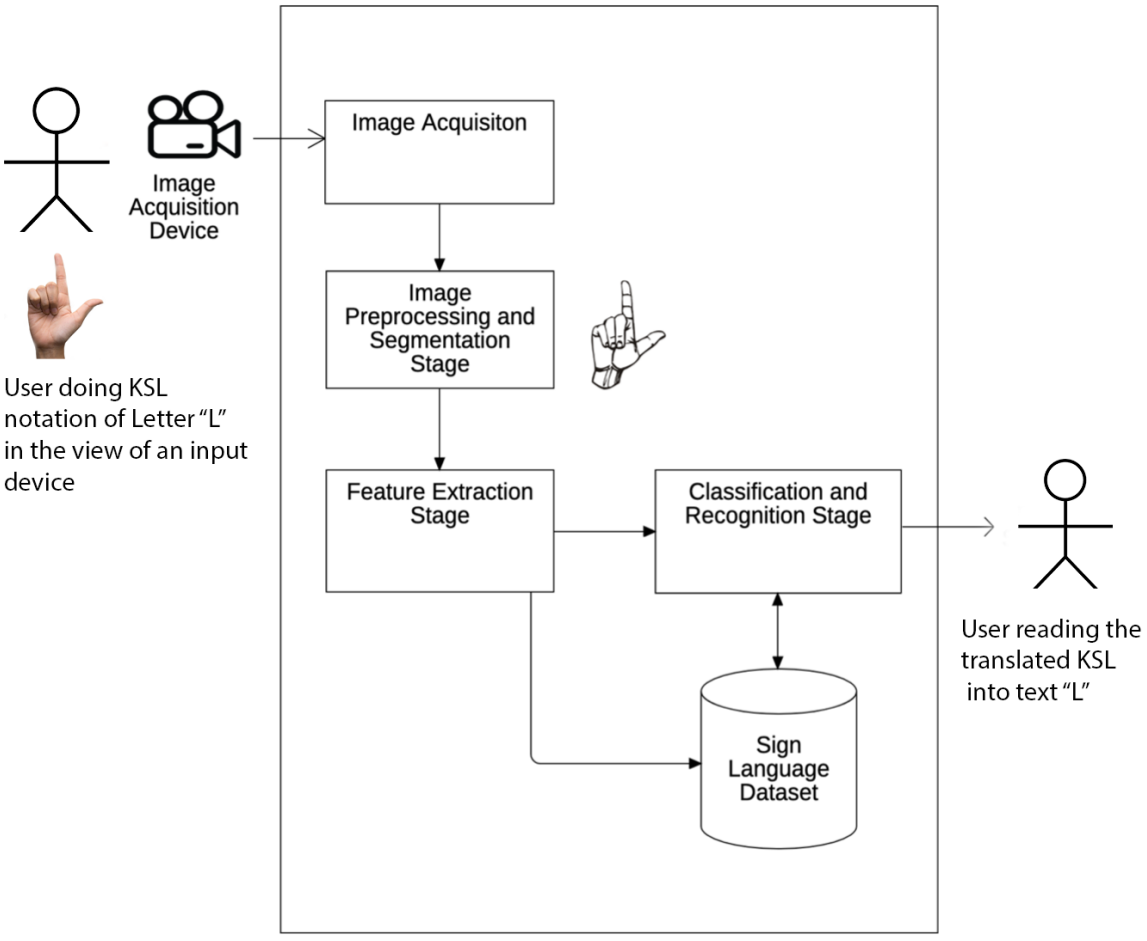


Figure 4.1 System Architecture

4.3.2 Use case diagram

A use case diagram is a design tool and techniques used in the representation of the user's interaction with the system at a glance. it shows a high-level design of the model and how it is being accessed, used and maintained by the actors involved. The proposed system contains three (3) actors namely, the developer, administrator and the hearing user and the hearing impaired (deaf/dumb) user.

i. **Developer** – The main role of the developer is to do updates to the model. This includes training, testing and carrying out maintenance to the model. The developer can also add other sign language notation classification in as an update to the system.

ii. **The administrator** – the role of the administrator is to manage users of the model. This involves creating user accounts, update users and delete users of the model.

iii. **User** – the user uses the model to identify the meanings to the sign language notations done by a hearing-impaired individual. Figure 4.2 summarizes the concept.

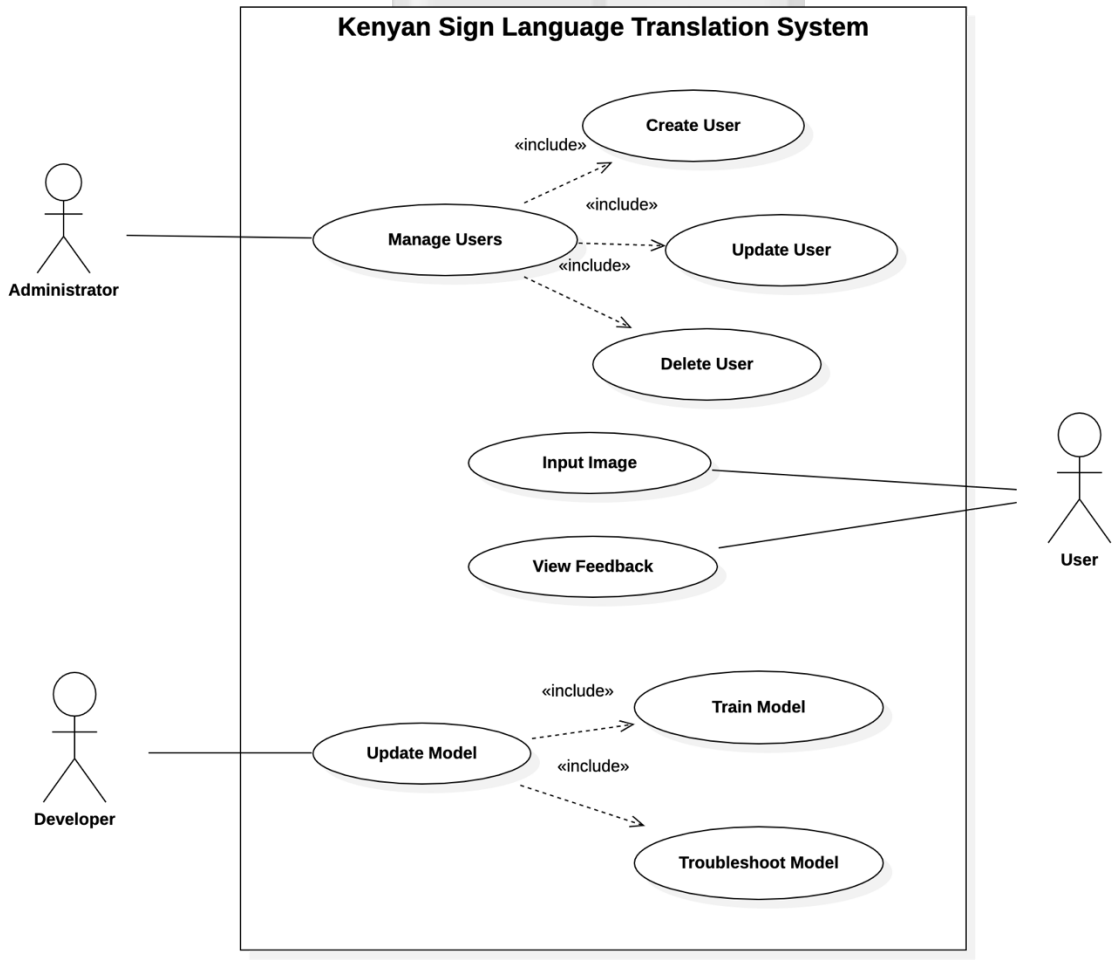


Figure 4.2 Use Case Diagram

Use case descriptions

Table 4.1 Use Case Description

Actors	Use Case	Description
1. **User:** The case where a user wants to know the meaning of a Kenyan sign language notation done by another user.	1.1. Input Image	1.1.1. The model launches the webcam to get live images. 1.1.2. The webcam is positioned in front of the user to capture the sign language notations. 1.1.3. The model pre-processes the image and does the classification.
	1.2. View Feedback	1.2.1. The model returns the results regarding the sign language notation in readable text. 1.2.2. The user views the translated results.
2. **Developer:** The case where a developer wants to update the model.	2.1. Update model	2.1.1. The developer can update the model by making changes in the existing parameters. 2.1.2. The model is saved with the changes made by the developer.
	2.2. Train Model	2.2.1. The model should be able to be trained by the developer. 2.2.2. This training should allow it to accept new sign language notations. 2.2.3. The developer should be able to save the trained model.

Actors	Use Case	Description
	2.3. Troubleshoot Model	2.3.1. The model occasionally may provide errors in capturing input images and video stream.
3. **Administrator:** The case where user accounts are managed	3.1. Create User	3.1.1. The administrator creates user accounts for new users. 3.1.2. The account creation captures usernames, passwords and email addresses.
	3.2. Modify Users	3.2.1. The administrator should be able to access user accounts and modify its parameters. 3.2.2. The modifiable parameters include account names, passwords and email.
	3.3. Delete Users	3.3.1. The administrator should be able to delete existing users in the model. 3.3.2. Once a user account is deleted, it seizes to exist in the model and cannot be accessed.

4.3.3 Sequence diagram

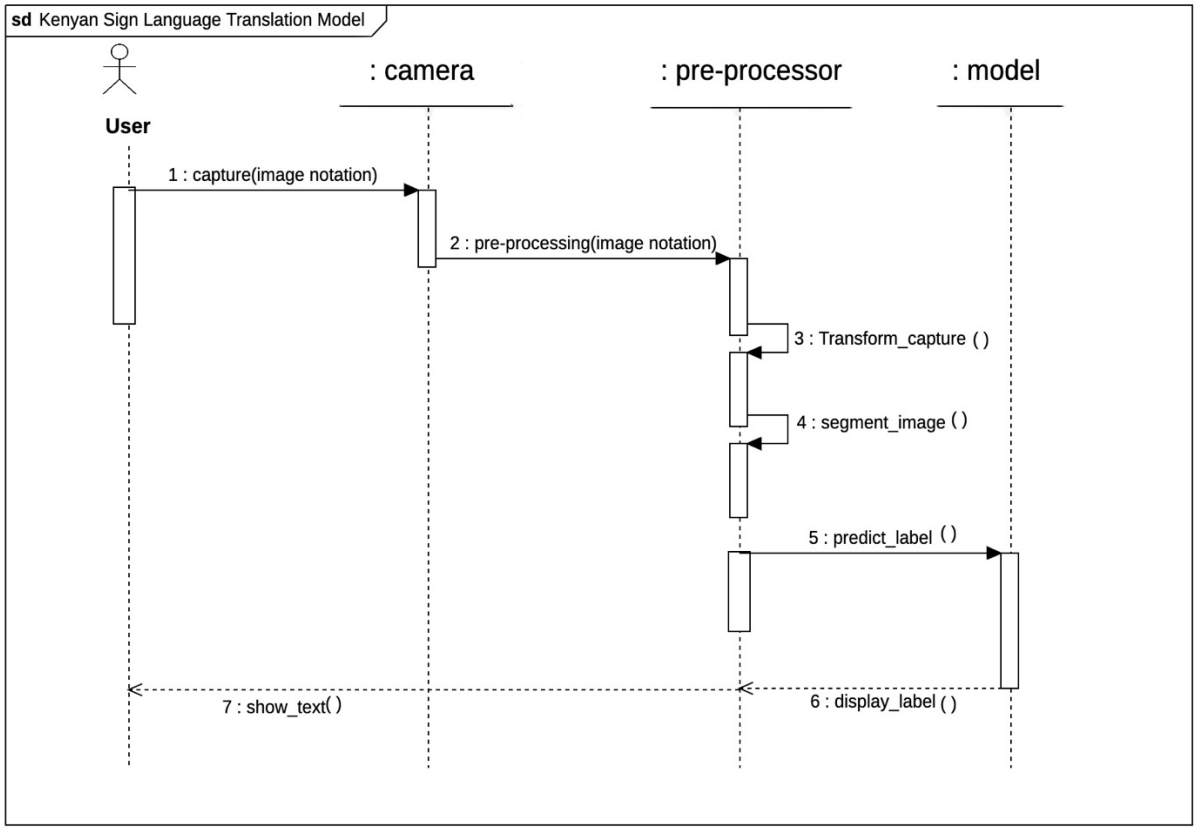


Figure 4.3 Sequence Diagram

The sequence diagram in figure 4.3 showcases the sequence of interactions between the user and the difference instances of the model in its entirety. It shows the different messages that are sent to the respective instances and the replies received as well.

i. The capture () message initializes the camera to start accepting visual inputs in the form of video and images. This camera is pointed to the target doing the Kenyan sign language notation.

ii. The pre-processing () message does the intermediary analysis of the input stream captured. It involves organizing the image using the OpenCV framework.

iii. The transform capture () message gets the images into streams of with each show casing the notations in each successive image. It simply queues the images in line to be worked on in the coming stages.

iv. The segment image () message highlights portions of the image into distinct notable objects. In this case, the objects being highlighted are the hands and fingers doing the sign language notations.

v. The predict label () message determines the meaning of the Kenyan sign language notation by cross-referencing the mapping to the database which has a trained model to determine the meaning.

vi. The display_label () message checks for the appropriate textual output to be sent to the user with the confidence level of the translation.

vii. The show_text () message displays the message of the translation to the user in the screen.

4.3.4 Context diagram

The context diagram depicts the model's high-level data flow and how users interact with it. Figure 4.4 depicts the numerous inputs and outputs from the categorization, with the user and the developer as the primary users of the model.

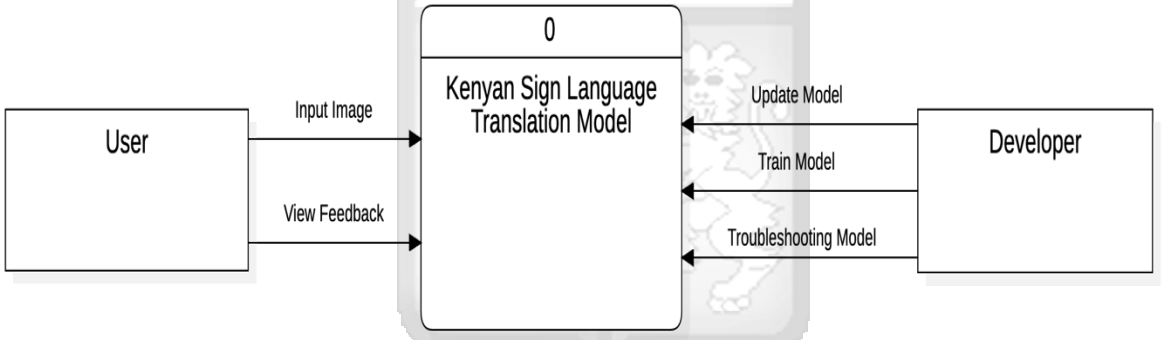


Figure 4.4 Context Diagram

The context diagram contains three key elements, the user the main model and the developer. The users in interact with the model by first capturing input as an image to be fed into the model (Kenyan sign language model). Feedback is then gotten from the model. The feedback is in the form of a translated text that is readable for the users to understand the meaning of the sign language notation.

The developer interacts with the system through updates to the model. Updates to the model entails expansion of the model to accommodate new sign language notation. Training the model allows the model to be able to accommodate and classify new sign language notations correctly. Training the model with training data effectively prepares the model to receive input maps the sign language notation to the text meaning. Troubleshooting the model involves code configuration and modifications as well as maintenance to the model.

4.3.5 Level 0 Data Flow Diagram (DFD)

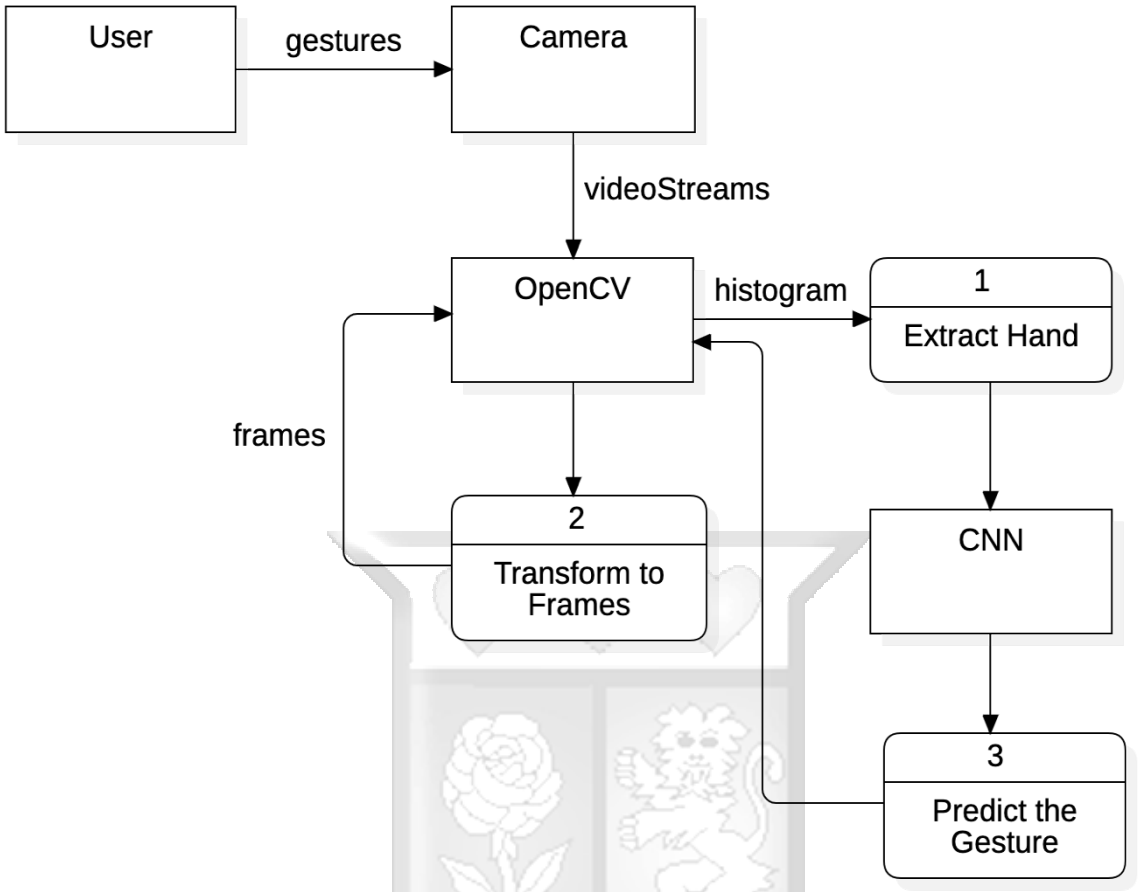


Figure 4.5 Level 0 Data Flow Diagram

The DFD is also referred to as a bubble chart. It is a basic graphical formalism that can be used to depict a system in terms of the system's input data, various processing performed on these data, and the system's output data. It depicts the information flow for any process or system, as well as how data is handled in terms of inputs and outputs.

Figure 4.5 showcases how data flows from the user into the other entities of the entire model. Data from the user flows to the camera as inputs in the form of gestures and Kenyan Sign Language notations (KSL). The camera outputs the video streams and images to the OpenCV framework for pre-processing. The key step is the transformation of the input video streams into frames. The OpenCV sends outputs for processing in the hand extraction and also to the frame's transformation processes. The CNN which is an object detection algorithm does the image classification to produce the desired output to the prediction process. The Gesture prediction process is able to predict the sign language notation received from the model.

4.3.6 Entity Relationship Diagram

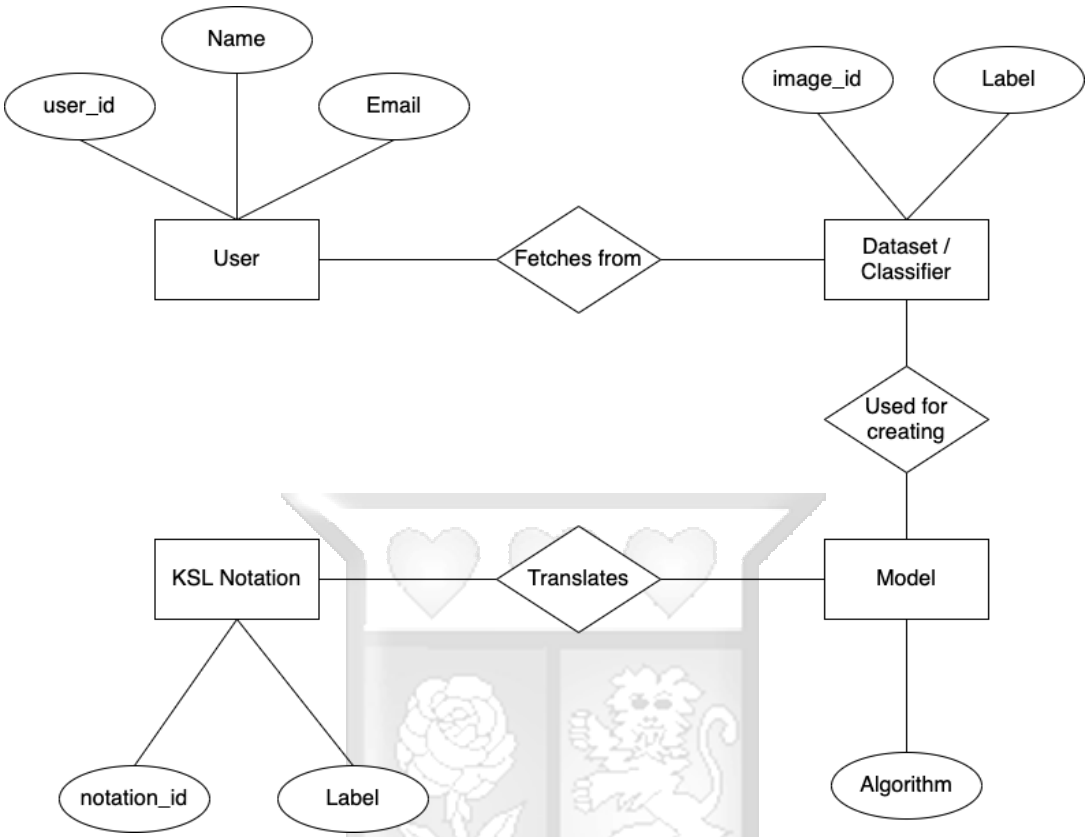


Figure 4.6 Entity Relationship Diagram

The figure 4.6 showcases the entity relationship diagram for the Kenyan Sign Language Translation Model.

i. The user entity is described by the user_id, name and email attributes respectively. The user_id denotes the unique identity of the user in the model. The name denotes that name of the user in full. The email attribute denotes the email address of the user.

ii. The dataset / classifier entity contains the dataset of the notations of all the Kenyan signl language used for the model. The notaions are attributed by labels that holds the meaning of each notation and also the image_id which is the image identifier of the notation.

iii. The model entity refers to the model that does the translation. The model is attributed by the algorithm which in this case is a CNN that does the deep learning and translation.

iv. The KSL Notation entity refers to the Kenyan sign language notations to be translated. These notations are described by the notation_id that uniquely identifies each notation in the model. The label description holds the meaning of the notation for the users to comprehend.

4.3.7 State Chart Diagram

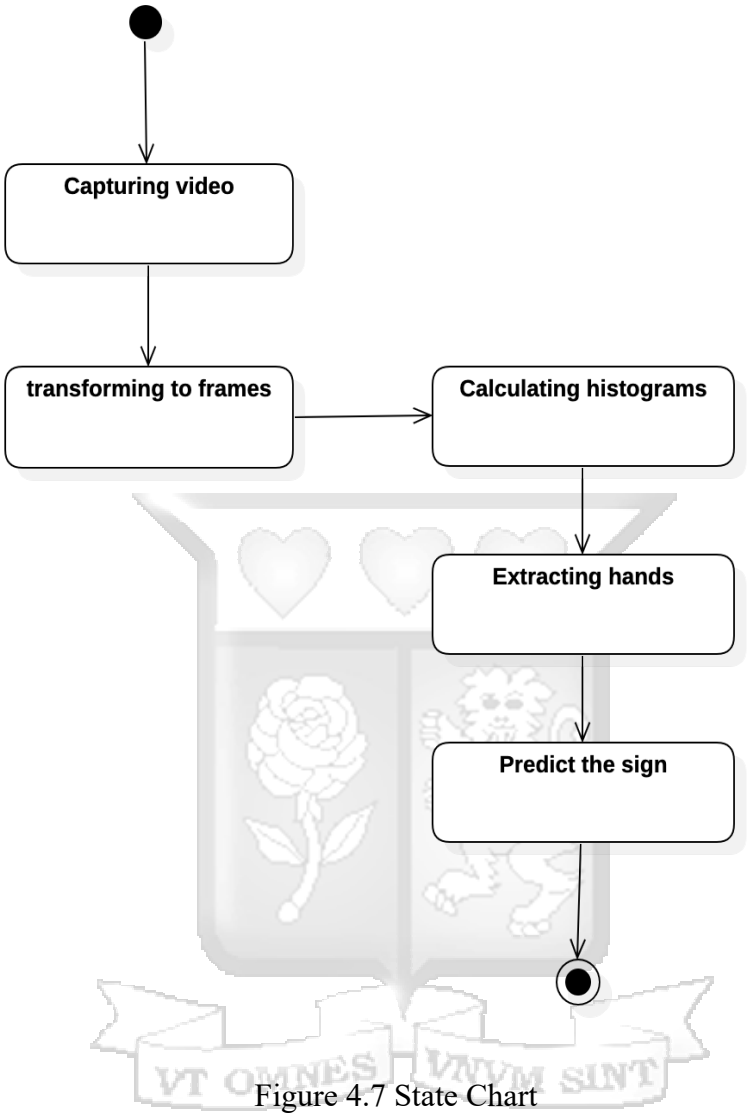


Figure 4.7 State Chart

The state diagram begins from capturing video. It involves capturing the images and videos through the camera. The next state is transforming to frames which breaks the inputs into frames. Each frame is then fed into the next frames for analysis. Calculating histograms involves mapping the respective images. Algorithms and pre-processors segment the images to provide clarity that allows the outputs to be fed to the next state. Extracting hands state involves isolation the hand gestures from the images that have been pre-processed. These hand gestures are what holds the key sign language notation that needs to be translated. The predict sign state involves the determination and mapping of the respective hand gestures into its corresponding textual meaning that outputs to the user to read in order to understand the meaning of the gesture.

4.3.8 Class Diagram

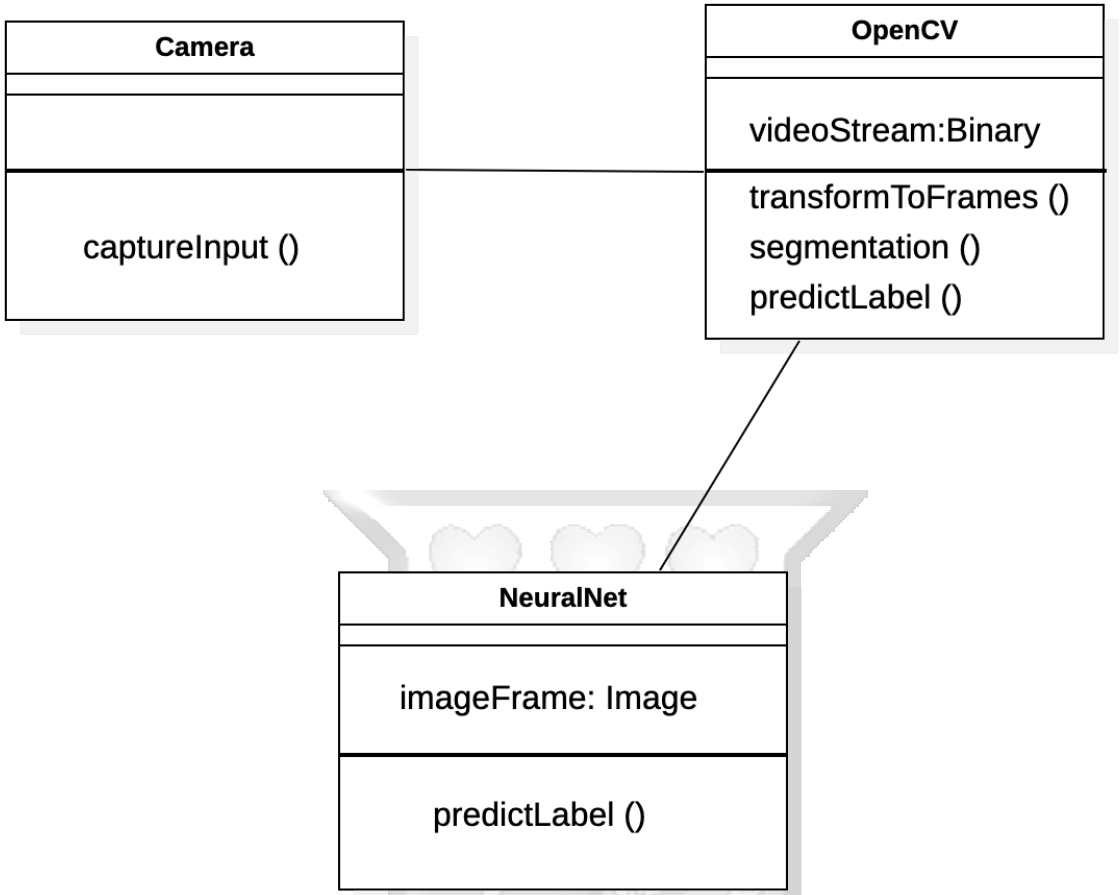


Figure 4.8 Class Diagram

Class diagrams use design features like as classes, packages, and objects to represent class structure and contents. Class diagrams represent the several perspectives that must be considered while developing a system: conceptual, specification, and implementation. Figure 4.7 showcases the different classes that are needed for this model. They include the camera class, OpenCV class and the CNN Class for the model training and classification.

i. The camera class has one method to use to capture input called captureInput ().
ii. The OpenCV class has an object called videoStream with type binary, meaning a video exists or not. The methods used by the OpenCV class include transformToFrames (), segmentation (), and predictLabel () respectively.
iii. The CNN has an object called imageFrame which passes and image as s parameter. The predictLabel () method is used to predict the sign language notation seen in the image to be output as readable text to the user.

4.3.9 Wireframes

A wireframe is a graphic or group of images that depicts the functional parts of a website or application, and is often used to outline the structure and functioning of a site. A wireframe is a visual guide that illustrates the skeleton framework of a website or application. It is also known as a page schematic or screen blueprint. A wireframe is a schematic or blueprint that can help you, your developers, and designers think about and communicate about the structure of the software or website you're creating. The Kenyan sign language translation model has a graphical interface that is used to interact with the users using the model. This section showcases the interfaces and how they are expected to look in the final product.

The registration screen is what users use to create user accounts to the application. This allows each user to have their own separate account that manages their profiles. The wireframe on figure 4.9 shows the registration page with the application title and the slogan to introduce the users on the aim of the application which is to translate Kenyan sign language notations into English. The fields indicated in the page include the username, email and password as well as a command button input for signing up.

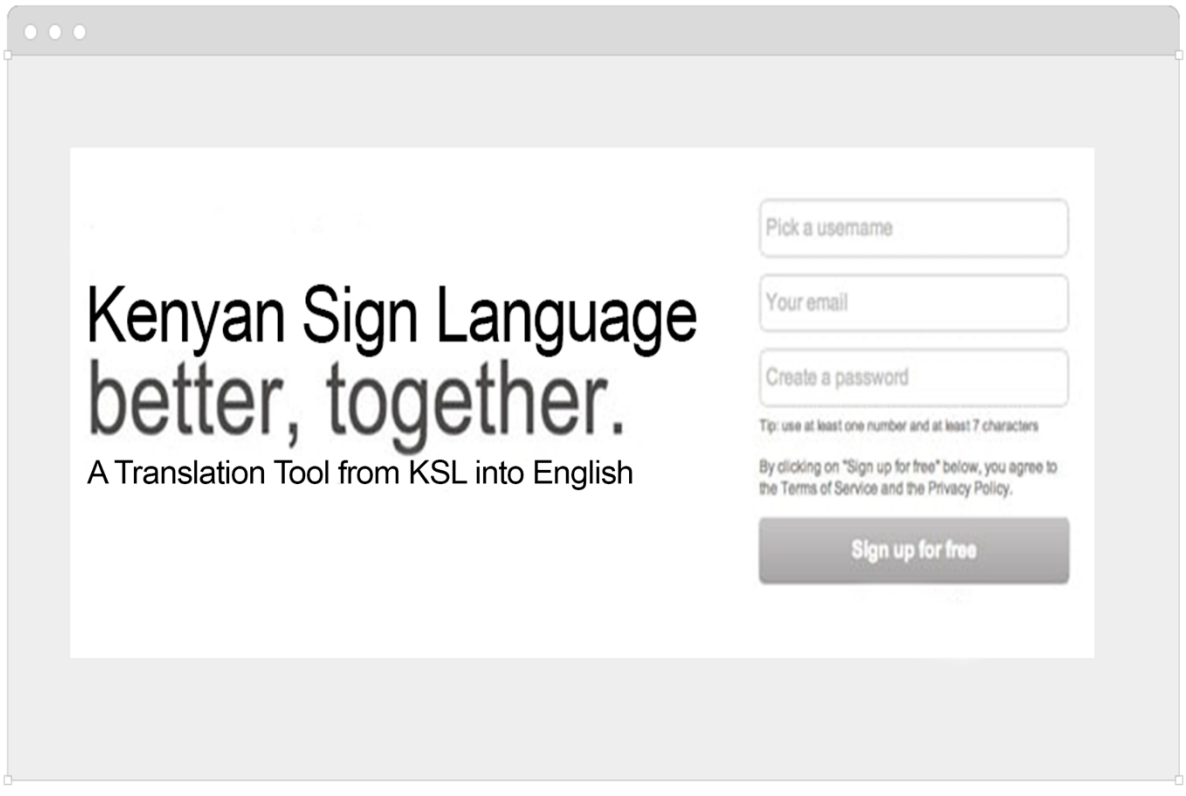


Figure 4.9 Registration Page

Users use the login page to access the application once they have done the initial registration. They each use the credentials they used to register to the application. A login page should

appear that has the email and the password field that allows users to enter their details. The forget and login command button allows users to reset their password and login to the application respectively. Figure 4.10 shows the wireframe for the login page.

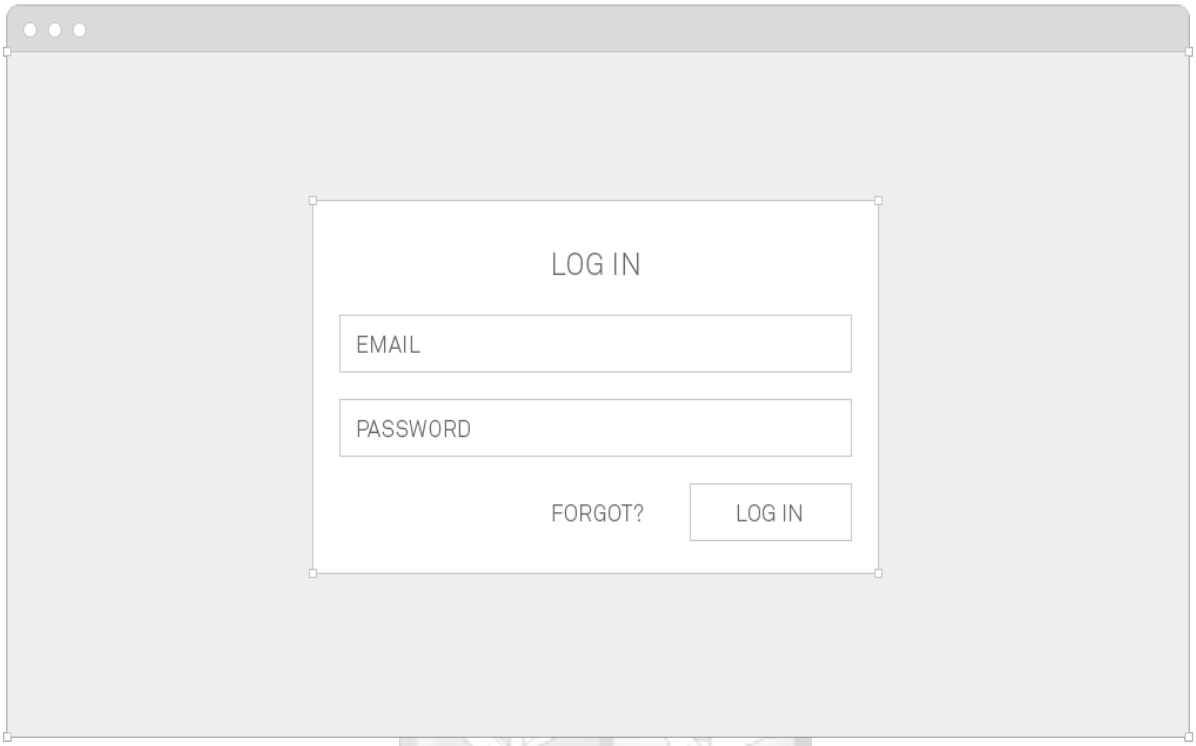


Figure 4.10 Login Page

The dialogue box in figure 4.8 shows the interface the users interact with while using the application. The sign language notation found in the screen is analyzed by the model to produce an output that is related to its intended meaning by the signer. It is crucial that the signer should have his/her notation be able to be covered within the view of the camera so as to have the entire notation captured without cutting our other critical hand gestures that may be needed for the full view of the notation being made for the model to translate.

The model should be able to translate that Kenyan sign language into readable text for the users using it. In this case the figure 4.9 below showcases the wireframe of the model window for the translation.

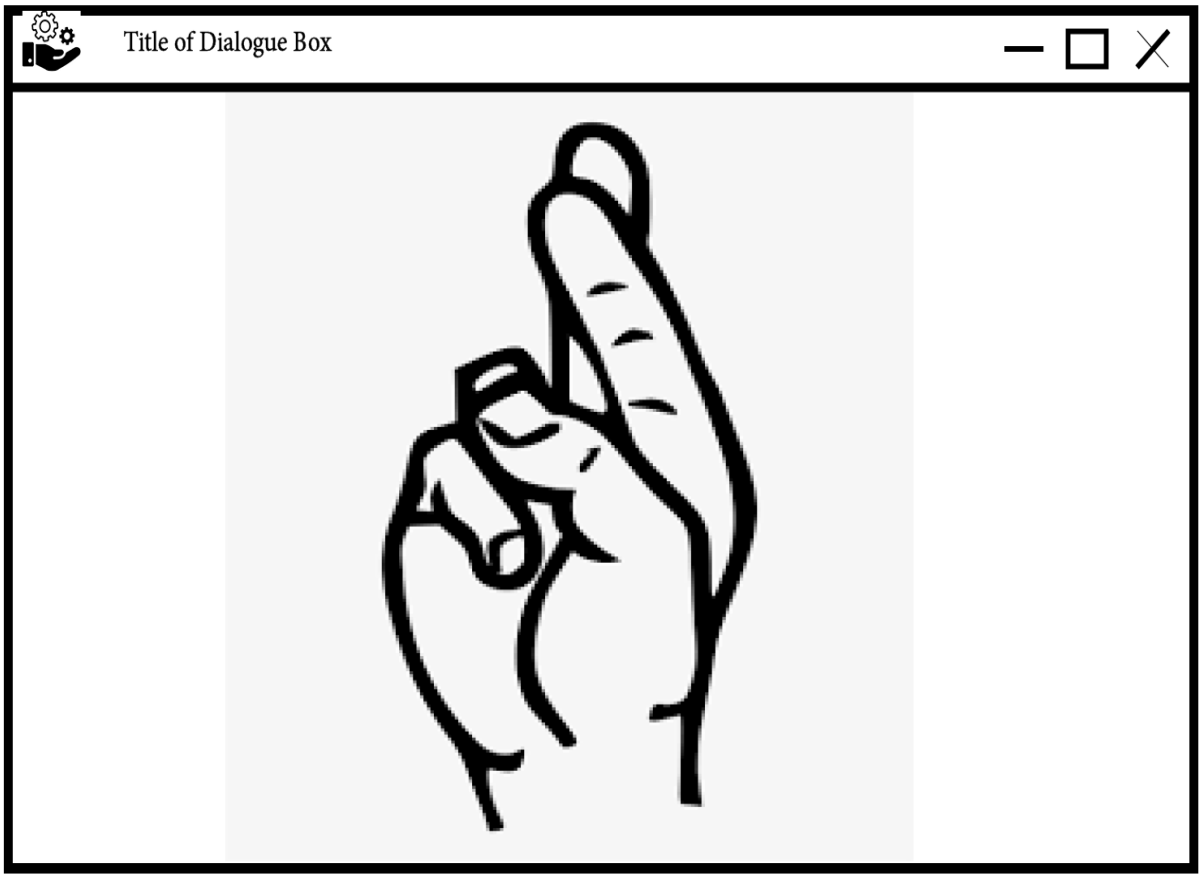


Figure 4.11 Model Dialogue Box

The interface showcases a dialogue window that contains the normal controls for a usual application. This involves the icon of the window, the title of the window and the windows controls that minimizes, expands and closes the application window of the model.

The main window space has the camera view of the signer doing the Kenyan Sign Language notation. What happens here is that the model tries to scan the entire camera view to detect for any notation. If it does find one, it encloses the signer's notation into a square box. The square box tracks the signer's notation while outputting the related meaning of the notation as text on the screen in real-time. As the signer continuously does the notations, the model continuously tracks the notation and outputs the corresponding textual meaning. Each textual output indicates the confidence level of the translation. If the model is able to match the notation to the textual translation, the confidence level is then indicated as a percentage of 90 or above. In this case the confidence level keeps varying depending on the variation of the sign language notation being performed. The aim of this research is to make sure that the model is carefully optimized and trained to correctly do the translations correctly and accurately to avoid overfitting and low confidence levels of the translation.

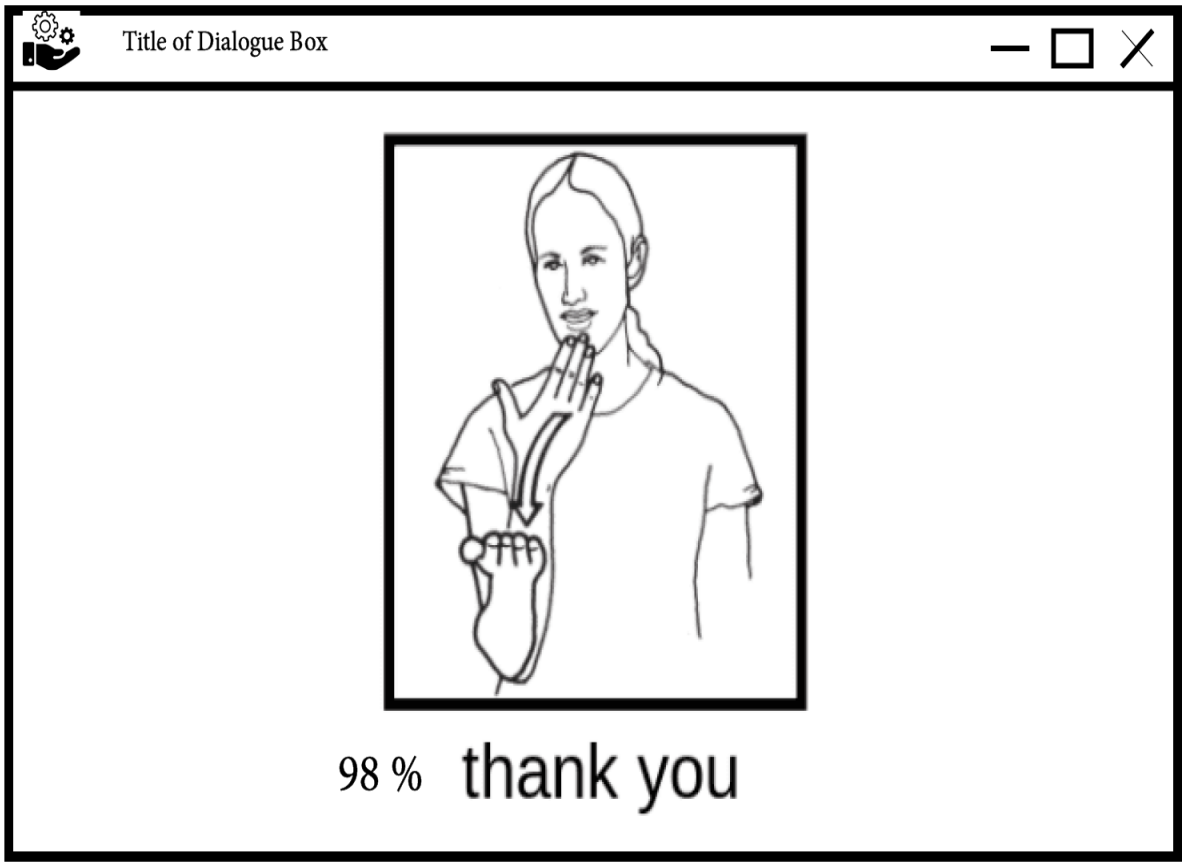


Figure 4.12 Live Kenyan Sign Language Translation

Figure 4.12 showcases the operation window of the model and how it is expected to look like to the users during its operation. Configurations in the code is made so that the frame size is big enough to cover enough real estate of the screen to allow visibility and interactivity visually with the user. A 1000 by 750 pixel is sufficient for the OpenCV video capture display.

Chapter 5 System Implementation and Testing

5.1 Introduction

The aim of this study is to develop a deep learning model that translates Kenyan Sign Language into English. This helps individuals to understand the various sign language notations done by the signers. This chapter dives deep in the implementation and testing of the model. The implementation focuses on the data preparation (pre-processing), the various sections and systems of the model, their implementation and functionality and the tools used in the implementation. Testing focuses on the verification of the model's ability to satisfy the functional and usability requirements.

5.2 System Implementation

5.2.1 Development Environment

The model development was carried out on the Jupyter notebook as the integrated development environment which allows for the development of the codebase for the model. The platform was preferred as it offers access to the important python frameworks needed for the model. The environment is integrated in the web browser that allows for quick implementation and development without the huddle for an entire code editor with all the bells and whistles that a normal editor has. The platform was also preferred due to the capability to use hardware acceleration of the GPU (Graphical processing unit) which is used in the training of the model and making sure that the tensor flow is able to quickly train the model as expected. The jupyter notebook platform also provided access to the powerful libraries such as Tensorflow, OpenCV and python code editor. It also provided the automated backup of the notebooks containing the code and the image datasets.

There are hardware devices that was used in the development of the model that had various characteristics that was comfortable for the model implementation. It consisted the graphics card, CPU, RAM memory and the hard drive. Table 5.1 shows the hardware resources that was used for the development of the model. The main rationale in the choice of the various details of the components was that the training of the model would be done faster to allow finetuning the model at each successive training. This would allow for an accurate model that was able to carefully translate the sign language notations to its respective textual meaning that users can appreciate.

Table 5.1 Hardware Details

Hardware	**Details**	
Mackbook Pro Laptop	GPU	4 GB Integrated Memory
	CPU	M1 Processor
	RAM	16 GB
	Disk	512 GB

The software resources included the programming language that was used the model and the libraries as well. Python was the language used. The libraries used consisted of those commonly used in machine learning and computer vision. These libraries provided the different functionalities needed for the model development. The jupyter notebook was used for the code writing and editing. The table 5.2 shows the software resources that were used in the development of the model.

Table 5.2 Software Details

Software	**Version**
Windows OS	11
Python	3.11.2
Git for Windows	2.40.0
Jupyter Notebook	6.5.3
Opencv-python	4.7.0.72
LabelImg	1.8.6
Tensorflow	2.12.0

The above software applications and libraries are installed in the laptop in preparation for the next steps of the implementation. The steps of the installation involved:

i. Having a windows 11 operating system ready for the package installations.

ii. Python programming language is installed from the python website repository. During the installation, the checkbox that allowed for the python application to be accessible as a global variable that allows in to be accessible at the command line was checked.

iii. Git for windows, 64 bit was downloaded from the git website and installed. By default, git, is accessible to the windows commandline.

iv. Jupyter notebook is the main code editor for the python coding tasks. It is installed usin the *pip install jupyter notebook* command in the windows commandline. It is launched by the *run jupyter notebook* command in the commandline.

v. OpenCV, the computer vison library is installed using the windows commandline. the command to install it is *pip install opencv-python*. This installs opencv and numpy which is a dependency for opencv installation.

vi. labelImg is an application that is used to label the images containing the notation. It is cloned from a repository in github. It is a popular image annotation tool created in python. (https://github.com/heartexlabs/labelImg). It can also be installed in the commandline using *pip install labelImg.* After installation, you can run it by simply typing *labelImg* in the commandline. A graphical user interface is then lauched.

vii. Pyqt5 is a python GUI framework that is used to run python graphical interface applications. It is a dependency needed by labelImg to run the application. Pyqt5 is intalled by running the following command in the commandline. *pip install pyqt5.* This command installs pyqt5.

5.2.2 Cloning of repositories

The first step is to first clone the TensorFlow object detection model from a GitHub repository. The GitHub application is first installed to the operation system and the version used for this study was 3.2.0. This allowed for the cloning of the repository to the drive D: partition of the hard drive. The D: drive acted as the working space for this study. The image below shows the cloning process and the GitHub link respectively.

git clone https://github.com/tensorflow/models.git

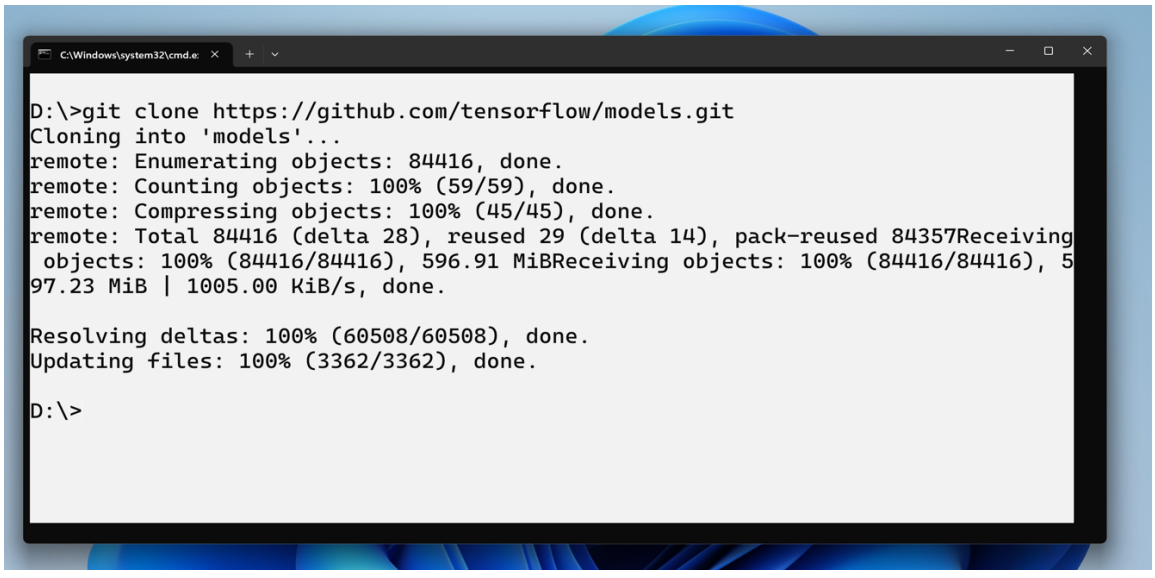


Figure 5.1 Cloning the TensorFlow git repository

5.2.3 Collecting the image dataset

The source of our image dataset is Kaggle. As mentioned earlier, In a web-based data-science environment, Kaggle enables users to explore, construct models, and find and publish data sets. It is a repository of published datasets that can be used by any users for studies and research. An account on Kaggle was created with an email address and a password. These credentials were used to log in and download the Kenyan Sign Language Image dataset (KSLC: Dataset). Figure 5.2 shows the dataset that was downloaded to the D: drive in preparation for the pre-processing steps ahead. The url is https://www.kaggle.com.

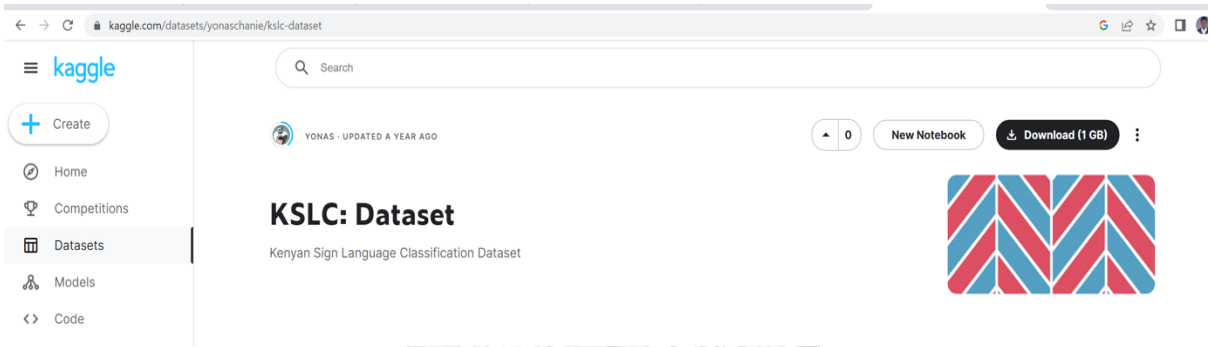


Figure 5.2 KSL Dataset Download

In order to supplement the dataset with the notations from the Kenya National Association of the Deaf, the images were captured from a camera while doing the sign language notations. These notations were replicated from what were in the booklet in order to capture the signer doing the notations and taking photos of them. Several images were taken of each notation in various lighting and variations as well to get a varied dataset that facilitates a better model to train. The following notations were collected as s proof of concept of this research study. They included church, enough, friend, hello, i_am_married, i_love_you, love, me, mosque, no, seat, temple, thank_you_so_much and yes respectively. The reason behind the choice is that these are the most popular phrases used among the deaf community in Kenya. They portray a common basic notation that are used in any setting like home, school or work as well as circumstances such as one to one interactions at events or gatherings.

To achieve the image collection through the webcam, the OpenCV library was used. A code was written to initiate the opencv library to capture timed images of the various variation of the signers notations. The images were stored in the respective folders of the notation meanings as shown in figure 5.3 below. The code for this image capture process in listed in the Appendix section of this study.

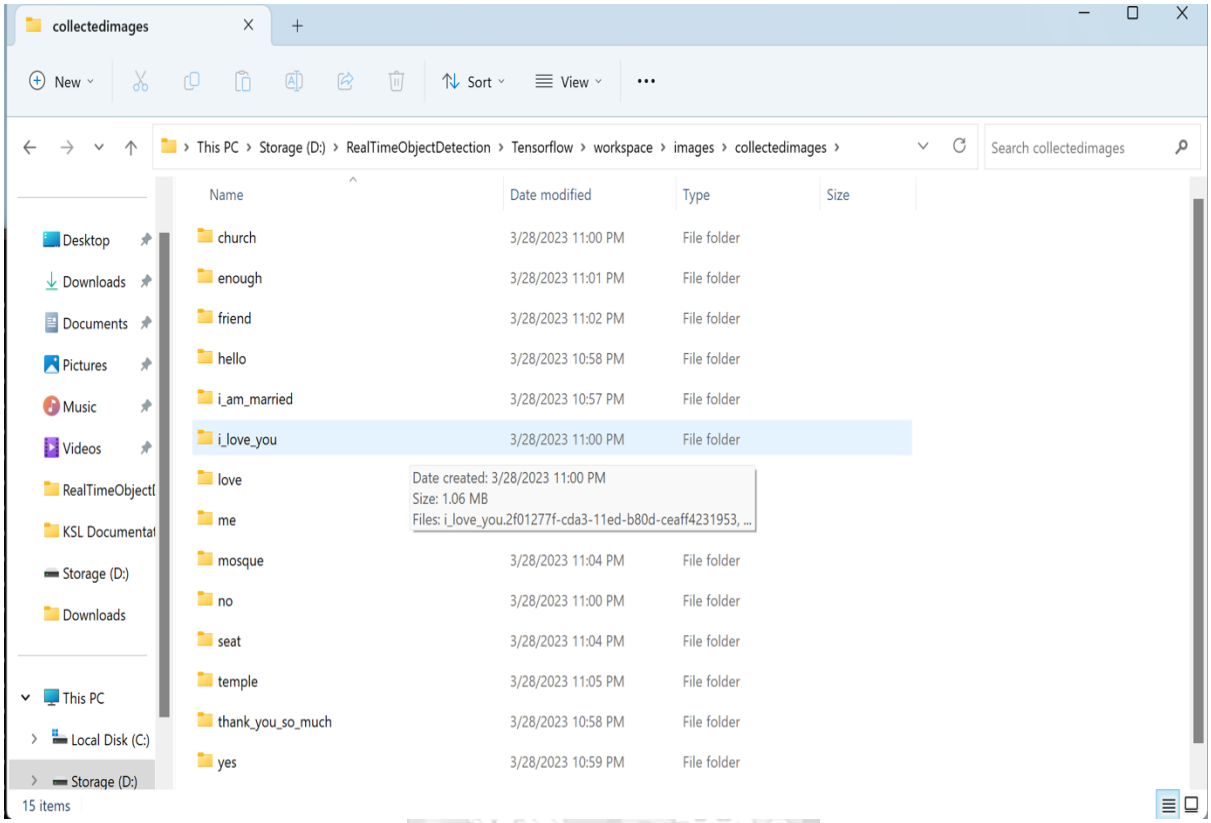


Figure 5.3 Image Collection using OpenCV

This capturing process continued to gather all the Kenyan sign language notations from the Kenya National Association of the deaf (KNAD), basic words and phrases in Kenyan Sign Language (2011) edition booklet. The figure 5.4 shows the mappings of the notations used in the model.

Each main category constitutes a group of KSL notations that ware gathered in the form of images both from the Kaggle repository and also the custom images collected from the KNAD booklet that contained numerous notations to supplement the Kaggle repository so as to enrich the collected data for the training and validation of the model. Variations to the notations were made to make sure that the model was able to train effectively and be ablet to translate the notations with minimal errors. This shall be seen in the validation and testing of the model in the end of the implementation. The images are all in their respective folders that are created to avoid confusion when it comes to the labelling phase which allows for easy distinguishing of each notation. The annotating application (labelImg) transverses into each folder and does the annotation in the respective named folders.

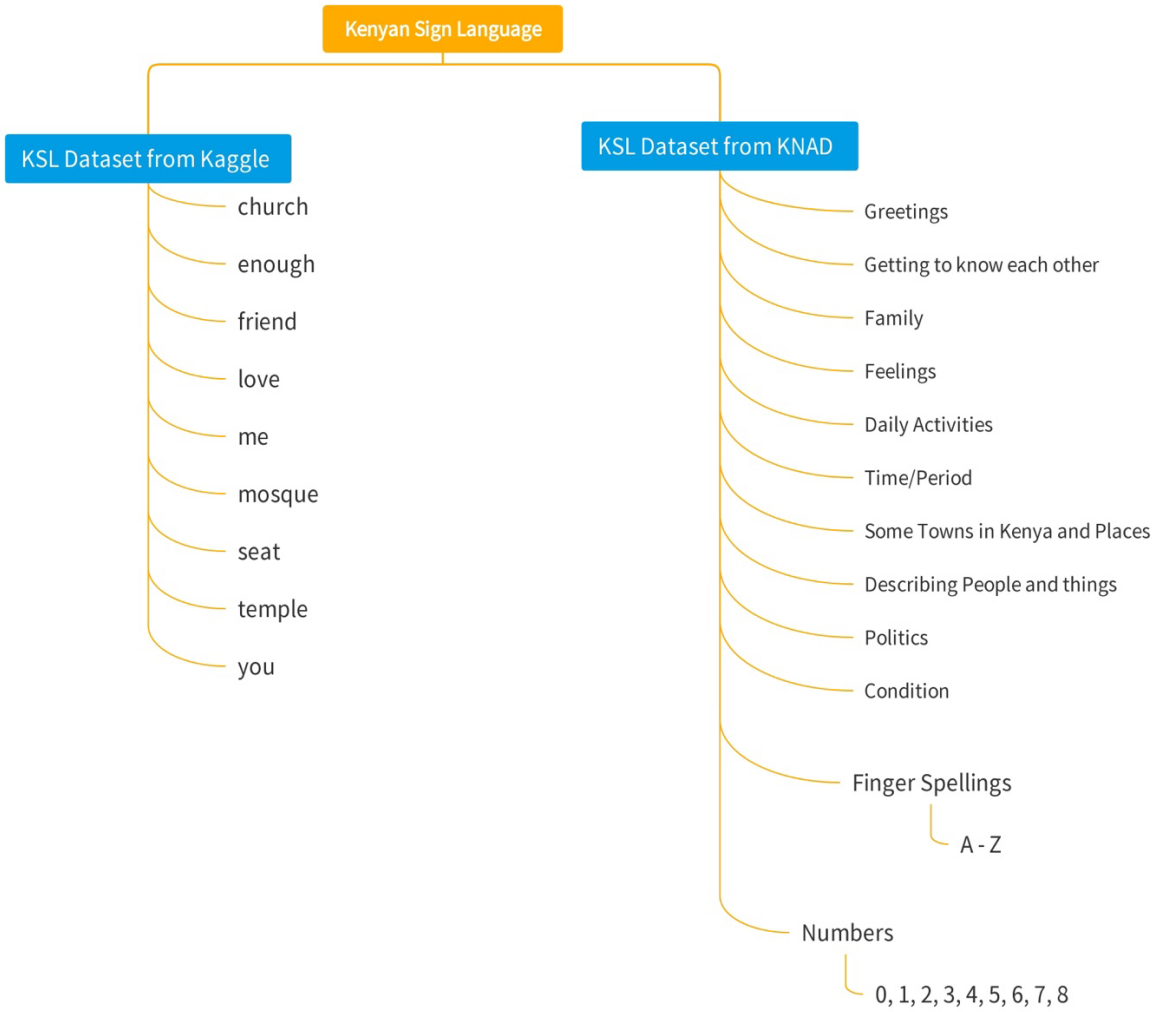


Figure 5.4 Collected data on Kenyan Sign Language Notation Images

5.2.4 Labeling the images using LabelImg package

With the help of this open-source image package, models for object detection can be used to label photos. The labelImg package enables users to choose images and annotate specific areas of those images. These annotations are made to any entities visible in the photographs that the user wants to annotate, including objects, people, animals, cars, and other living things. An illustration of the labelImg package during the annotation process is shown in figure 5.5.

Its graphical user interface (GUI) makes annotating photographs simple. Visual boxes are drawn around the objects in each image to provide this annotation. The captioned image xml files are output to a separate file. For best practice, save the xml files in the same folder as the images, e.g., the image and the xml file of the image after the annotation.

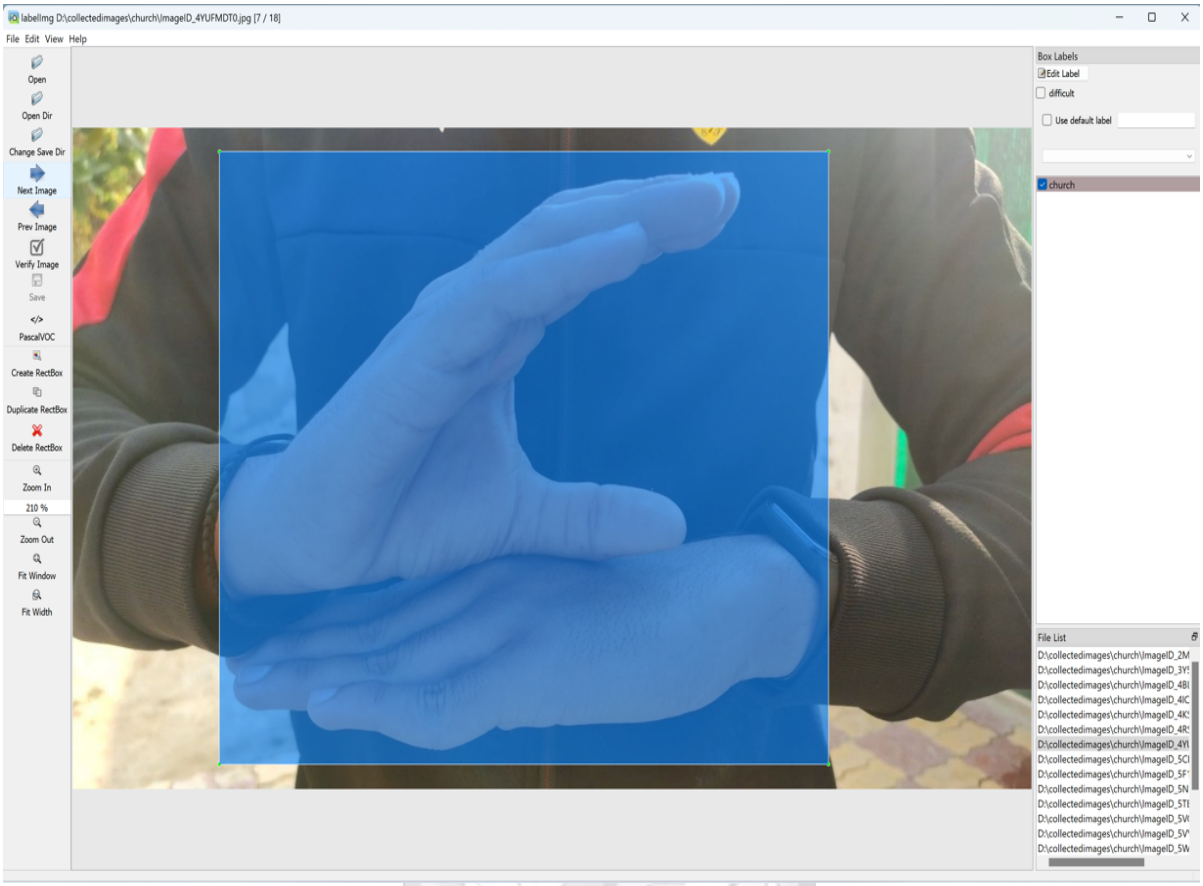


Figure 5.5 Annotating images using LabelImg package

5.2.5 Segmenting the images into training set and testing sets

The dataset was segmented into their respective training set and testing set. In this research 80% of the dataset was copied to the training folder of the tensorflow model folder and the other 10% was copied to the testing folder respectively. This was done in line with the data collection protocol in the research design of this study. All the image notations with their respective html files generated by the labelImg package are copied to their respective folders, that is the train and the test folder.

5.2.6 Model Development and Training

The model was implemented by the use of the TensorFlow 2 version of the TensorFlow machine learning platform. The TensorFlow was chosen because it has an object detection suite of models that was leveraged for this research for the notation detection. The model used was picked from a suit of models from the TensorFlow 2 model zoo from GitHub (*https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md).* The SSD MobileNet V2 FPNLite 320x320 variant was selected which had a balance of a shorter training time and perfect accuracy score compared to the rest. It leverages a transfer learning capability for the model development to speed up the learning process. This

model was selected among the rest because it works well with real-time object detection in video inputs. It has a higher detection speed. The chosen model had the following characteristics:

i. Speed in Milliseconds of 22.
ii. Mean Average Precision (mAP) of 20.2.
iii. And the detection output of boxes.

To achieve the learning experience, this model used supervised learning. Supervised learning is a form of machine learning technique in which a model is trained using labeled data. With this scenario, the KSL notation images were labeled to aid with model training. The algorithm utilized was a classifier, which is fully integrated in the TensorFlow and Keras models, making them ideal for this research. Mobilenet SSD was used as the model suite, which is an object detection model that generates bounding boxes from the input images. SSD is an abbreviation for single shot detector, a model that is utilized with Mobilenet as a backbone and is employed because it allows quick object detection designed for portable or mobile devices.

The model configuration also had inbuilt features such as data augmentation which involves using modified versions of the datasets made from existing data. It means that the training set is augmented artificially. It entails making little changes to the dataset or, using deep learning to add new data points. It does some of the preprocessing stages of the images such as feature extraction, normalization and post processing among others. It has an inbuilt image resizer that outputs the images to 320 by 320 pixels which is then used the other phases of the training of the model. The figure 5.6 shows the TensorFlow 2 model suite with their respective image resolution outputs listed in the TensorFlow GitHub repository.

SSD MobileNet v2 320x320	19	20.2	Boxes
SSD MobileNet V1 FPN 640x640	48	29.1	Boxes
SSD MobileNet V2 FPNLite 320x320	22	22.2	Boxes
SSD MobileNet V2 FPNLite 640x640	39	28.2	Boxes
SSD ResNet50 V1 FPN 640x640 (RetinaNet50)	46	34.3	Boxes
SSD ResNet50 V1 FPN 1024x1024 (RetinaNet50)	87	38.3	Boxes
SSD ResNet101 V1 FPN 640x640 (RetinaNet101)	57	35.6	Boxes
SSD ResNet101 V1 FPN 1024x1024 (RetinaNet101)	104	39.5	Boxes
SSD ResNet152 V1 FPN 640x640 (RetinaNet152)	80	35.4	Boxes
SSD ResNet152 V1 FPN 1024x1024 (RetinaNet152)	111	39.6	Boxes

Figure 5.6 Tensorflow 2 Model suite

Once all the required parameters were set, the training of the model commenced. The python script used in the training of the model is shown in figure 5.7.

```
153  # # 6. Train the model
154
155  # In[19]:
156
157
158  print("""python {}/research/object_detection/model_main_tf2.py --model_dir={}/{}
159  --pipeline_config_path={}/{}/pipeline.config
160  --num_train_steps=20000"""
161  .format(APIMODEL_PATH, MODEL_PATH,CUSTOM_MODEL_NAME,MODEL_PATH,CUSTOM_MODEL_NAME))
162
163
```

Figure 5.7 Model Training Script

The model was trained with 20000 steps to maximize on its accuracy. It was first tested with 500 steps and 10000 steps to see its detection accuracy. The 20000 training steps gave a perfect balance of detection accuracy and the duration of training which took 8 hours to complete.

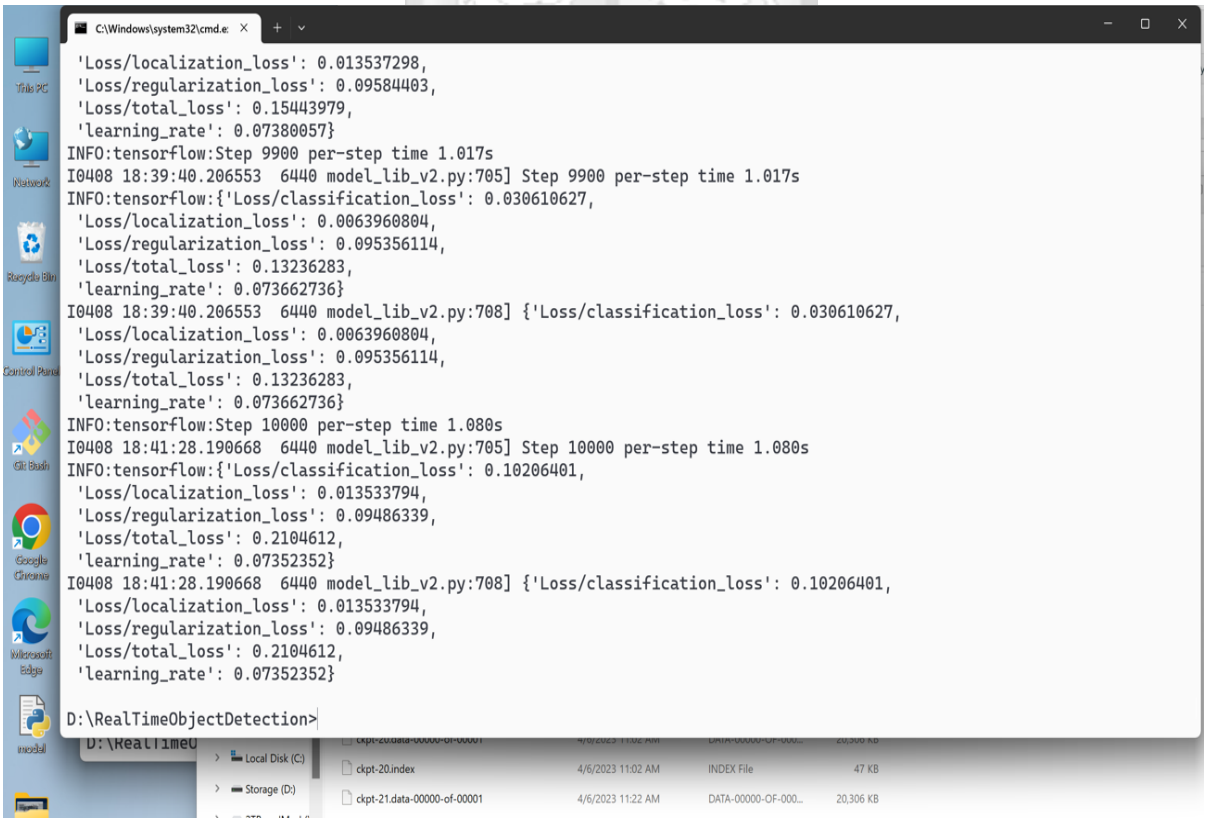


Figure 5.8 Snapshot of the model during training

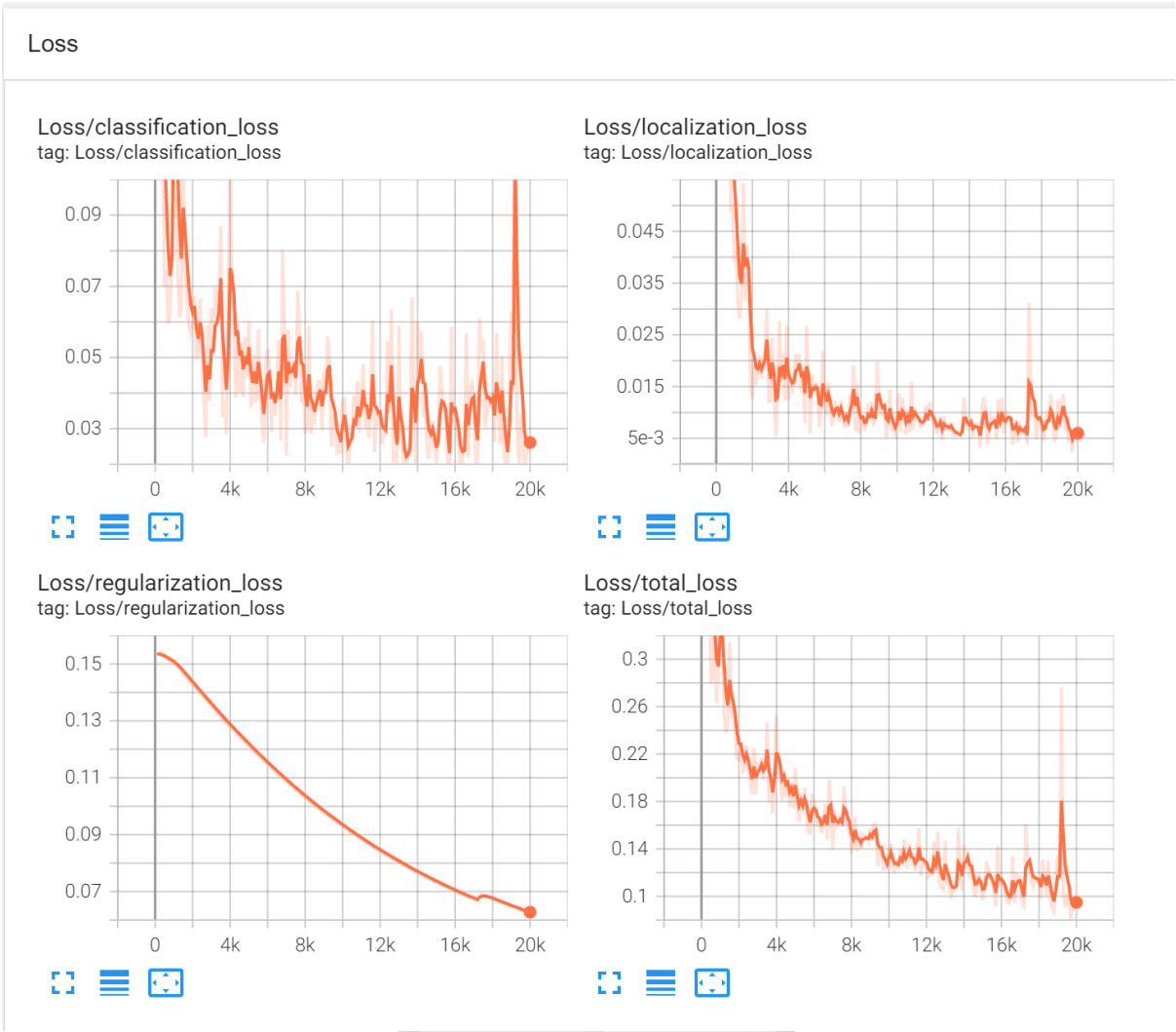


Figure 5.9 Training and Validation Loss

The graph shown in figure 5.3 shows the performance of the model during the training phase. It shows the training loses of the model. You can see that overall total loses were kept to a minimal. A bad guess results in a loss. In other words, loss represents how poorly a single case was predicted by the model. If the model's forecast is true, the loss is zero; otherwise, it is larger.

The model's training parameters are displayed on the x-axis. The model's training-related loss levels are displayed on the y-axis. The performance of a model in case prediction is typically evaluated using graphs. These graphs are used to fine-tune models to ensure that object detection models have the necessary levels of detection sensitivity and prediction accuracies.

5.2.7 Model Testing

The model underwent functional testing to confirm that it could satisfy the fundamental functional criteria. The model was also subjected to non-functional testing test cases including

reliability and scalability testing. Table 5.3 shows the test parameters against the results of the model. The model was tested against four (4) notations namely hello, thanks, yes and l love you.

Table 5.3 Testing Results of the Model

No#	Functional Requirements	Test Results
1.	Does the model accept an Image from the user in the form of videos and images?	Pass
2.	Does the model transform each of the images in the required format and extract the relevant features during pre-processing?	Pass
3.	Does the image identify the sign language notations of the people in the image?	Pass
4.	Does the system classify the notation from the image accordingly when properly fine-tuned?	Pass
5.	Does the system provide an output on text recommending the meaning of the sign language notation?	Pass
6.	The system should display the confidence level of the sign language notation and the image on the digital screen?	Pass

5.2.8 Model Deployment and Validation

The model deployment was done on a web application. This was achieved using a front-end react framework and flask backend. This is to allow the model to be accessible by the public and not just the developer alone. The essence of having the web-based user interface is to package the application in a manner that is usable by the users; in this case, anyone who wants to translate the Kenyan sign language notations into understandable English text to foster communication. The backend handles the user login logic and the engine for the model logic. The front-end handles the streaming of the video and the translation of the notations.

For this to be achieved, the model checkpoint was converted for the TensorFlow model format to the TensorFlow's format which is a JavaScript based version of the TensorFlow based format which allows the model to be interfaced to a web interface. The tensorflowjs format is used to allow models to be deployed in the web application and run in web browser. The web browser launches the webcam which starts streaming live input for the model to translate. Users

doing the Kenyan sign language notations have the web browser translate the notations into text.

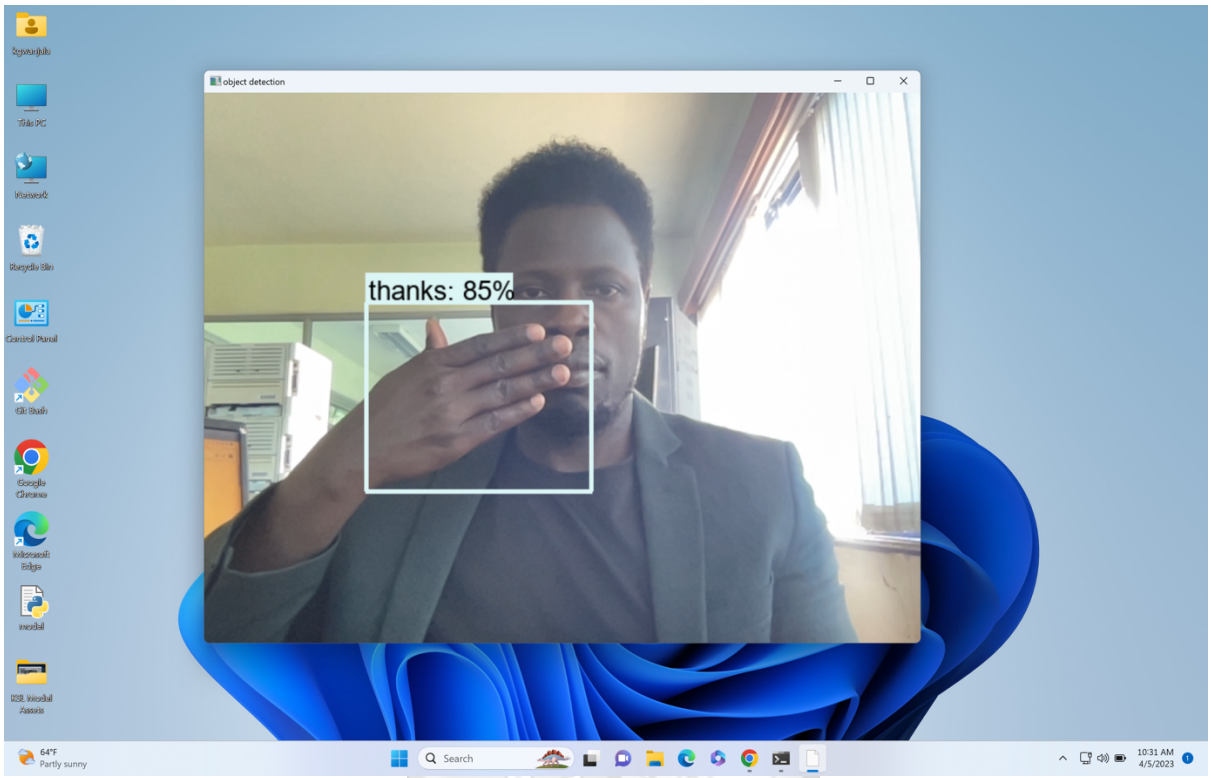


Figure 5.10 Model Operation

Figure 5.4 shows the model during operation. The box highlights the hand gesture which is in accordance with the labeling of the gestures during the pre-processing stage of the images. It predicts that hand gesture based on the traninng dataset and is able to output a textual meaning or the gesture above the box. This translation is accompanied by a confidence level in the form of a percentage. In this case the users using the model are able to understand the users doing the notation by reading through the text on the screen. This application can be used online in communicating with individuals during online meetings or collaboration activities. The hearing population are able to interact with the deaf online through the notation that they use. The online hosting of the application allows accessibility to more users.

5.2.9 Model Robustness

The resilience of a model ensures that it continues to produce accurate predictions even when confronted with difficult scenarios, in this case sign language notations on which it has not been trained. In other words, a model must generalize effectively to new unseen KSL notations. Robustness is an important aspect of model performance, maintenance, and security. Robustness helps in noisy hand notations that are not clearly done in the field of view. The

model is able to still detect hand gestures that are done further away from the field of view of the camera in as far as 4 meters. The model is able to detect hand gestures in lower lighting conditions.

5.2.10 Hyperparameter Tuning

An experimental study approach served as the foundation for the construction of the deep learning model. To track the model's performance, a number of parameters were changed. The training steps and the number of epochs were two of the parameters that were changed. The model development had to undergo several levels of finetuning to reach to the expected levels of detection accuracy for the textual inference of the notation presented in the real-time video input via the webcam. This was made possible by the ample time for testing and varying the parameters plus also the gpu capability of the MacBook being used as one requirements of this research endeavor. The parameters and the corresponding results are as show in table 5.4 below.

Table 5.4 Hyperparameter Tuning Values

Hyper Parameter	**Results**
Training Steps	
500	A shorter training period of 3.5 hours with a maximum average precision of 0.58 (58%)
10000	A moderate training period of 5 hours with a maximum average precision of 0.6 (60%)
20000	A longer training period of 8 hours and a maximum average precision of 0.85 (85%)
Number of Epochs	
20	Accuracy of 75.%
30	Accuracy of 85%
40	Accuracy of 85%

Chapter 6 Results and Discussions

6.1 Introduction

The main goal of the study was to develop a Kenyan sign language recognition model that translates notations into English text. The model considered a several classes of Kenyan sign language notations for this matter. This chapter discusses the model evaluation results and performance, as well as detailed discussions on them.

6.2 Model Evaluation

The model's performance was assessed using the following metrics: precision recall (maxim um average precision - mAP) and average recall. They can be obtained in the following ways: True positive is abbreviated as TP, and False positive is abbreviated as FP. Recall, on the other hand, indicates the percentage of true positives that are correctly classified. It assesses the model's ability to identify all relevant instances. Recall is calculated using equation a, where FN stands for false negative.

(a) $\text{Precision} = \frac{TP}{TP+FP}$

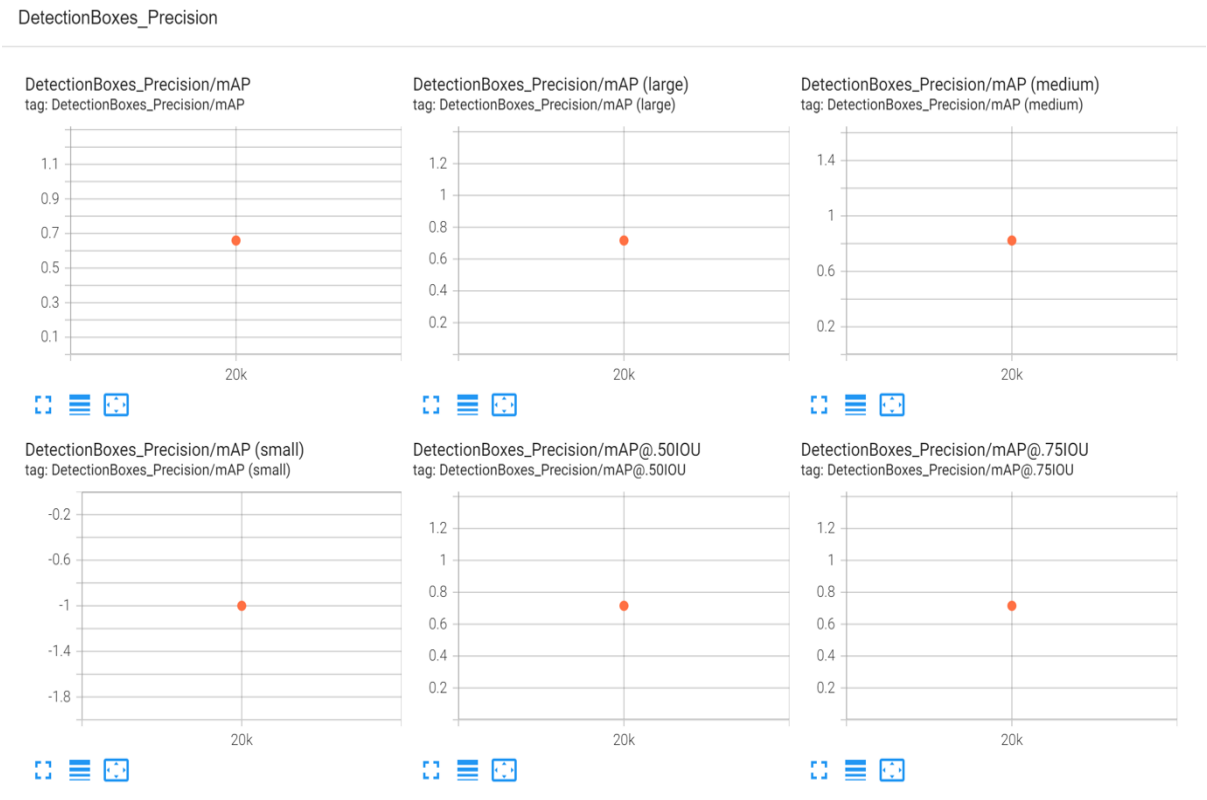


Figure 6.1 Snapshot of the Average Precision Results

In figure 6.1, the average precision is indicated to be 0.78 which translates to 78 %. These outputs are screenshots from the tensorboard module which is a TensorFlow extension that is used to provide performance metrics into a graphical output. The results were taken when the model was trained at 20000 steps which was taken as the optimal parameter with the best results.

Recall is given by equation b as:

$$\text{(b) Recall} = \frac{\text{TP}}{\text{TP+FN}}$$

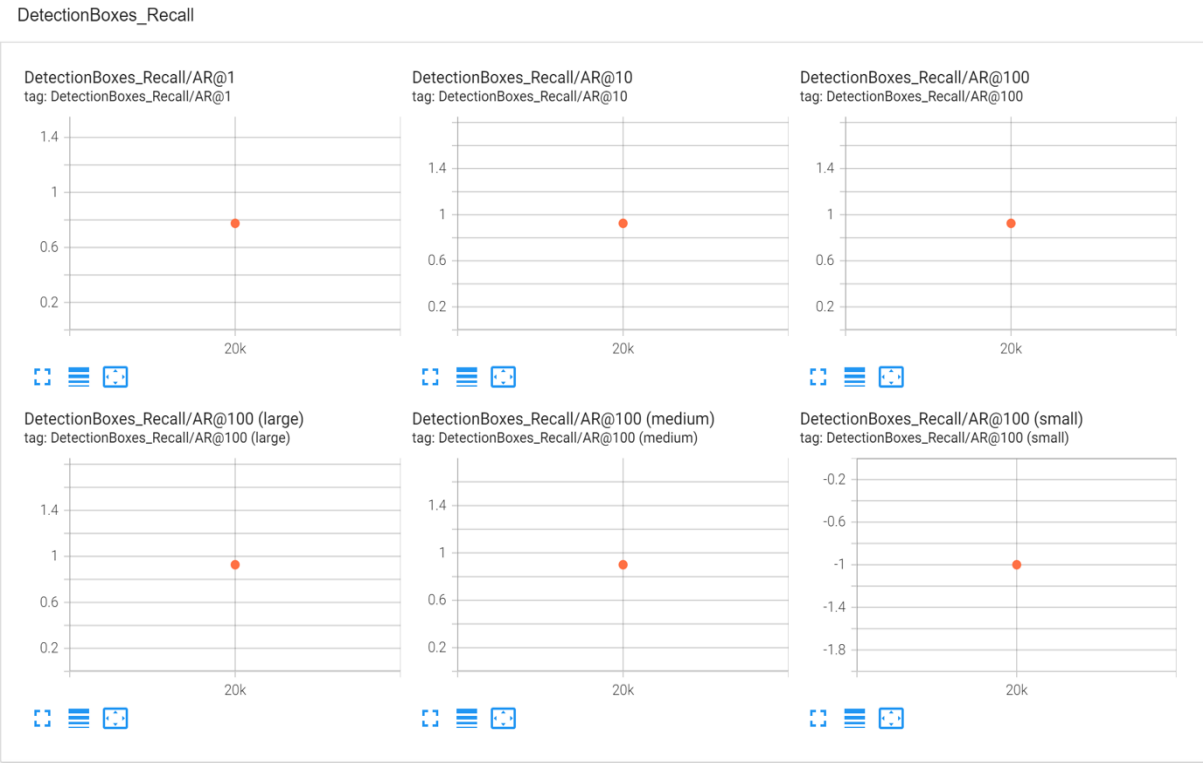


Figure 6.2 Snapshot of the Average Recall Results

The output shown in figure 6.2 shows the average recall values of the model. The best recall value the model was able to attain with the fine-tuned parameters was 0.85 which translates to 85%. This value meets the threshold for any object detection model that aims to detect objects within images and videos. This shows that the model is able to give us consistent results based on the input notation from the webcam or any live stream input.

So how does these results mean in relation to the problem statement. It simply means that for most of the time the model is able to detect and translate notations accurately. Occasional fails

may occur when there is a variation in the hand pose which in some cases, users may be doing a lazy version of the notation instead of doing it as it is expected. The model can occasionally extrapolate and detect the notation but fail in some instances. The user has a key part to play in realizing this benefit by doing the notation correctly rather than training the model in this lazy variation of the notations which hinder the effectiveness of this research in the first place.

There are several factors and variables that have the potential to raise the success threshold of any image classifier model for sign language translation. These approaches spans from the amount of dataset collected, the variation of the dataset in terms of image dynamics like lighting conditions, background information and gesture variation as well as the model training parameters and in some cases the algorithms used.

i. **Amount of dataset collected** – In supervised learning, labeled datasets are given to a model for training. The different types of notations one wants the model to be detected are referred as classes in machine learning terms. In this case a class can be a KSL notation like “hello”. This is considered a one class in machine learning. The more the classes the model aims to classifier or detect in the case of KSL notations, the more the datasets needed for each class in order to properly train the model in distinguishing between them. This means that more work is needed in data collection in collecting a lot of datasets for the model to train on.

ii. **Variation of the datasets** – It is also noted that variation of images is needed of a particular class to be classified or detected by the model. Take for example the “hello” KSL notation; you can have 60 images in outdoor setting, maybe another 60 images in an indoor setting, another 60 in natural light setting and perhaps another 60 in inhouse light condition or in a group setting. In this manner the model is able to clearly detect the notation since it has been able to be trained in the various surrounding conditions of the notations use cases and this increases the accuracy of the detection. Getting such quality images with different variations may be challenging to the researcher and this needs to be intentionally be put into consideration for optimal results.

iii. **Gesture variations** – KSL involved notations that were not just static postures but rather a movement of the hands into different motions to communication. The model was not able to clearly articulate the exact notation for detection. This also formed as one of the challenges in this research as mentioned in chapter 7, section 4 respectively.

iv. **Model training parameters** – They play a key part in the model development and determined the accuracy level of the model. Careful experimentation was done to get better results by finetuning the parameters but a lot of time is be needed in this task.

The phrases that the model struggled with were those that involved movements of the hand positions to communicate the notation. This involved notations like “help” which involved two hands forming a motion to communication the notation among others.

Chapter 7 Conclusions, Recommendations and Future Work

7.1 Conclusion

The ultimate objective of this research was to develop a Kenyan sign language translation model that could translates these notations into readable English text. This facilitates users both signers and non-signers who interact with the deaf individuals in order to aid in their interaction with them and foster communication. The notations that were covered spanned from finger spellings, numbers and basic day to day notations such as greetings, family, feelings, daily-activities, time-period as well as notations on interaction. The study was set on a sequential list of milestones that ware carefully done to meet the objectives of the study mentioned in section 1.3. of this study.

The initial objective was to investigate the challenges in Kenyan Sign Language translation in Kenya as a country. This objective was achieved by doing a thorough literature review and paper summarization on the issues that exist and impact the users and the KSL as a whole. The section 2.2 of this study highlighted that some of the difficulties and challenges faced was communication barriers between the deaf and normal hearing individuals. This deaf group to face hinderance in the labor market and also economic day to day transactions which are key to their survival and living just like any other normal hearing population. In some places stigmatization is also an issue where some individuals may fail to relate with them favorably. The children experience cognitive development delays because they lack a home language of communication with their parents and siblings when their parents do not learn how to sign or learn sign language to interact with their kids. Sign language classes are expensive both for the deaf and the hearing population to learn hence the need of technology. In the technological landscape different methods have been used of which some are expensive to implement and acquire. Special hardware is needed to have fully fledged solutions that cater for the needs of the deaf in communication with the hearing population.

The second objective was to review existing computer vision algorithms and models used in Sign Language recognition. The study reviewed both the hardware based and software-based algorithms that have been used in sign language translation. Several computer vision techniques models and algorithms were analyzed this formed a gap from the previous studies that formed the basis for this study. This was showcased in sections 2.3 and 2.6 respectively.

The review found that a model to translate Kenyan sign language is needed as well as a software-based solution is critical due to is affordability compared to hardware-based solutions such as glove based and the rest.

The third objective was to design a computer vision model that recognizes Kenyan Sign Language. The computer vision model was developed using transfer learning by the prototyping-based methodology and on a quantitative research design. The development environment consisted of Jupyter notebook for the IDE, python programming language, TensorFlow for the deep learning model and the OpenCV for the image processing. The model development parameters included were as follows: image size of 320 by 320 jpeg format, steps of 20000 and a learning rate of 0.054 average. Chapter 4 touched on the modelling section of the model and application. All the use cases, designs, sequence diagrams, entity relationship diagrams and class diagrams were developed and documented. The implementation of the model and its various parts are showcased in the fifth chapter.

The final objective was to validate the computer vision model. A graph was plotted using the matplotlib library which is a python library that allows for the plotting of graphs from input parameters of the model. This plot showed the performance of the model against detection of the sign language notation input from the webcam. The function takes a live input from the webcam and shows the notation detected and its related textual meaning. The model achieved an accuracy of 76%.

7.2 Research Contributions

The research made the following contributions:

The developed computer vision model provides a solution for translating Kenyan sign language to English text. This is relevant in fostering communication between the signers and the non-signers predominantly the hearing population. This is to enable day to day interaction in gatherings, events and meetings. Online sessions in the case of meetings, zoom meetings and classes can utilize this tool to be able to be understood by online users who have a need to understand whatever is being communicated. The deaf are be able to communication and be understood by his/her online audience because the model is be outputting textual translation from the Kenyan sign language notations they are doing on the other end of the online communication.

The model has been able to address the aspect of the format generation and the costing as was stated in the problem statement. The format being focused on this study is the English textual mapping which comes from the translation of the KSL notations to English text. The hearing population are able to understand the signers in this regard. The aspect of cost is addressed by implementing a KSL model that does not need gloves and sensors but purely software. The tools and applications that are used on opensource and thus minimal financial resources were used. Still on the aspect of cost, users are able to bypass the need to take KSL classes to learn how to sign but simply use this model to do the translation for them. Parents may not have the need to take KSL classes to understand their disabled children thus cutting on training fees and other costs extensively.

7.3 Recommendations

The research came up with some few interesting findings. The Kenyan sign language notations are many and keep evolving as time goes on. Newer notations and meanings are being created to add onto the existing basic day to day notations in order to enrich the KSL. As the society, communities and culture changes, these also create new notations that are used by the KSL community to better understand each other thus for those creating such solutions, a regular update of the models should be critical to maintain relevance.

7.4 Limitations of the research

i. The models main focus was on the hand and finger notations. It did not consider other gestures of communication. It is known that sign language also takes into account of body positions, hand movements as well as facial expressions. This takes away the enriching gestures for effective Kenyan sign language translation by simply detection only the hand gestures.

ii. The model does not provide solutions to managing other issues that exist that affect the deaf community which are communication related. Matters to do with partial representation of a Kenyan sign language notation instead of doing it correctly as intended is not resolved or taken into account. Some notations need both hands to correctly represent the meaning. Doing it with one hand alone may not showcase the intended meaning and this model may not correct that.

iii. Some notations that are closely related in the essence of meaning may cause some issues in detection. This is due to almost or nearly the same hand pose is done to represent the

different meanings. This meant that the model may need a lot of image dataset, training steps and variation of the images for the model to correctly distinguish them.

iv. The model does not recognize other sign language notations that it has not been trained on. Someone doing and Indian or Korean sign language notation may not be detected. Perhaps a universal solution to this can be achieved in future for all sign language notations detection.

v. The model detection depends on the position of the hand gestures in the image or camera view. The model performs well when the hand gesture is well exposed and are within the view of the camera for maximum efficiency in translation.

7.5 Future Work

The basis for the future works that is being outlined here is pegged on the shortcomings of this research. The following are suggested for future work:

i. Training of the model with a huge dataset containing more and varied notations to enhance its overlapping features of the notations. This increases its performance of recognition of the notations.

ii. The model should be trained to also capture other bodily gestures that go along with the hand gestures during signing by the signers. This enables the capturing of meaningful gestures that are not captured in hand notations alone.

iii. More data should be captured especially on cases where notations that closely look similar in order to make the model distinguish it from the others more accurately. This increases the model's accuracy.

References

Aggarwal, C. C. (2018). *Neural Networks and Deep Learning*.

Amin, M. S., Rizvi, S. T. H., & Hossain, M. M. (2022). A Comparative Review on Applications of Different Sensors for Sign Language Recognition. In *Journal of Imaging* (Vol. 8, Issue 4). MDPI. https://doi.org/10.3390/jimaging8040098

Anam, R., Rahman, M., Haque, M. O., & Islam, Md. S. (2015). Computer Vision based Automation System for Detecting Objects. *International Journal of Intelligent Systems and Applications*, *7*(12), 68–74. https://doi.org/10.5815/ijisa.2015.12.07

Bantupalli, K., & Xie, Y. (2019). American Sign Language Recognition using Deep Learning and Computer Vision. *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018*, 4896–4899. https://doi.org/10.1109/BigData.2018.8622141

Bulugu, I. (2021). Sign language recognition using Kinect sensor based on color stream and skeleton points. *Tanzania Journal of Science*, *47*(2), 769–778. https://doi.org/10.4314/tjs.v47i2.32

Cabañas De Paz, R., Flores, M. J., & Martínez-Gómez, J. (2011). *Dynamic Bayesian Networks for Gesture Recognition* (Issue 1). http://www.eyetoy.com/

Charniak, & Eugene. (2018). *Introduction to Deep Learning*.

Crowe, K., Marschark, M., Dammeyer, J., & Lehane, C. (2017). Achievement, language, and technology use among college-bound deaf learners. *Journal of Deaf Studies and Deaf Education*, *22*(4), 393–401. https://doi.org/10.1093/deafed/enx029

Davies, E. R. (2018). *Computer Vision* (Fifth Edition). Academic Press.

Development Initiatives. (2020). *Status of disability in Kenya*.

Dhake, D., P. Kamble, M., S. Kumbhar, S., & M. Patil, S. (2020). Sign Language Communication with Dumb and Deaf People. *International Journal of Engineering Applied Sciences and Technology*, *5*(4), 254–258. https://doi.org/10.33564/ijeast.2020.v05i04.038

Hammoudeh, M. A. A., Alsaykhan, M., Alsalameh, R., & Althwaibi, N. (2022). Computer Vision: A Review of Detecting Objects in Videos – Challenges and Techniques. *International Journal of Online and Biomedical Engineering*, *18*(1), 15–27. https://doi.org/10.3991/ijoe.v18i01.27577

Hassaballah, M., & Awad, A. I. (2020). *Deep Learning in Computer Vision*.

Holmer, E., Heimann, M., & Rudner, M. (2017). Computerized sign language-based literacy training for deaf and hard-of-hearing children. *Journal of Deaf Studies and Deaf Education*, *22*(4), 404–421. https://doi.org/10.1093/deafed/enx023

Kenya National Commission on Human Rights (KNCHR). (2014). *From Norm to Practice, A Status Report on the Implementation of the Rights of Persons with Disabilities in Kenya*.

Keras. (2022). https://keras.io/

Khan, S., Rahmani, H., Shah, S. A. A., & Bennamoun, M. (2018). A Guide to Convolutional Neural Networks for Computer Vision. *Synthesis Lectures on Computer Vision*, *8*(1), 1–207. https://doi.org/10.2200/s00822ed1v01y201712cov015

Kusters, A., & Hou, L. (2020). Linguistic Ethnography and Sign Language Studies. In *Sign Language Studies* (Vol. 20, Issue 4, pp. 561–571). Gallaudet University Press. https://doi.org/10.1353/SLS.2020.0018

Lucas, C., & Bayley, R. (2011). Variation in sign languages: Recent research on ASL and beyond. *Linguistics and Language Compass*, *5*(9), 677–690. https://doi.org/10.1111/j.1749-818X.2011.00304.x

Luchivya, R. O., Omolo, T. M., & Onditi, S. A. (2020). *Challenges parents face in learning Kenyan Sign Language*. www.iprjb.org

Mor, B., Garhwal, S., & Kumar, A. (2021). A Systematic Review of Hidden Markov Models and Their Applications. *Archives of Computational Methods in Engineering*, *28*(3), 1429–1448. https://doi.org/10.1007/s11831-020-09422-4

Mweri, J. G. (2018). Kenya Sign Language (KSL) Phonology: Articulatory Properties and Phonological Processes. *Linguistics and Literature Studies*, *6*(4), 169–182. https://doi.org/10.13189/lls.2018.060403

Neapolitan, R. E., & Jiang, X. (2018). *Artificial Intelligence : With an Introduction to Machine Learning*.

Neethu, P. S., Suguna, R., & Sathish, D. (2020). An efficient method for human hand gesture detection and recognition using deep learning convolutional neural networks. *Soft Computing*, *24*(20), 15239–15248. https://doi.org/10.1007/s00500-020-04860-5

Papastratis, I., Chatzikonstantinou, C., Konstantinidis, D., Dimitropoulos, K., & Daras, P. (2021). Artificial intelligence technologies for sign language. *Sensors*, *21*(17). https://doi.org/10.3390/s21175843

Parida, M., & Sinha, M. (2021). Pandemic and disability: Challenges faced and role of technology. In *Technology and Disability* (Vol. 33, Issue 4, pp. 245–252). IOS Press BV. https://doi.org/10.3233/TAD-200311

Patel, K. (2022). An Assistance System for Deaf, Dumb, Blind, and Learning Disability Individuals. *INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT*, *06*(05). https://doi.org/10.55041/ijsrem13314

Quinto-Pozos, D., & Cooley, F. (2020). A developmental disorder of signed language production in a native deaf signer of ASL. *Languages*, *5*(4), 1–18. https://doi.org/10.3390/languages5040040

Rguibi, Z., Hajami, A., Zitouni, D., Elqaraoui, A., & Bedraoui, A. (2022). CXAI: Explaining Convolutional Neural Networks for Medical Imaging Diagnostic. *Electronics (Switzerland)*, *11*(11). https://doi.org/10.3390/electronics11111775

Sultan, A., Makram, W., Kayed, M., & Ali, A. A. (2022). Sign Language Identification and Recognition: A comparative study. *Open Computer Science*, *12*(1), 191–210. https://doi.org/10.1515/comp-2022-0240

TensorFlow. (2022). https://www.tensorflow.org

Wolfe, R. (2021). Special issue: Sign language translation and avatar technology. In *Machine Translation* (Vol. 35, Issue 3, pp. 301–304). Springer Science and Business Media B.V. https://doi.org/10.1007/s10590-021-09270-4

Yangqing, J. (2014). *Caffe: Convolutional Architecture for Fast Feature Embedding*. https://caffe.berkeleyvision.org

Zhao, R., Wang, Y., Jia, P., Li, C., Ma, Y., & Zhang, Z. (2021). Review of Human Gesture Recognition Based on Computer Vision Technology. *IEEE Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, 1599–1603. https://doi.org/10.1109/IAEAC50856.2021.9390889

Appendices

Appendix A: Code

1. Code for webcam image collection and pre-processing.

```
#!/usr/bin/env python
# coding: utf-8

# Resources Used
# - wget.download('https://tensorflow-object-detection-api-tutorial.readthedocs.io/en/latest/_downloads/da4babe668a8afb093cc7776d7e630f3/generate_tfrecord.py')
# - Setup https://tensorflow-object-detection-api-tutorial.readthedocs.io/en/latest/install.html

# # 0. Setup Paths

# In[1]:


WORKSPACE_PATH = 'Tensorflow/workspace'
SCRIPTS_PATH = 'Tensorflow/scripts'
APIMODEL_PATH = 'Tensorflow/models'
ANNOTATION_PATH = WORKSPACE_PATH+'/annotations'
IMAGE_PATH = WORKSPACE_PATH+'/images'
MODEL_PATH = WORKSPACE_PATH+'/models'
PRETRAINED_MODEL_PATH = WORKSPACE_PATH+'/pre-trained-models'
CONFIG_PATH = MODEL_PATH+'/my_ssd_mobnet/pipeline.config'
CHECKPOINT_PATH = MODEL_PATH+'/my_ssd_mobnet/'


# # 1. Create Label Map

# In[2]:


labels = [
    {'name':'zero', 'id':1},{'name':'one', 'id':2},{'name':'two', 'id':3},
    {'name':'three', 'id':4},{'name':'four', 'id':5},{'name':'five', 'id':6},
    {'name':'six', 'id':7},{'name':'seven', 'id':8},{'name':'eight', 'id':9},
    {'name':'A', 'id':10},{'name':'B', 'id':11},{'name':'C', 'id':12},
    {'name':'D', 'id':13},{'name':'E', 'id':14},{'name':'F', 'id':15},
    {'name':'G', 'id':16},{'name':'H', 'id':17},{'name':'I', 'id':18},
    {'name':'J', 'id':19},{'name':'K', 'id':20}, {'name':'L', 'id':21},
    {'name':'M', 'id':22},{'name':'N', 'id':23},{'name':'O', 'id':24},
    {'name':'P', 'id':25},{'name':'Q', 'id':26},{'name':'R', 'id':27},
    {'name':'S', 'id':28},{'name':'T', 'id':29},{'name':'U', 'id':30},
    {'name':'V', 'id':31},{'name':'W', 'id':32},{'name':'X', 'id':33},
    {'name':'Y', 'id':34},{'name':'Z', 'id':35},{'name':'church', 'id':36},
    {'name':'enough', 'id':37},{'name':'friend', 'id':38},{'name':'love', 'id':39},
    {'name':'me', 'id':40},{'name':'mosque', 'id':41},{'name':'seat', 'id':42},
    {'name':'temple', 'id':43},{'name':'you', 'id':44},{'name':'fine', 'id':45},
    {'name':'good', 'id':46},{'name':'goodbye', 'id':47},{'name':'happy', 'id':48},
    {'name':'hello', 'id':49},{'name':'home', 'id':50},{'name':'meet', 'id':51},
    {'name':'thanks', 'id':52},{'name':'litte', 'id':53},{'name':'age', 'id':54},
    {'name':'born', 'id':55},{'name':'how', 'id':56},{'name':'how are you', 'id':57},
    {'name':'name', 'id':58},{'name':'when', 'id':59},{'name':'where', 'id':60},
    {'name':'who', 'id':61},{'name':'adult', 'id':62},{'name':'baby', 'id':63},
    {'name':'daughter', 'id':64},{'name':'married', 'id':65},{'name':'son', 'id':66},
    {'name':'help', 'id':67},{'name':'tired', 'id':68},{'name':'sick', 'id':69},
    {'name':'pray', 'id':70},{'name':'sleep', 'id':71},{'name':'study', 'id':72},
    {'name':'afternoon', 'id':73},{'name':'evening', 'id':74},{'name':'morning', 'id':75},
    {'name':'night', 'id':76},

]


# In[3]:


with open(ANNOTATION_PATH + '\label_map.pbtxt', 'w') as f:
```

```
64      for label in labels:
65          f.write('item { \n')
66          f.write('\tname:\'{}\'\n'.format(label['name']))
67          f.write('\tid:{}\n'.format(label['id']))
68          f.write('}\n')
69
70
71  # # 2. Create TF records
72
73  # In[4]:
74
75
76  import os
77
78
79  # In[5]:
80
81
82  get_ipython().system("python {SCRIPTS_PATH + '/generate_tfrecord.py'} -x {IMAGE_PATH +
    '/train'} -l {ANNOTATION_PATH + '/label_map.pbtxt'} -o {ANNOTATION_PATH +
    '/train.record'}")
83  get_ipython().system("python {SCRIPTS_PATH + '/generate_tfrecord.py'} -x {IMAGE_PATH +
    '/test'} -l {ANNOTATION_PATH + '/label_map.pbtxt'} -o {ANNOTATION_PATH + '/test.record'}"
    )
84
85
86  # # 3. Download TF Models Pretrained Models from Tensorflow Model Zoo
87
88  # In[38]:
89
90
91  get_ipython().system('cd Tensorflow && git clone https://github.com/tensorflow/models')
92
93
94  # In[6]:
95
```

2. Model training code

```
#wget.download('http://download.tensorflow.org/models/object_detection/tf2/20200711/ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.tar.gz')
#!mv ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.tar.gz {PRETRAINED_MODEL_PATH}
#!cd {PRETRAINED_MODEL_PATH} && tar -zxvf ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.tar.gz


# # 4. Copy Model Config to Training Folder

# In[6]:


CUSTOM_MODEL_NAME = 'my_ssd_mobnet'


# In[7]:


get_ipython().system("mkdir {'Tensorflow\\workspace\\models\\\\'+CUSTOM_MODEL_NAME}")
get_ipython().system("cp {PRETRAINED_MODEL_PATH+'/ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8/pipeline.config'} {MODEL_PATH+'/'+CUSTOM_MODEL_NAME}")


# # 5. Update Config For Transfer Learning

# In[8]:


import tensorflow as tf
from object_detection.utils import config_util
from object_detection.protos import pipeline_pb2
from google.protobuf import text_format


# In[9]:


CONFIG_PATH = MODEL_PATH+'/'+CUSTOM_MODEL_NAME+'/pipeline.config'


# In[10]:


config = config_util.get_configs_from_pipeline_file(CONFIG_PATH)


# In[11]:


config


# In[12]:


pipeline_config = pipeline_pb2.TrainEvalPipelineConfig()
with tf.io.gfile.GFile(CONFIG_PATH, "r") as f:
    proto_str = f.read()
    text_format.Merge(proto_str, pipeline_config)

```

```
58
59  # In[13]:
60
61
62  pipeline_config.model.ssd.num_classes = 76
63  pipeline_config.train_config.batch_size = 4
64  pipeline_config.train_config.fine_tune_checkpoint = PRETRAINED_MODEL_PATH+
    '/ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8/checkpoint/ckpt-0'
65  pipeline_config.train_config.fine_tune_checkpoint_type = "detection"
66  pipeline_config.train_input_reader.label_map_path= ANNOTATION_PATH + '/label_map.pbtxt'
67  pipeline_config.train_input_reader.tf_record_input_reader.input_path[:] = [
    ANNOTATION_PATH + '/train.record']
68  pipeline_config.eval_input_reader[0].label_map_path = ANNOTATION_PATH +
    '/label_map.pbtxt'
69  pipeline_config.eval_input_reader[0].tf_record_input_reader.input_path[:] = [
    ANNOTATION_PATH + '/test.record']
70
71
72  # In[14]:
73
74
75  config_text = text_format.MessageToString(pipeline_config)


76  with tf.io.gfile.GFile(CONFIG_PATH, "wb") as f:


77      f.write(config_text)
78
79
80  # # 6. Train the model
81
82  # In[15]:
83
84
85  print("""python {}/research/object_detection/model_main_tf2.py --model_dir={}/{}
    --pipeline_config_path={}/{}/pipeline.config --num_train_steps=20000""".format(
    APIMODEL_PATH, MODEL_PATH,CUSTOM_MODEL_NAME,MODEL_PATH,CUSTOM_MODEL_NAME))
86
```

3. KSL detection code

```
# # 7. Load Train Model From Checkpoint

# In[4]:


import os
from object_detection.utils import label_map_util
from object_detection.utils import visualization_utils as viz_utils
from object_detection.builders import model_builder


# In[5]:


# Load pipeline config and build a detection model
configs = config_util.get_configs_from_pipeline_file(CONFIG_PATH)
detection_model = model_builder.build(model_config=configs['model'], is_training=False)

# Restore checkpoint
ckpt = tf.compat.v2.train.Checkpoint(model=detection_model)
ckpt.restore(os.path.join(CHECKPOINT_PATH, 'ckpt-21')).expect_partial()

@tf.function
def detect_fn(image):
    image, shapes = detection_model.preprocess(image)
    prediction_dict = detection_model.predict(image, shapes)
    detections = detection_model.postprocess(prediction_dict, shapes)
    return detections


# # 8. Detect in Real-Time

# In[6]:


import cv2
import numpy as np


# In[7]:


category_index = label_map_util.create_category_index_from_labelmap(ANNOTATION_PATH+
'/label_map.pbtxt')


# In[ ]:


cap = cv2.VideoCapture(0)
while True:
    ret, frame = cap.read()
    image_np = np.array(frame)

    input_tensor = tf.convert_to_tensor(np.expand_dims(image_np, 0), dtype=tf.float32)
    detections = detect_fn(input_tensor)

    num_detections = int(detections.pop('num_detections'))
    detections = {key: value[0, :num_detections].numpy()
                  for key, value in detections.items()}
    detections['num_detections'] = num_detections

    # detection_classes should be ints.
    detections['detection_classes'] = detections['detection_classes'].astype(np.int64)

    label_id_offset = 1
    image_np_with_detections = image_np.copy()
```

```
67
68      viz_utils.visualize_boxes_and_labels_on_image_array(
69                  image_np_with_detections,
70                  detections['detection_boxes'],
71                  detections['detection_classes']+label_id_offset,
72                  detections['detection_scores'],
73                  category_index,
74                  use_normalized_coordinates=True,
75                  max_boxes_to_draw=1,
76                  min_score_thresh=.95,
77                  agnostic_mode=False)
78
79      cv2.imshow('object detection',  cv2.resize(image_np_with_detections, (800, 600)))
80
81      if cv2.waitKey(1) & 0xFF == ord('q'):
82          cap.release()
83          break
84  cap.release()
85
```

4. Web interface Login Code

```
import React from "react";
import loginImg from "../../login.svg";

export class Login extends React.Component {
  constructor(props) {
    super(props);
  }

  render() {
    return (
      <div className="base-container" ref={this.props.containerRef}>
        <div className="header">Login</div>
        <div className="content">
          <div className="image">
            <img src={loginImg} />
          </div>
          <div className="form">
            <div className="form-group">
              <label htmlFor="username">Username</label>
              <input type="text" name="username" placeholder="username" />
            </div>
            <div className="form-group">
              <label htmlFor="password">Password</label>
              <input type="password" name="password" placeholder="password" />
            </div>
          </div>
        </div>
        <div className="footer">
          <button type="button" className="btn">
            Login
          </button>
        </div>
      </div>
    );
  }
}

```

5. Web interface Registration Code

```
import React from "react";
import loginImg from "../../login.svg";

export class Register extends React.Component {
  constructor(props) {
    super(props);
  }

  render() {
    return (
      <div className="base-container" ref={this.props.containerRef}>
        <div className="header">Register</div>
        <div className="content">
          <div className="image">
            <img src={loginImg} />
          </div>
          <div className="form">
            <div className="form-group">
              <label htmlFor="username">Username</label>
              <input type="text" name="username" placeholder="username" />
            </div>
            <div className="form-group">
              <label htmlFor="email">Email</label>
              <input type="text" name="email" placeholder="email" />
            </div>
            <div className="form-group">
              <label htmlFor="password">Password</label>
              <input type="text" name="password" placeholder="password" />
            </div>
          </div>
        </div>
        <div className="footer">
          <button type="button" className="btn">
            Register
          </button>
        </div>
      </div>
    );
  }
}

```

6. Web App Landing Page Code

```
<!DOCTYPE html>
<html>
<head>
    <title></title>
    <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs"></script>
    <script src="https://cdn.jsdelivr.net/npm/@tensorflow-models/blazeface"></script>
    <link rel="stylesheet" href=
    "https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" integrity=
    "sha384-JcKb8q3iqJ61gNV9KGb8thSsNjpSL0n8PARn9HuZOnIxN0hoP+VmmDGMN5t9UJ0Z" crossorigin
    ="anonymous">
    <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css"
     integrity="sha384-lZN37f5QGtY3VHgisS14W3ExzMWZxybE1SJSEsQp9S+oqd12jhcu+A56Ebc1zFSJ"
    crossorigin="anonymous">
</head>
<style>

</style>
<body >
    <div class="container p-0" >
        <header >
          <nav class="navbar navbar-dark bg-dark rounded">
            <a class="navbar-brand">
                <i class="fas fa-camera"></i>
                <strong>Masks Detection</strong>
            </a>
          </nav>
        </header>
    </div>


    <div class="container ">
        <div class="row bg-light" style="height:480px">
            <video id="video" playsinline class="border " style=
            "margin:auto;display:inline-block;"></video>
            <canvas id="output" class="canvas-output" style=
            "margin:auto;position:relative;top:-480px;left:10px"></canvas>
            <div class="float-right">
                <a href="https://www.youtube.com/user/chirpieful">
                    <img src="../static/img/yicong.jpg" class="rounded" style=
                    "height:30px;width:30px;position:relative;top:-30px;">
                </a>
            </div>
        </div>
    </div>
</body>
<script>

    var model, mask_model, ctx, videoWidth, videoHeight, canvas;
    const video = document.getElementById('video');
    const state = {
      backend: 'webgl'
    };
    async function setupCamera() {
        const stream = await navigator.mediaDevices.getUserMedia({
            'audio': false,
            'video': { facingMode: 'user' },
        });
        video.srcObject = stream;
        return new Promise((resolve) => {
            video.onloadedmetadata = () => {
              resolve(video);
            };
        });
    }

    const renderPrediction = async () => {
        tf.engine().startScope()
```

```
        ctx.clearRect(0, 0, canvas.width, canvas.height);
        //estimatefaces model takes in 4 parameter (1) video, returnTensors,
        flipHorizontal, and annotateBoxes
        const predictions = await model.estimateFaces(video, true,false,false);
        const offset = tf.scalar(127.5);
        //check if prediction length is more than 0
        if (predictions.length > 0) {
            //clear context

            for (let i = 0; i < predictions.length; i++) {
                var text=""
                var start = predictions[i].topLeft.arraySync();
                var end = predictions[i].bottomRight.arraySync();
                var size = [end[0] - start[0], end[1] - start[1]];
                if(videoWidth<end[0] && videoHeight<end[0]){
                    console.log("image out of frame")
                    continue
                }
                var inputImage = tf.browser.fromPixels(video).toFloat()
                inputImage = inputImage.sub(offset).div(offset);
                inputImage=inputImage.slice([parseInt(start[1]),parseInt(start[0]),0],[
                parseInt(size[1]),parseInt(size[0]),3])
                inputImage=inputImage.resizeBilinear([224,224]).reshape([1,224,224,3])
                result=mask_model.predict(inputImage).dataSync()
                result= Array.from(result)
                ctx.beginPath()
                if (result[1]>result[0]){
                    //no mask on
                    ctx.strokeStyle="red"
                    ctx.fillStyle = "red";
                    text = "No Mask: "+(result[1]*100).toPrecision(3).toString()+"%";
                }else{
                    //mask on
                    ctx.strokeStyle="green"
                    ctx.fillStyle = "green";
                    text = "Mask: "+(result[0]*100).toPrecision(3).toString()+"%";
                }
                ctx.lineWidth = "4"
                ctx.rect(start[0], start[1],size[0], size[1])
                ctx.stroke()
                ctx.font = "bold 15pt sans-serif";
                ctx.fillText(text,start[0]+5,start[1]+20)
            }
        }
        //update frame
        requestAnimationFrame(renderPrediction);
        tf.engine().endScope()
    };

    const setupPage = async () => {
        await tf.setBackend(state.backend);
        await setupCamera();
        video.play();

        videoWidth = video.videoWidth;
        videoHeight = video.videoHeight;
        video.width = videoWidth;
        video.height = videoHeight;

        canvas = document.getElementById('output');
        canvas.width = videoWidth;
        canvas.height = videoHeight;
        ctx = canvas.getContext('2d');
        ctx.fillStyle = "rgba(255, 0, 0, 0.5)";

        model = await blazeface.load();

```

```
125             mask_model = await tf.loadLayersModel('../static/models/model.json');
126
127             renderPrediction();
128         };
129
130         setupPage();
131
132     </script>
133     <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity=
        "sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin=
        "anonymous"></script>
134     <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"
        integrity="sha384-9/reFTGAW83EW2RDu2S0VKaIzap3H66lZH81PoYlFhbGU+6BZp6G7niu735Sk7lN"
        crossorigin="anonymous"></script>
135     <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"
        integrity="sha384-B4gt1jrGC7Jh4AgTPSdUtOBvfO8shuf57BaghqFfPlYxofvL8/KUEfYiJOMMV+rV"
        crossorigin="anonymous"></script>
136
137     </html>
```

Appendix B: Use Case Details

Use Case:	Input Image
Primary Actors:	User
Brief Description:	This use case involves capturing of a Kenyan sign language notation done by another user. The user launches the camera and points at the target capture the image.
Pre-Condition:	Their need to exist a target doing a sign language notation and a working camera.
Post-Condition:	The camara captures an image with a target doing a sign language notation.
Major Steps Performed:	• The user launches the camera and points the camera to the target doing sign language notation. • The model captures the image with the sign language notation. • The model loads the image for pre-processing.

Use Case:	View Feedback
Primary Actors:	User
Brief Description:	This use case involves knowing the meaning of a Kenyan sign language notation done by another user. The user views the translated meaning of the Kenyan sign language notation in text.
Pre-Condition:	Inference carried out by the model from the sign language notation into readable text.
Post-Condition:	A translated readable text on the screen of the meaning of the Kenyan sign language notation.
Major Steps Performed:	• The model performs an inference of the Kenyan sign language notation. • The system returns the meaning of the sign language notation in readable text. • User views the results.

Use Case:	Train Model
Primary Actors:	Developer
Brief Description:	This use case involves training the model to be able to recognize new Kenyan sign language notations.
Pre-Condition:	Trained Model.
Post-Condition:	Up to date Model
Major Steps Performed:	• The developer trains the model with new Kenyan sign language notation images and maps the meanings of the image notations into text. • The model is updated with the new Kenyan sign language notations with their meanings in text. • In the case of incorrect matching of the text to the image notation, the administrator remaps them correctly.

Use Case:	Manage Users
Primary Actors:	Administrator
Brief Description:	This use case involves creation, modification and the deletion of users in the model.
Pre-Condition:	The model is working.
Post-Condition:	Updated users accounts.
Major Steps Performed:	• The administrator accesses the model. The administrator creates the user accounts be entering the name, email and password. The administrator saves the changes. • The administrator accesses the model. The administrator selects a user account and edits the three details (Name, email and password) and saves the changes. • The administrator accesses the user accounts and selects a user. The administrator deletes a user and saves the changes.

Appendix C: Consent Form

A Model for Sign Language Recognition for Kenyan Sign Language

Consent to take part in research

- I ……. voluntarily agree to participate in this research study.
- I understand that even if I agree to participate now, I can withdraw at any time or refuse to answer any question without any consequences of any kind.
- I understand that I can withdraw permission to use data from my interview within two weeks after the interview, in which case the material will be deleted.
- I have had the purpose and nature of the study explained to me in writing and I have had the opportunity to ask questions about the study.
- I understand that I will not benefit directly from participating in this research.
- I agree to my interview being audio-recorded or in cases of photos, partial masking of facial identity may be implemented subject to participants consent.
- I understand that all information I provide for this study will be treated confidentially.
- I understand that in any report on the results of this research, my identity will remain anonymous. This will be done by changing my name and disguising any details of my interview which may reveal my identity or the identity of the people I speak about.
- I understand that disguised extracts from my interview may be quoted in the researcher's thesis.
- I understand that signed consent forms and original audio recordings or photos will be retained at Strathmore university until the exam board confirms the results of the researcher's thesis.
- I understand that a transcript of my interview in which all identifying information has been removed will be retained for two years from the date of the exam board.

- I understand that under freedom of information legalization, I am entitled to access the information I have provided at any time while it is in storage as specified above.

- I understand that I am free to contact any of the people involved in the research to seek further clarification and information.

147823 Geoffrey K. Wanjala
MSc in IT
geoffrey.wanjala@strathmore.edu
Strathmore University

Signature of research participant

---------- ----------

Signature of participant Date

Signature of researcher

I believe the participant is giving informed consent to participate in this study

---------- ----------

Signature of researcher Date

Appendix D: Strathmore University Ethical Review Board Approval

Strathmore UNIVERSITY

16th March 2023

Mr Wanjala Geoffrey Kasembeli,
geoffrey.wanjala@strathmore.edu

Dear Mr Wanjala,

RE: A Model for Sign Language Recognition for Kenyan Sign Language

This is to inform you that SU-ISERC has reviewed and **approved** your above **SU- master's** research proposal. Your application reference number is **SU-ISERC1606/23.** The approval period is from **16th March 2023 to 15th March 2024.**

This approval is subject to compliance with the following requirements:

i. Only approved documents including (informed consents, study instruments, and MTA) will be used
ii. All changes including (amendments, deviations, and violations) are submitted for review and approval by SU-ISERC.
iii. Death and life-threatening problems and serious adverse events or unexpected adverse events whether related or unrelated to the study must be reported to SU-ISERC within 48 hours of notification
iv. Any changes, anticipated or otherwise, that may increase the risks or affect the safety or welfare of study participants and others or affect the integrity of the research must be reported to SU-ISERC within 48 hours
v. Clearance for the export of biological specimens must be obtained from relevant institutions.
vi. Submission of a request for renewal of approval at least 60 days prior to the expiry of the approval period. Attach a comprehensive progress report to support the renewal.
vii. Submission of an executive summary report within 90 days of completion of the study to SU-ISERC.

Before commencing your study, you will be expected to obtain a research license from National Commission for Science, Technology, and Innovation (NACOSTI) https://research-portal.nacosti.go.ke/ and obtain other clearances needed.

Yours sincerely,

for: **Dr Ben Ngoye,**
Secretary; SU-ISERC

Cc: Mr Ambrose Rachier,
Chairperson; SU-ISERC

STRATHMORE UNIVERSITY INSTITUTIONAL SCIENTIFIC AND ETHICAL REVIEW COMMITTEE
(SU-ISERC)
17-Mar-2023
Email:ethicsreview@strathmore.edu
P.O BOX 59857-00200
NAIROBI-KENYA

Ole Sangale Rd, Madaraka Estate. PO Box 59857-00200, Nairobi, Kenya. Tel +254 (0)703 034000
Email admissions@strathmore.edu www.strathmore.edu

Appendix E: NACOSTI Research Permit/License

REPUBLIC OF KENYA

NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Ref No: **666717**

Date of Issue: **04/April/2023**

RESEARCH LICENSE

This is to Certify that Mr. Geoffrey Kasembeli Wanjala of Strathmore University, has been licensed to conduct research as per the provision of the Science, Technology and Innovation Act, 2013 (Rev.2014) in Nairobi on the topic: A Model for Sign Language Recognition for Kenyan Sign Language for the period ending : 04/April/2024.

License No: **NACOSTI/P/23/24876**

666717

Applicant Identification Number

Director General
NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Verification QR Code

NOTE: This is a computer generated License. To verify the authenticity of this document, Scan the QR Code using QR scanner application.

See overleaf for conditions

Appendix F: Similarity Checker Report

Submission status

Attempt number	This is attempt 3.
Submission status	Submitted for grading
Grading status	Not graded
Time remaining	Assignment was submitted 24 days 23 hours early
Last modified	Tuesday, 6 June 2023, 12:18 AM
File submissions	147823_A Model for Sign Language Recognition for Kenyan Sign Language.docx6 June 2023, 12:18 AM Turnitin ID: 2109781817 18%