”REAL-TIME HAND GESTURE RECOGNITION USING CONVOLUTIONAL NEURAL NETWORK”

Minor project report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

Abiram Ganiyada	(19UECS0295)	**(15600)**
R.Bhuvana Shanmukha sai sivani	(19UECS0840)	**(12597)**
G. sai rishi	(19UECS0346)	**(15049)**

Under the guidance of
Mr.S.Girirajan.M.TECH.,
ASSISTANT PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A Grade
CHENNAI 600 062, TAMILNADU, INDIA

June , 2022

CERTIFICATE

It is certified that the work contained in the project report titled ”REAL-TIME HAND GESTURE RECOGNITION USING CONVOLUTIONAL NEURAL NETWORK” by Abiram Ganiyada (19UECS0295), R.Bhuvana Shanmukha sai sivani (19UECS0840) ,G. sai rishi(19UECS0346) has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor
Mr.S.Girirajan.,
ASSISTANT PROFESSOR
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr.Sagunthala R&D
Institute of Science & Technology
June,2022

Signature of Head of the Department
Dr. V. Srinivasa Rao
Professor & Head
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr.Sagunthala R&D
Institute of Science & Technology
June,2022

DECLARATION

We declare that this written submission represents my ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)
(Abiram Ganiyada)
Date: / /

(Signature)
(R,Bhuvana shanmukha sai sivani)
Date: / /

(Signature)
(G.sai rishi)
Date: / /

APPROVAL SHEET

This project report entitled ”REAL-TIME HAND GESTURE RECOGNITION USING CONVOLUTIONAL NEURAL NETWORK” by Abiram Ganiyada (19UECS0295), R.Bhuvana Shanmukha sai sivani (19UECS0840) ,G. sai rishi(19UECS0346) is approved for the degree of B.Tech in Computer Science & Engineering.

Examiners **Supervisor**

Mr.S.Girirajan, M.TECH,

Date: / /

Place:

ACKNOWLEDGEMENT

We express our deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO),D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

We are very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN,** for providing us with an environment to complete our project successfully.

We record indebtedness to our **Dean & Head, Department of Computer Science & Engineering Dr.V.SRINIVASA RAO, M.Tech., Ph.D.,** for immense care and encouragement towards us throughout the course of this project.

We also take this opportunity to express a deep sense of gratitude to our Internal Supervisor **Mr.S.GIRIRAJAN,M.TECH** for his/her cordial support, valuable information and guidance, he/she helped us in completing this project through various stages.

A special thanks to our **Project Coordinators Mr. V. ASHOK KUMAR, M.Tech., Ms. C. SHYAMALA KUMARI, M.E., Ms.S.FLORENCE, M.Tech.,** for their valuable guidance and support throughout the course of the project.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

Abiram Ganiyada	(19UECS0295)
R.Bhuvana Shanmukha sai sivani	(19UECS0840)
G. sai rishi	(19UECS0346)

ABSTRACT

The interface between humans and machines involves the most common modes of communication such as speech and hand gestures. These types of interactions are intuitive and user-friendly. In general, people have used remote controls and joysticks as controlling devices for many human–machine interfaces. However, to operate those devices, a trained user is needed. On the other hand, a hand-gesture-based interface provides higher flexibility while also being user-friendly because the user has to operate a machine using only his hand in front of the camera. Several applications that use static hand gesture recognition systems are sign language interpretation, automatic television control, smart home interactive control, gaming, control of a software interface and control of virtual environments. In real-time application, high accuracy and robustness of background interference are required for the design of an efficient gesture recognition system. Therefore, the precision of the hand gesture recognition (HGR) system still provides several challenges to researchers.

Keywords: Convolution Neural Network, Hand Gesture Recognition,Graphical User Interface,Deep Learning.

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

AI	Artificial Intilligence
CNN	Convolutional Neural Networks
FDDB	Face Detection Data Set and Benchmark Home.
HGR	Hand gesture recogition
NLP	Natural Language Processing
SRS	System Requirement Specification

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Introduction

Humans are able to recognize body and sign language easily. This is possible due to the combination of vision and synaptic interactions that were formed along brain development . In order to replicate this skill in computers, some problems need to be solved: how to separate objects of interest in images and which image capture technology and classification technique are more appropriate, among others. The evolution of computing and the ease of access of new technologies motivated the development , which are examples of innovation in input device technologies . In this way, these devices are capable of capturing human gestures, developing a new medium of human-machine interaction. The uses of these devices are present in the most diverse areas, such as robotics, medicine, sign language translation, computer graphics, and augmented reality.In this work, we used two image bases of 24 gestures, some segmentation techniques and the use of convolutional neural networks (CNNs) for classification. Thus, with the proposed methodology, we demonstrated that with simple architectures of convolutional neural networks, it is possible to achieve excellent results for static gesture classification. We compared the proposed architectures with other existing networks in the literature and other gesture recognition methodologies. In the next sections, we present a brief description of the techniques we used, our proposed methodology, and the experiments we carried out. The final sections of this work show the results we obtained, a discussion and comparison with other works and, lastly, our conclusions and perspectives for future work.In this work, in order to provide a practical solution, we have developed a vision based gesture recognition approach using deep convolutional neural networks (CNNs) on raw video data. Currently, CNNs provide the state-ofthe-art results for not only image based tasks such as object detection, image segmentation and classification, but also for video based tasks such as activity recognition and action localization as well as gesture recognition

1.2 Aim of the project

The main aim of the project is to recognize human intentions through noncontact communication modes as humans do, such as by sound, facial expressions, body language, and gestures. Among these modes, hand gestures are an important part of human language, and hence, the development of hand gesture recognition affects the nature and flexibility of human–computer interaction.

1.3 Project Domain

We used the Deep Learning Algorithm called convolution neural network (CNN) could be a sort of artificial neural network algorithm employed in image recognition and processing that's specifically designed to process pixel data. CNNs are powerful image processing, AI that use the deep learning to perform the both generative and the descriptive tasks,often using machine vision that the features image recognition and video recognition,together with recommander systems and the natural language processing (NLP). A CNN uses a system very like a multilayer perceptron that has been designed for reduced processing requirements. The layers of a CNN include an input layer,an output layer and a hidden layer that has multiple convolution layers, pooling layers,fully connected layers and normalization layers.

1.4 Scope of the Project

we demonstrated that with simple architectures of convolutional neural networks, it is possible to achieve excellent results for static gesture classification. We compared the proposed architectures with other existing networks in the literature and other gesture recognition methodologies. In the next sections, we present a brief description of the techniques we used, our proposed methodology, and the experiments we carried out.

Chapter 2

LITERATURE REVIEW

[1].Krishna Dharavath, Fazal Ahmed Talukdar and Rabul Hussain Laskar(ETAL) (2021), "*Improving Face Recognition Rate with Image Preprocessing*", Indian Journal of Science and Technology, 7(8), pp. 1170-1175.

Quality of image plays a vital role in increasing face recognition rate. A good quality image gives better recognition rate than noisy images. It is more difficult to extract features from such noisy images which in-turn reduces face recognition rate. To overcome problems occurred due to low quality image,pre-processing is done before extracting features from the image. In this paper we will analyze the effect of pre-processing prior to feature extraction process with respect to the face recognition rate. This also gives a qualitative description of various pre-processing techniques and feature extraction schemes that were used for our analysis.Results were analyzed with the help of bar graphs.the combined method of feature extraction shows superior performance than individual feature extraction schemes. Also, this combined method gives good recognition results even without pre-processing of the image..

[2] Ankit V Ponkia and Jitendra Chaudhari(ETAL)(2020), "*Face Recognition Using PCA Algorithm*", Inventi Rapid: Image Video Processing Journal, 4(1), pp. 519-524.

One of the simplest and most effective Principal Component Analysis(PCA) approaches used in face recognition systems is the so-called eigenface approach. This approach transforms faces into a small set of essential characteristics, eigenfaces, which are the main components of the initial set of learning images (training set). Recognition is done by projecting a new image in the eigenface subspace, after which person is classified by comparing its position in eigenface space with position of known individuals. The advantage of this approach over other face recognition systems is in its simplicity, speed and insensitivity to small or gradual changes on the face. The problem is limited to files that can be used to recognize the face. Namely, the images must be vertical frontal views of human faces.

From the above literature survey, we conclude that the performance of the HGR system mainly depends on system accuracy and distinguishable features between the gesture classes in a dataset. Therefore, in this work, a score-level fusion technique between two finetuned CNNs is proposed to recognize static hand gestures in the vision-based environment.

[3]. B J Bipin, K Nihar and C. Adarsh(ETAL)(2020).”*Comparative Binarization Approach for Degraded Agreement Document Image from Various Pharmacies*”, IEEE journal, 12(4), pp. 806-814.

Inputting image is the initial step of this procedure. Images are captured by camera or obtained by using secondary sources such as scanner. After completing the above step, we need to verify the image to know the format of the image. When the inputted image is in RGB format then the image will be transferred into grayscale format. But, if the image is in grayscale, then it will be preprocessed in the next stage. The Preprocessing step is the next stage for the grayscale image.Usually,preprocessing is required to eliminate the imperfections of the input image. This step facilitates the hybridization step. The contrast of grayscale image is made better and enhanced by applying CLAHE (Contrast Limited Adaptive Histogram Equalization). In each segment of the image, it improves the local contrast and restricts the amplification of noise. After grayscale enhancement, we go through a hybrid thresholding technique by uniting four ideal methods: Otsu’s technique, Sauvola’s technique, Nick’s

method and local adaptive thresholding technique. Otsu’s technique performs best for the images with apparent bi-modal model and Sauvola’s technique makes enhanced outcome with least amount of noise and but it is not able to differentiate text with very low contrast and illuminated-texture.

Chapter 3

PROJECT DESCRIPTION

3.1 Existing System

The existing method defined called the deep CNN method which for the recognition of hand gesture. Two methods are used in deep CNN for achieving the hand gesture. This proposed method can be used in the home appliances and the interactions between the system and the user. The convex hull of the hand involved in the recognition process can use skin separation. The lens facing is a more complex process in the hand gesture. The reason for approaching the convolutional neural network is that it is the efficient recognition model for hand gestures. It provides solutions not only for the images also for the various image processes. The hand gesture process several characteristics that need to be proved as accuracy, efficiency, recognition, etc., some proposed methods consisting different CNN and it forces the power and the storage.

Disadvantages of existing system

Latency and faster detection rate are most vital points to contemplate which must be improved in preceding upto now detection in crowded places and associating face identity information is crucial to impose fine and restriction rules for policing just in case of no gesture in position.

3.2 Proposed System

An overview of the proposed hand gesture recognition system is the recognition of static hand gesture images is achieved by the following steps: data acquisition, preprocessing and recognition of hand gestures using proposed technique. it explains the various stages. Initially, the input image is selected. The input image is taken from the different hand gesture databases. A different dataset is also chosen for collecting more data about the image. The preprocessing can be used for removing unwanted information about the system. And the final steps consist of feature extraction and feature prediction. The above block diagram explains the hand gesture process.

Advantages of proposed system:

Pros:
- CNN model is strong in identifying object detectioN.
- It proves efficient in understanding emotions and posture recognition .
- Approach this system to gesture detection with hand movement.
-It obtained effective lead to achieving accurate identification observation in several angle with the assistance of CNN.

cons:
-Latency and faster detection rate are most important points .
- It improved in preceding cases up to more movements of multiple people.

3.3 Feasibility Study

An important outcome of preliminary investigation is the determination of system requests is possible. This is often possibly providing it's feasible within limited resource and time. The numerous feasibilities that need to be analyzed are:

3.3.1 Economic Feasibility

Economic Feasibility is an assessment of economic justification for a computer based project. As hardware was installed from the start for many purposes thus the price on project of hardware is low. Since the system could also be a network based, any number of employees connected to the Local Area Network(LAN) within that organization we can use this tool at any time. The Virtual Private Network is to be developed using this resources of the organization. that the project is economically feasible.

3.3.2 Technical Feasibility

Technical Feasibility is that the assessment of the technical resources of the organization. The organization needs IBM compatible machines with a graphical programme connected to the online and Intranet. The system is developed for platform Independent environment. The technical feasibility has been administered. The system is technically feasible for development and will be developed with this facility.

3.3.3 Social Feasibility

Social feasibility is also an full study on how one interacts with others within a system or a corporation. Social impact analysis is an exercise aimed toward identifying and analyzing such impacts so on grasp the size and reach of the project's social impacts.This project is acknowledge the persons with disabilities and it helps them to acknowledge everything.

3.4 System Specification

A System Requirements Specification (SRS) could even be a document or set of documentation that describes the features and behaviour of a system or software application.

3.4.1 Hardware Specification

For developing the application the following are the Hardware Requirements
- Processor : Minimum Intel i3
- RAM : Minimum 4 GB
- Space on Hard Disk : 1TB
- Graphics card : 512GB
- Operating system : windows9, Linux

3.4.2 Software Specification

- Anaconda 3.7
- OpenCV
- keras 2.2 version
- Tensor flow 1.00 version

3.4.3 Standards and Policies

Anaconda (Python distribution)
Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS.

TENSORFLOW(Machine learning software library)
This standard defines a framework and architectures for machine learning in which a model is trained using encrypted data that has been aggregated from multiple sources and is processed by a trusted third party. It specifies functional components, workflows, security requirements, technical requirements, and protocols.

Open CV (computer vision library)
This standard defines deep learning based metrics of content analysis and quality of experience (QoE) assessment for visual contents, which is an extension of Standard for the Quality of Experience (QoE) and Visual-Comfort Assessments of Three-Dimensional (3D) Contents Based on Psychophysical Studies (IEEE 3333.1.1™) and Standard for the Perceptual Quality Assessment of Three Dimensional (3D) and Ultra High Definition (UHD) Contents (IEEE 3333.1.2™)

Chapter 4

METHODOLOGY

4.1 General Architecture

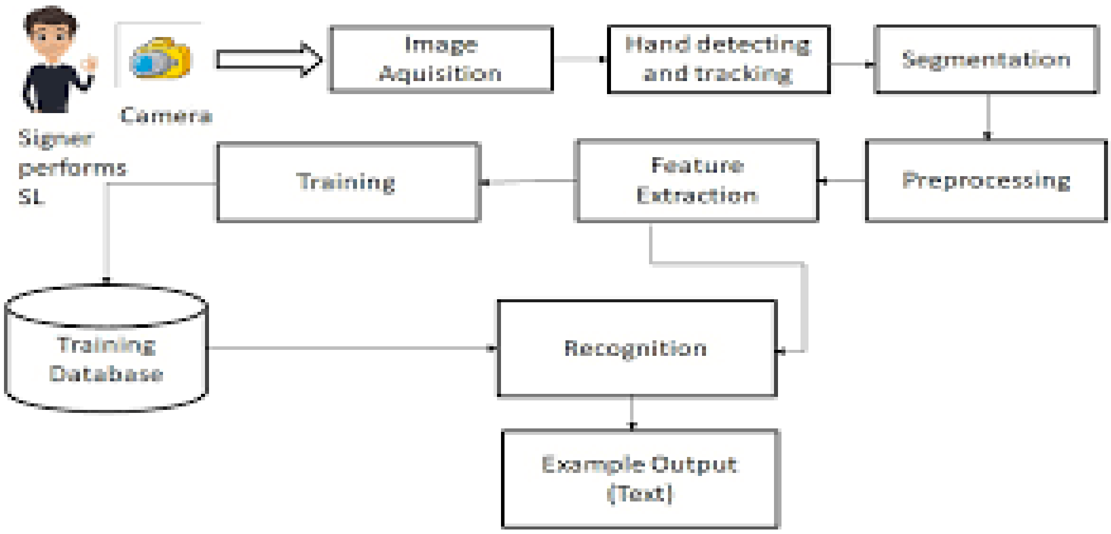


Figure 4.1: **Architecture diagram of hand gesture using CNN**

Description:

Initially, the images were taken in the form of RGB. But the RGB color is more sensitive to the different light conditions. So, change this format and converted it to the YCbCr.This format is dealing the information held in the image. This format requires a white background. Because some background interferences the hand gesture image recognition. Previous articles explained that this input image is converted in binary format. Before processing, it eliminates all the noise inside the background.For removing, the noises behind the background this process uses some noise elimination methods to remove all the smudges considered with the pixels. These smudges should less than the pixel value. The pixel value is denoted as P the process hole filling is done in the converted image. The holes mean that the combination of several background pixels but the pixel cannot remove the holes in the images. These two processes are held in the binary images.

4.2 Design Phase

4.2.1 Data Flow Diagram

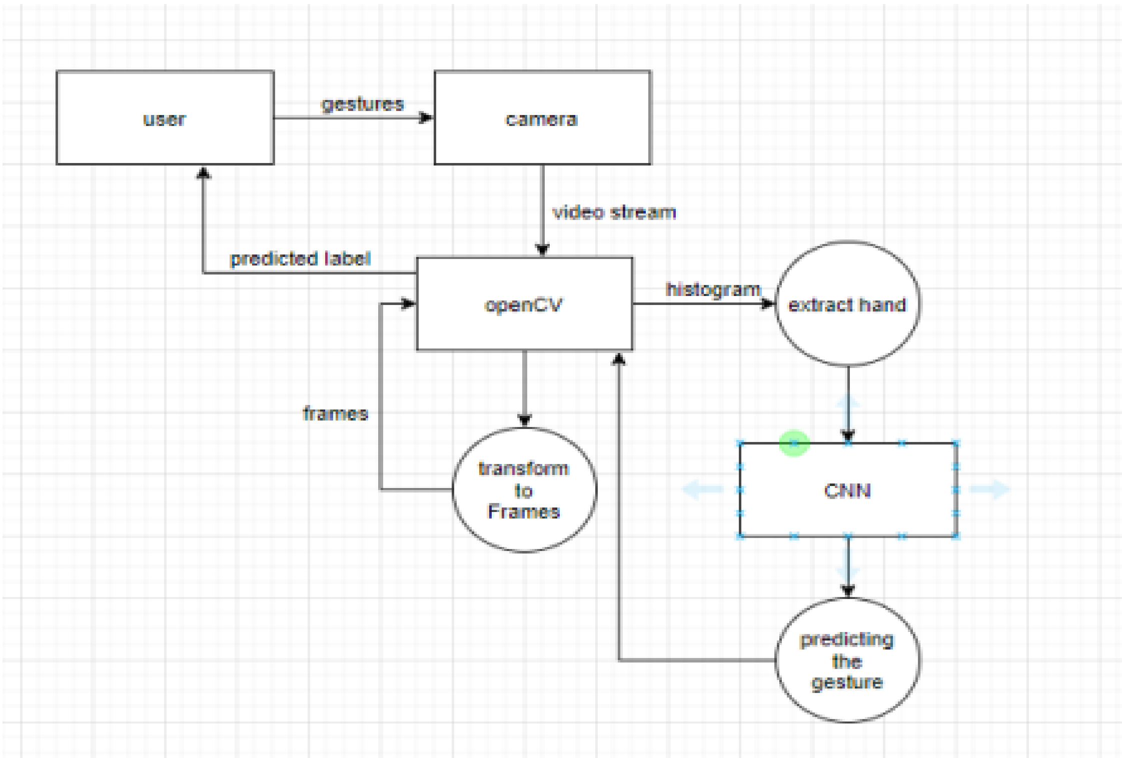


Figure 4.2: **Data Flow Diagram of sign language recognition system**

Description

A dataflow represents a package of information flowing between two objects in the data-flow diagram, Data flows are used to model the flow of information into the system, out of the system and between the elements within the system. Representation.The convolutional neural networks are performed in hand gesture recognition for getting accurate results. This explains the various techniques used in the proposed technique as explained in the block diagram.

4.2.2 Use Case Diagram

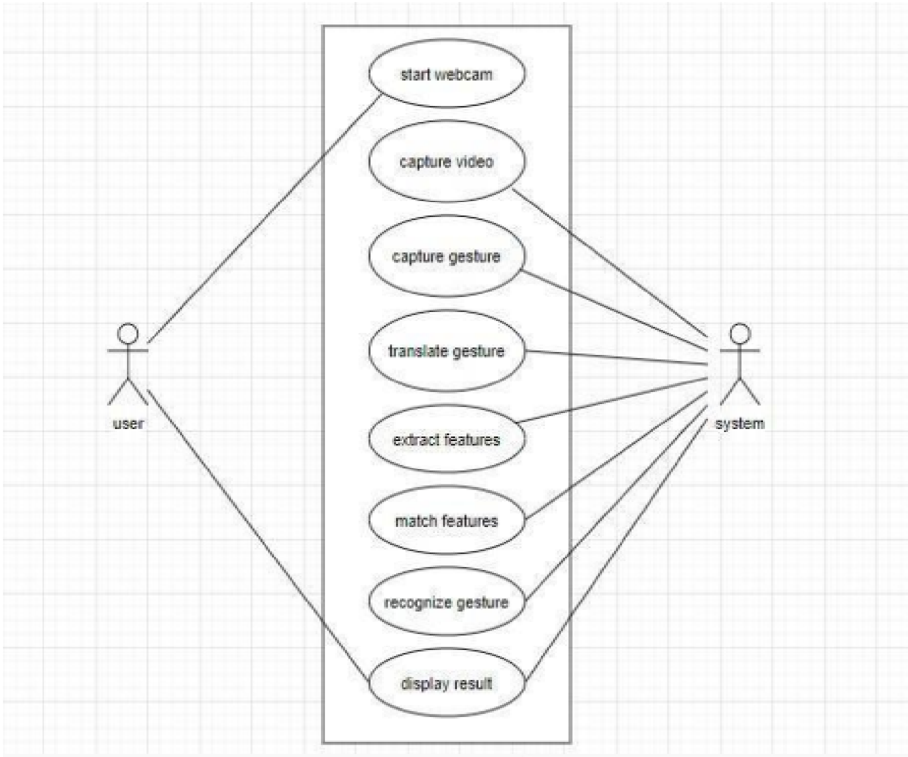


Figure 4.3: **Use case diagram of sign language recognition system**

Description

Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified. When the initial task is complete, use case diagrams are modelled to present the outside view. Data is collected from the user and building a model should be done based on data and those images are taken as input. Input images can also be taken through live streaming. After applying the CNN algorithm classification should be done and extract the images then detection of the hand gesture movements, The output of this gives the text about the gesture movement,

4.2.3 Class Diagram

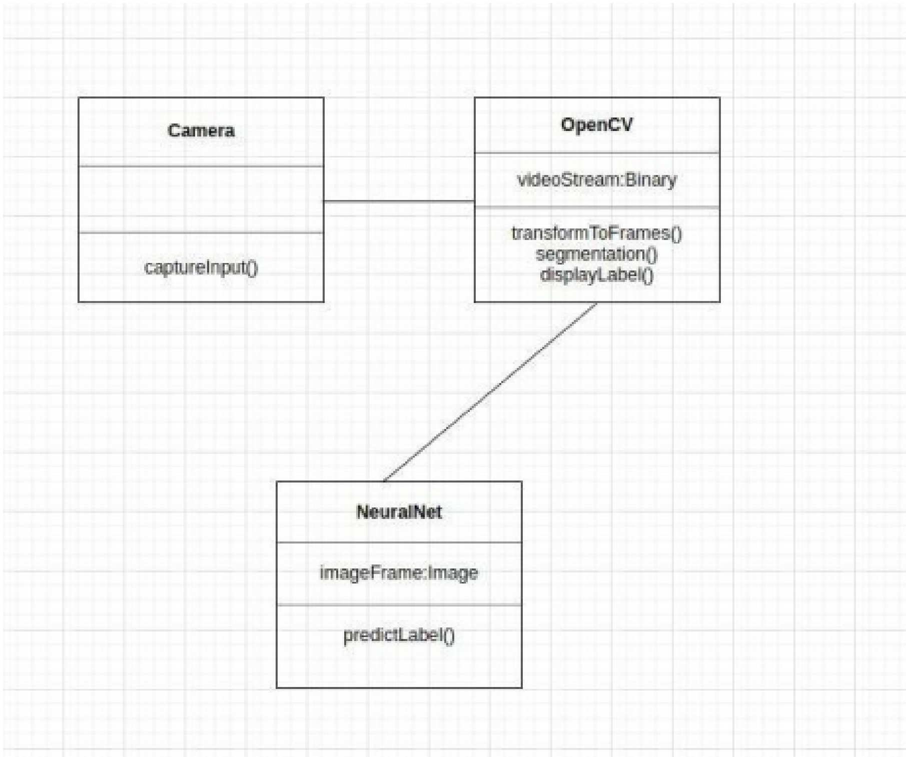


Figure 4.4: **Class diagram of sign language recognition system**

Description

Class diagrams model class structure and contents using design elements such as classes, packages and objects. Class diagram describe the different perspective when designing a system-conceptual, specification and implementation. Classes are composed of three things: name, attributes, and operations. Class diagram also display relationships such as containment, inheritance, association etc. The association relationship is most common relationship in a class diagram. The association shows the relationship between instances of classes.

4.2.4 Sequence Diagram

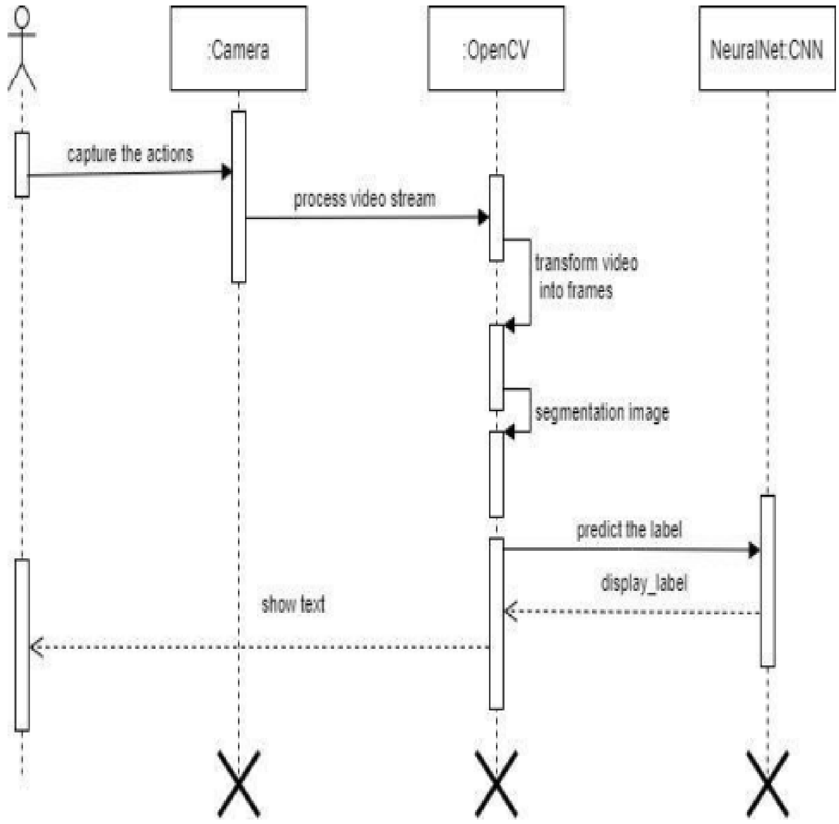


Figure 4.5: **Sequence diagram of sign language recognition system**

Description

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios. A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner. If the lifeline is that of an object, it demonstrates a role. Leaving the instance name blank can represent anonymous and unnamed instances

4.2.5 Collaboration diagram

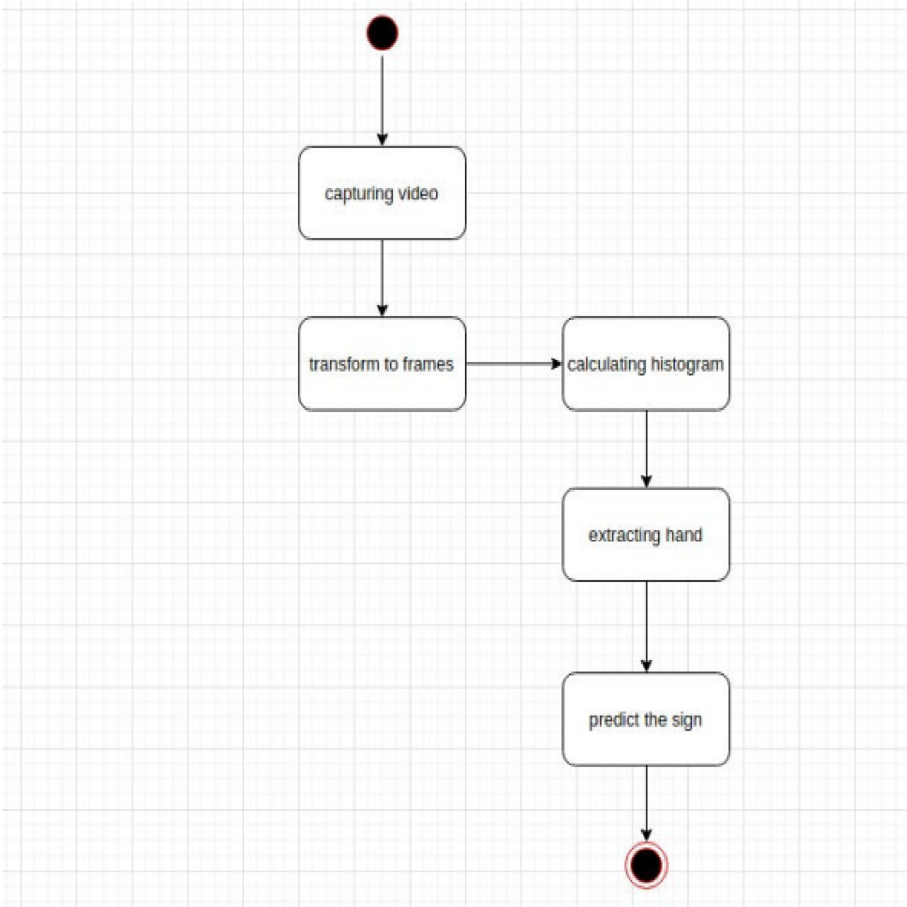


Figure 4.6: **Collaboration diagram of sign language recognition system.**

Description

Data is collected from the website kaggle.com and stored it in database.The stored data should be collected from the user and upload the data for training and testing by building a model. The input images should be extracted and classify the data to detect mask/no mask. The input should be taken as live streaming also. The output should be displayed as green or red annotation representing the mask or not.

4.2.6 Activity Diagram

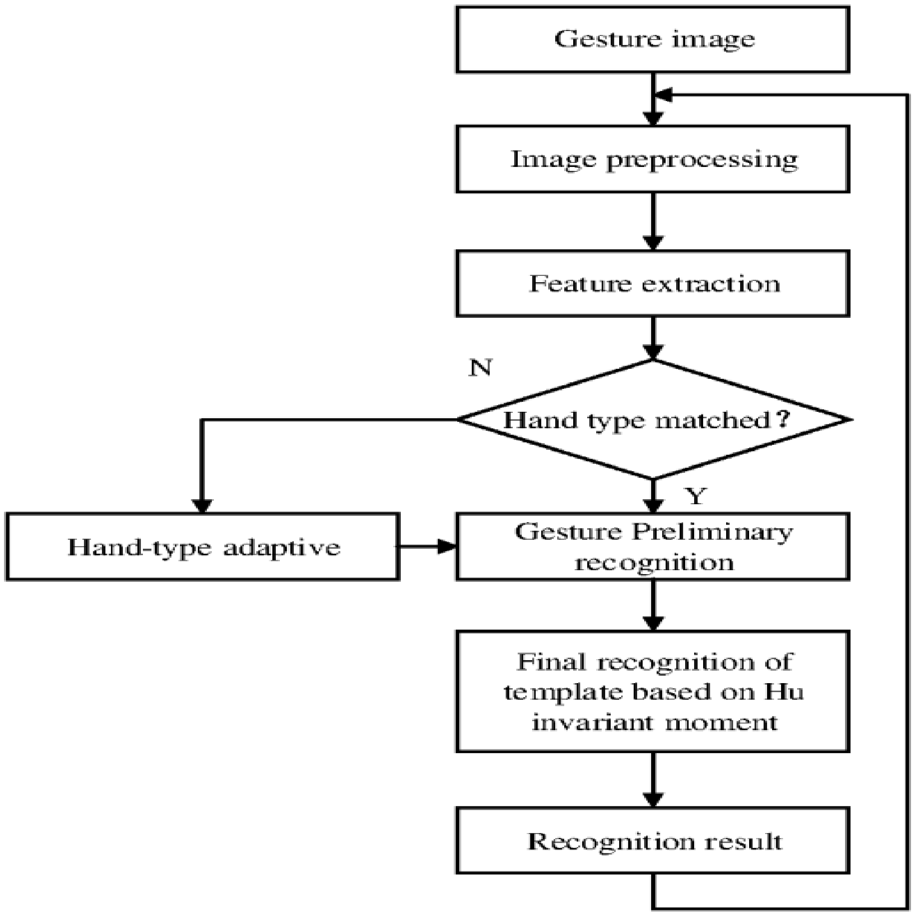


Figure 4.7: **Activity diagram of sign gesture recognition system.**

Description

An activity diagram is a behavioral diagram i.e. it depicts the behavior of a system. An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed.

4.3 Algorithm & Pseudo Code

4.3.1 Algorithm

Step 1: Start
Step 2: Import the datasets needed.
Step 3: Pre-process the data
Step 4: Split the data set into training and testing data.
Step 5: Apply the suitable algorithm to train the data.

Step 6: Send the testing data to trained model
Step 7: Predicting the output

4.3.2 Pseudo Code

begin
Read Data set file using it's path
train the data
labels dict=hand recognition:'movement'
gesture dict=recognise thing(0,0,255)
If video= hand gesture
return classes
else
NO detection

4.4 Module Description

4.4.1 Dataset collection

The sample data has been collected from website which consists of all the records of individuals. The dataset collected consists of nearly 4,500 datasets. Those datasets will be used as training and testing data set. we've got collected the datasets from the web site "kaggle.com".

4.4.2 Splitting Dataset Into Training And Testing Data

The train-test split procedure is suitable after you have a awfully large dataset, a costly model to coach, or require a decent estimate of model performance quickly. The way to use the scikit-learn machine learning to perform the train test split procedure. The way to evaluate machine learning algorithms for classification and regression using the train test split.

4.5 Steps to execute/run/implement the project

4.5.1 Collecting the data

For this project, we need datasets containing with mask and no mask, those datasets is taken from the website called ”kaggle.com”. With mask and without mask images are saved in seperate folder called ”dataset”.

4.5.2 Model implementation

We used Tensorflow for implementing the project, input web video can be taken in two ways detection of wrist from video and detection of gesture movement from webcam/live streaming. Here, Collected data sets are sent to training and that data will be saved as Model files with the extension of ”.h5” file.

4.5.3 Model Testing and Analyzing the Result

During this phase a second set of data is loaded. This data set has never been seen by the model and therefore it’s true accuracy will be verified. After the model training is complete, and it is understood that the model shows the right results.

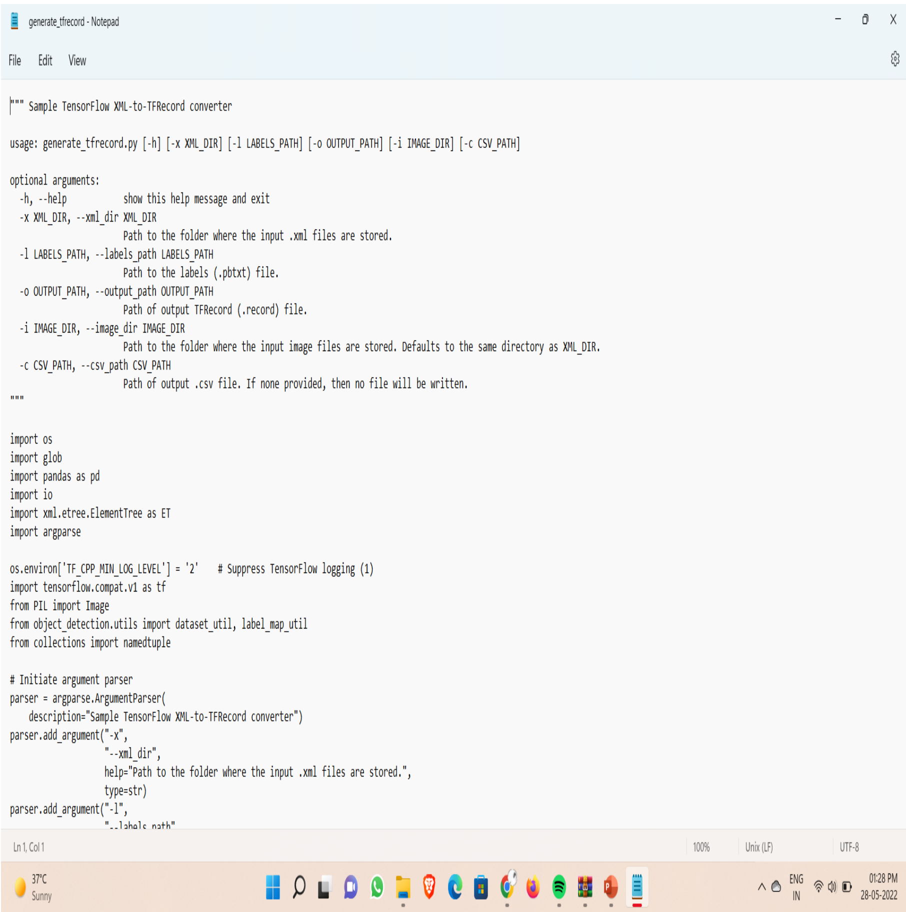
Chapter 5

IMPLEMENTATION AND TESTING

Implementation is the venture when the hypothetical plan is transformed out into the working framework. Along with these lines there is a very well accomplished method which may be viewed as the most basic stage in accomplishing a fruitful new framework and in giving the client,certainty that the new framework will work and be effective. The execution includes some cautious arranging,examination of the current framework and the requirements on usage,the planning of techniques to accomplish changeover and assessment of changeover strategies.

5.1 Input and Output

5.1.1 Input Design




```
""" Sample TensorFlow XML-to-TFRecord converter

usage: generate_tfrecord.py [-h] [-x XML_DIR] [-l LABELS_PATH] [-o OUTPUT_PATH] [-i IMAGE_DIR] [-c CSV_PATH]

optional arguments:
  -h, --help            show this help message and exit
  -x XML_DIR, --xml_dir XML_DIR
                        Path to the folder where the input .xml files are stored.
  -l LABELS_PATH, --labels_path LABELS_PATH
                        Path to the labels (.pbtxt) file.
  -o OUTPUT_PATH, --output_path OUTPUT_PATH
                        Path of output TFRecord (.record) file.
  -i IMAGE_DIR, --image_dir IMAGE_DIR
                        Path to the folder where the input image files are stored. Defaults to the same directory as XML_DIR.
  -c CSV_PATH, --csv_path CSV_PATH
                        Path of output .csv file. If none provided, then no file will be written.
"""

import os
import glob
import pandas as pd
import io
import xml.etree.ElementTree as ET
import argparse

os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'    # Suppress TensorFlow logging (1)
import tensorflow.compat.v1 as tf
from PIL import Image
from object_detection.utils import dataset_util, label_map_util
from collections import namedtuple

# Initiate argument parser
parser = argparse.ArgumentParser(
    description="Sample TensorFlow XML-to-TFRecord converter")
parser.add_argument("-x",
                    "--xml_dir",
                    help="Path to the folder where the input .xml files are stored.",
                    type=str)
parser.add_argument("-l",
                    "--labels_path"
```

5.1.2 Output Design

5.2 Testing

5.3 Types of Testing

In order to make sure that the system does not have errors, the different levels of testing strategies that are applied at different phases of software development are :

5.3.1 Unit testing

Unit testing indicates the test cases planning which validates that the used programming logic is working properly or not, that program inputs produce correct outputs. 22 Validation for all the decision branches and code flow should be done. it's the testing of individual software units of the appliance .It is done after the completion of a non public unit before integration. Here,we use threaded video stream to grab the frame and then we will read the frame and resize the frame and then detect faces if they are wearing mask or not using frame, face Net, mask Net and determines the class label and color we'll use to draw the bounding box and text.

Input

```
    tf_example = tf.train.Example(features=tf.train.Features(feature={
        'image/height': dataset_util.int64_feature(height),
        'image/width': dataset_util.int64_feature(width),
        'image/filename': dataset_util.bytes_feature(filename),
        'image/source_id': dataset_util.bytes_feature(filename),
        'image/encoded': dataset_util.bytes_feature(encoded_jpg),
        'image/format': dataset_util.bytes_feature(image_format),
        'image/object/bbox/xmin': dataset_util.float_list_feature(xmins),
        'image/object/bbox/xmax': dataset_util.float_list_feature(xmaxs),
        'image/object/bbox/ymin': dataset_util.float_list_feature(ymins),
        'image/object/bbox/ymax': dataset_util.float_list_feature(ymaxs),
        'image/object/class/text': dataset_util.bytes_list_feature(classes_text),
        'image/object/class/label': dataset_util.int64_list_feature(classes),
    }))
    return tf_example


def main(_):

    writer = tf.python_io.TFRecordWriter(args.output_path)
    path = os.path.join(args.image_dir)
    examples = xml_to_csv(args.xml_dir)
    grouped = split(examples, 'filename')
    for group in grouped:
        tf_example = create_tf_example(group, path)
        writer.write(tf_example.SerializeToString())
    writer.close()
    print('Successfully created the TFRecord file: {}'.format(args.output_path))
    if args.csv_path is not None:
        examples.to_csv(args.csv_path, index=None)
        print('Successfully created the CSV file: {}'.format(args.csv_path))
```

Test result

Correct: The UI design will be displayed for identifying faces.
Incorrect: The UI design will not be displayed.

5.3.2 System testing

System testing ensures that the whole integrated package meets requirements. It tests a configuration to substantiate known and predictable results. System testing depends on code flows and process descriptions and mainly on pre driven links and integration points. Now,we have to extract the confidence associated with the detection and then filter out the weak detectionsby ensuring the confidence if greater than the minimum confidence. Compute the coordinates of bounding box for the image and then extract the face ROI, convert it from BGR to RGB channel. This testing is done to check by giving input and checking whether expected or appropriate output is producing or not.

Input

```
import os
import glob
import pandas as pd
import io
import xml.etree.ElementTree as ET
import argparse

os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'    # Suppress TensorFlow logging (1)
import tensorflow.compat.v1 as tf
from PIL import Image
from object_detection.utils import dataset_util, label_map_util
from collections import namedtuple

# Initiate argument parser
parser = argparse.ArgumentParser(
    description="Sample TensorFlow XML-to-TFRecord converter")
parser.add_argument("-x",
                    "--xml_dir",
                    help="Path to the folder where the input .xml files are stored.",
                    type=str)
parser.add_argument("-l",
                    "--labels_path",
                    help="Path to the labels (.pbtxt) file.", type=str)
parser.add_argument("-o",
                    "--output_path",
                    help="Path of output TFRecord (.record) file.", type=str)
parser.add_argument("-i",
                    "--image_dir",
                    help="Path to the folder where the input image files are stored. "
                         "Defaults to the same directory as XML_DIR.",
                    type=str, default=None)
```

Test Result

Correct: Image is captured correctly.

Incorrect: Sometimes,image may not be captured correctly

5.3.3 Test Result

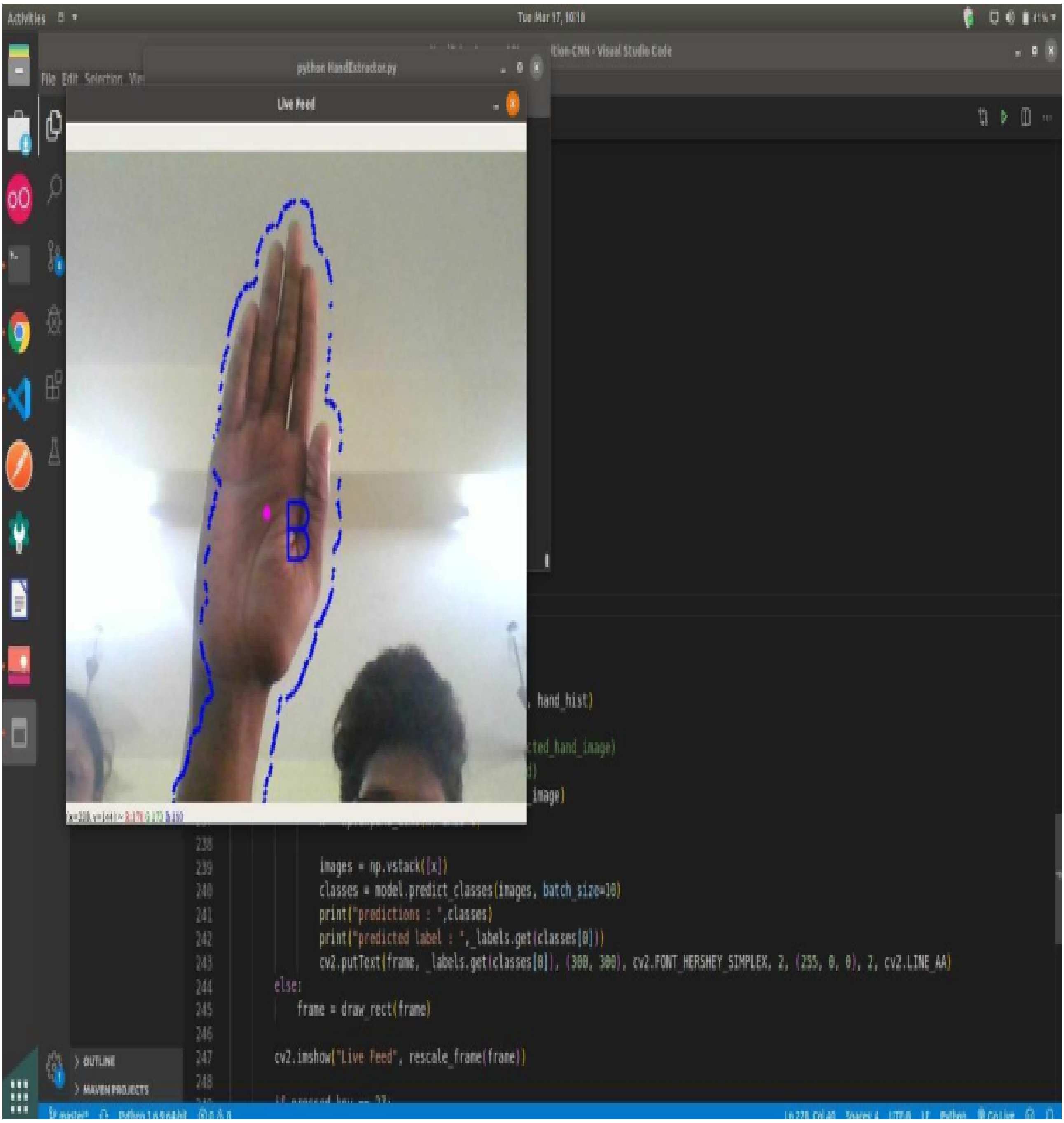


Figure 5.1: **Test Image**

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

The proposed method on multiple subjects, we trained and tested the hand images of the seven gesture types from seven subjects. The average recognition rate was 95.96 percent. The proposed system also had the satisfactory results on the transitive gestures in a continuous motion using the proposed rules. In the future,a high-level semantic analysis will be applied to the current system to enhance the recognition capability for complex future works.The dataset consists of different gestures, With an extensive dataset containing 45,000 images, our technique achieves outstanding accuracy of 98.2 percent.

6.2 Comparison of Existing and Proposed System

Existing system:(Intelligent Sign Language Recognition Using Image Processing)

Computer recognition of sign language is an important research problem for enabling communication with hearing impaired people. This project introduces an efficient and fast algorithm for identification of the number of fingers opened in a gesture representing an alphabet of the Binary Sign Language. The system does not require the hand to be perfectly aligned to the camera. The project uses image processing system to identify, especially English alphabetic sign language used by the deaf people to communicate. The basic objective of this project is to develop a computer based intelligent system that will enable dumb people significantly to communicate with all other people using their natural hand gestures.The system is we are implementing for Binary sign language but it can detect any sign language with prior image processing

Proposed system:(Sign recognition using CNNs)

Our proposed system is sign language recognition system using convolution neural networks which recognizes various hand gestures by capturing video and converting it into frames. Then the hand pixels are segmented and the image it obtained and sent for comparison to the trained model. Thus our system is more robust in getting exact text labels of letters.

6.3 Sample Code

```python
def xml_to_csv(path):
    """Iterates through all .xml files (generated by labelImg) in a given directory and combines
    them in a single Pandas dataframe.

    Parameters:
    ----------
    path : str
        The path containing the .xml files
    Returns
    -------
    Pandas DataFrame
        The produced dataframe
    """

    xml_list = []
    for xml_file in glob.glob(path + '/*.xml'):
        tree = ET.parse(xml_file)
        root = tree.getroot()
        for member in root.findall('object'):
            value = (root.find('filename').text,
                     int(root.find('size')[0].text),
                     int(root.find('size')[1].text),
                     member[0].text,
                     int(member[4][0].text),
                     int(member[4][1].text),
                     int(member[4][2].text),
                     int(member[4][3].text)
                     )
            xml_list.append(value)
    column_name = ['filename', 'width', 'height',
                   'class', 'xmin', 'ymin', 'xmax', 'ymax']
    xml_df = pd.DataFrame(xml_list, columns=column_name)
    return xml_df


def class_text_to_int(row_label):
    return label_map_dict[row_label]


def split(df, group):
```

```
    data = namedtuple('data', ['filename', 'object'])
    gb = df.groupby(group)
    return [data(filename, gb.get_group(x)) for filename, x in zip(gb.groups.keys(), gb.groups)]


def create_tf_example(group, path):
    with tf.gfile.GFile(os.path.join(path, '{}'.format(group.filename)), 'rb') as fid:
        encoded_jpg = fid.read()
    encoded_jpg_io = io.BytesIO(encoded_jpg)
    image = Image.open(encoded_jpg_io)
    width, height = image.size

    filename = group.filename.encode('utf8')
    image_format = b'jpg'
    xmins = []
    xmaxs = []
    ymins = []
    ymaxs = []
    classes_text = []
    classes = []

    for index, row in group.object.iterrows():
        xmins.append(row['xmin'] / width)
        xmaxs.append(row['xmax'] / width)
        ymins.append(row['ymin'] / height)
        ymaxs.append(row['ymax'] / height)
        classes_text.append(row['class'].encode('utf8'))
        classes.append(class_text_to_int(row['class']))

    tf_example = tf.train.Example(features=tf.train.Features(feature={
        'image/height': dataset_util.int64_feature(height),
        'image/width': dataset_util.int64_feature(width),
        'image/filename': dataset_util.bytes_feature(filename),
        'image/source_id': dataset_util.bytes_feature(filename),
        'image/encoded': dataset_util.bytes_feature(encoded_jpg),
        'image/format': dataset_util.bytes_feature(image_format),
        'image/object/bbox/xmin': dataset_util.float_list_feature(xmins),
        'image/object/bbox/xmax': dataset_util.float_list_feature(xmaxs),
        'image/object/bbox/ymin': dataset_util.float_list_feature(ymins),
        'image/object/bbox/ymax': dataset_util.float_list_feature(ymaxs),
        'image/object/class/text': dataset_util.bytes_list_feature(classes_text),
        'image/object/class/label': dataset_util.int64_list_feature(classes),
    }))
    return tf_example


def main(_):

    writer = tf.python_io.TFRecordWriter(args.output_path)
    path = os.path.join(args.image_dir)
```

```
examples = xml_to_csv(args.xml_dir)
grouped = split(examples, 'filename')
for group in grouped:
    tf_example = create_tf_example(group, path)
    writer.write(tf_example.SerializeToString())
writer.close()
print('Successfully created the TFRecord file: {}'.format(args.output_path))
if args.csv_path is not None:
    examples.to_csv(args.csv_path, index=None)
    print('Successfully created the CSV file: {}'.format(args.csv_path))
```

Output

Figure 6.1: **Output 1**

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

Nowadays, applications need several kinds of images as sources of information for elucidation and analysis. Several features are to be extracted so as to perform various applications. When an image is transformed from one form to another such as digitizing, scanning, and communicating, storing, etc. degradation occurs. Therefore ,the output image has to undertake a process called image enhancement, which contains of a group of methods that seek to develop the visual presence of an image. Image enhancement is fundamentally enlightening the interpretability or awareness of information in images for human listeners and providing better input for other automatic image processing systems. Image then undergoes feature extraction using various methods to make the image more readable by the computer.Sign language recognition system is a powerful tool to preparae an expert knowledge, edge detect and the combination of inaccurate information from different sources. 0ntend of convolution neural network is to get the appropriate classification

7.2 Future Enhancements

The proposed sign language recognition system used to recognize sign language letters can be further extended to recognize gestures facial expressions. Instead of displaying letter labels it will be more appropriate to display sentences as more appropriate translation of language.This also increases readability.The scope of different sign languages can be increased. More training data can be added to detect the letter with more accuracy. This project can further be extended to convert the signs to speech.In the experiment, we conducted 4-fold cross-validation on the system where 600 and 200 images from a subject were used to train and test, respectively and the results showed that the average recognition rates of the seven gesture types were around 99 percent. To test the proposed method on multiple subjects, we trained and tested the hand images of the seven gesture types from seven subjects. The average recognition rate was 95.96 percent. The proposed system also had the satisfactory results on the transitive gestures in a continuous motion using the proposed rules. In the future,a high-level semantic analysis will be applied to the current system to enhance the recognition capability for complex future works.

Chapter 8

PLAGIARISM REPORT

Curiginal

Document Information

Analyzed document	minor_abi (1).pdf (D138287087)
Submitted	2022-05-28T10:40:00.0000000
Submitted by	Girirajan S
Submitter email	girirajancse@veltech.edu.in
Similarity	34%
Analysis address	girirajancse.veltec@analysis.urkund.com

Sources included in the report

SA	**Vel Tech Rangarajan Dr.Sagunthala R&D Inst. of S&T / Major_batch39 (9).pdf** Document Major_batch39 (9).pdf (D136824349) Submitted by: vvidhya@veltech.edu.in Receiver: vvidhya.veltec@analysis.urkund.com	3
SA	**Vel Tech Rangarajan Dr.Sagunthala R&D Inst. of S&T / major_project (5).pdf** Document major_project (5).pdf (D136133247) Submitted by: vvidhya@veltech.edu.in Receiver: vvidhya.veltec@analysis.urkund.com	3
SA	**Vel Tech Rangarajan Dr.Sagunthala R&D Inst. of S&T / SR-Major-Report.pdf** Document SR-Major-Report.pdf (D107551719) Submitted by: drthangamariappanl@veltech.edu.in Receiver: drthangamariappanl.veltec@analysis.urkund.com	2
SA	**Vel Tech Rangarajan Dr.Sagunthala R&D Inst. of S&T / Major_Project (2).pdf** Document Major_Project (2).pdf (D138133891) Submitted by: skarthika@veltech.edu.in Receiver: skarthika.veltec@analysis.urkund.com	1
SA	**FINAL REPORT MAJOR Project.pdf** Document FINAL REPORT MAJOR Project.pdf (D107623707)	17
SA	**Vel Tech Rangarajan Dr.Sagunthala R&D Inst. of S&T / Automatic-license plate detection-Major_Project_report (1).pdf** Document Automatic-license plate detection-Major_Project_report (1).pdf (D67779352) Submitted by: premak@veltech.edu.in Receiver: premak.veltec@analysis.urkund.com	1

Chapter 9

SOURCE CODE & POSTER PRESENTATION

9.1 Source Code

```
import os
import glob
import pandas as pd
import io
import xml.etree.ElementTree as ET
import argparse

os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'    # Suppress TensorFlow logging (1)
import tensorflow.compat.v1 as tf
from PIL import Image
from object_detection.utils import dataset_util, label_map_util
from collections import namedtuple

# Initiate argument parser
parser = argparse.ArgumentParser(
    description="Sample TensorFlow XML-to-TFRecord converter")
parser.add_argument("-x",
                    "--xml_dir",
                    help="Path to the folder where the input .xml files are stored.",
                    type=str)
parser.add_argument("-l",
                    "--labels_path",
                    help="Path to the labels (.pbtxt) file.", type=str)
parser.add_argument("-o",
                    "--output_path",
                    help="Path of output TFRecord (.record) file.", type=str)
parser.add_argument("-i",
                    "--image_dir",
                    help="Path to the folder where the input image files are stored. "
                         "Defaults to the same directory as XML_DIR.",
                    type=str, default=None)
parser.add_argument("-c",
                    "--csv_path",
                    help="Path of output .csv file. If none provided, then no file will be "
                         "written.",
```

```python
                     type=str, default=None)

args = parser.parse_args()

if args.image_dir is None:
    args.image_dir = args.xml_dir

label_map = label_map_util.load_labelmap(args.labels_path)
label_map_dict = label_map_util.get_label_map_dict(label_map)


def xml_to_csv(path):
    """Iterates through all .xml files (generated by labelImg) in a given directory and combines
    them in a single Pandas dataframe.

    Parameters:
    ----------
    path : str
        The path containing the .xml files
    Returns
    -------
    Pandas DataFrame
        The produced dataframe
    """

    xml_list = []
    for xml_file in glob.glob(path + '/*.xml'):
        tree = ET.parse(xml_file)
        root = tree.getroot()
        for member in root.findall('object'):
            value = (root.find('filename').text,
                     int(root.find('size')[0].text),
                     int(root.find('size')[1].text),
                     member[0].text,
                     int(member[4][0].text),
                     int(member[4][1].text),
                     int(member[4][2].text),
                     int(member[4][3].text)
                     )
            xml_list.append(value)
    column_name = ['filename', 'width', 'height',
                   'class', 'xmin', 'ymin', 'xmax', 'ymax']
    xml_df = pd.DataFrame(xml_list, columns=column_name)
    return xml_df


def class_text_to_int(row_label):
    return label_map_dict[row_label]


```

```python
def split(df, group):
    data = namedtuple('data', ['filename', 'object'])
    gb = df.groupby(group)
    return [data(filename, gb.get_group(x)) for filename, x in zip(gb.groups.keys(), gb.groups)]


def create_tf_example(group, path):
    with tf.gfile.GFile(os.path.join(path, '{}'.format(group.filename)), 'rb') as fid:
        encoded_jpg = fid.read()
    encoded_jpg_io = io.BytesIO(encoded_jpg)
    image = Image.open(encoded_jpg_io)
    width, height = image.size

    filename = group.filename.encode('utf8')
    image_format = b'jpg'
    xmins = []
    xmaxs = []
    ymins = []
    ymaxs = []
    classes_text = []
    classes = []

    for index, row in group.object.iterrows():
        xmins.append(row['xmin'] / width)
        xmaxs.append(row['xmax'] / width)
        ymins.append(row['ymin'] / height)
        ymaxs.append(row['ymax'] / height)
        classes_text.append(row['class'].encode('utf8'))
        classes.append(class_text_to_int(row['class']))

    tf_example = tf.train.Example(features=tf.train.Features(feature={
        'image/height': dataset_util.int64_feature(height),
        'image/width': dataset_util.int64_feature(width),
        'image/filename': dataset_util.bytes_feature(filename),
        'image/source_id': dataset_util.bytes_feature(filename),
        'image/encoded': dataset_util.bytes_feature(encoded_jpg),
        'image/format': dataset_util.bytes_feature(image_format),
        'image/object/bbox/xmin': dataset_util.float_list_feature(xmins),
        'image/object/bbox/xmax': dataset_util.float_list_feature(xmaxs),
        'image/object/bbox/ymin': dataset_util.float_list_feature(ymins),
        'image/object/bbox/ymax': dataset_util.float_list_feature(ymaxs),
        'image/object/class/text': dataset_util.bytes_list_feature(classes_text),
        'image/object/class/label': dataset_util.int64_list_feature(classes),
    }))
    return tf_example


def main(_):

    writer = tf.python_io.TFRecordWriter(args.output_path)
```

```
    path = os.path.join(args.image_dir)
    examples = xml_to_csv(args.xml_dir)
    grouped = split(examples, 'filename')
    for group in grouped:
        tf_example = create_tf_example(group, path)
        writer.write(tf_example.SerializeToString())
    writer.close()
    print('Successfully created the TFRecord file: {}'.format(args.output_path))
    if args.csv_path is not None:
        examples.to_csv(args.csv_path, index=None)
        print('Successfully created the CSV file: {}'.format(args.csv_path))


if __name__ == '__main__':
    tf.app.run()
```

9.2 Poster Presentation

References

[l] Ankit V Ponkia and Jitendra Chaudhari(2020), "*Face Recognition Using PCA Algorithm*", Inventi Rapid: Image Video Processing Journal, 4(1), pp. 519-524.

[2] B J Bipin, K Nihar and C. Adarsh(2020). "*A Comparative Binarization Approach for Degraded Agreement Document Image from Various Pharmacies*", IEEE journal, 12(4), pp. 806-814.

[3] K Suresh and V Pattabiraman(2021) "*An improved utility item sets mining with respect to positive and negative values using mathematical model*", International Journal of Pure and Applied Mathematics, 101(5), pp. 763772.

[4] MK J Khan, N Ud Din and B.S.Y. J.(2020), "*Interactive removal of microphone object in facial images*", Electronics, 8(10), pp.167-256.

[5]Xu, L. Ren, Jimmy Liu, C. Jia, J.. (2021). "*Deep convolutional neural network for image deconvolution Advances in Neural Information Processing Systems*", (6)2,pp: 1790-1798