A PROJECT REPORT ON SIGN LANGUAGE INTERPRETER

MINOR PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR

THE AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY

Information Technology

Submitted By:

Sukhmeet Singh(1905431)

Prikshat Kumar(1905426)

Sonu Pandit(1805917)

Submitted To:

Prof. Sidharath Jain

Assistant Professor

Minor Project Coordinator

Department of Information Technology

Guru Nanak Dev Engineering College,

Ludhiana-141006

Abstract

In this Sign Language Interpreter project, we have created a sign detector, which detects a number of hand signs from a live camera feed that can very easily be extended to cover a vast multitude of other signs and hand gestures including the alphabets. There have been several advancements in technology and a lot of research has been done to help the people who are deaf and dumb. Aiding the cause, Deep learning, and computer vision can be used too to make an impact on this cause. This can be very helpful for the deaf and dumb people in communicating with others as knowing sign language is not something that is common to all, moreover, this can be extended to creating automatic editors, where the person can easily write by just their hand gestures. Most research implementations for this task have used depth maps generated by depth cameras and high resolution images. The objective of this project was to see if we are able to identify hand signs using simple images of hands taken with a personal device such as a laptop webcam. This is in alignment with the motivation as this would make a future implementation of a real time Sign Language to oral/written language translator practical in an everyday situation.

ACKNOWLEDGEMENT

We are highly grateful to Dr. Sehajpal Singh, Principal, Guru Nanak Dev Engineering College (GNDEC), Ludhiana, for providing this opportunity to carry out the minor project work. The constant guidance and encouragement received from Prof. Sidharath Jain, IT Department, GNDEC Ludhiana has been of great help in carrying out the project work and is acknowledged with reverential thanks. We would like to express a deep sense of gratitude and thanks profusely to Prof. Sidharath Jain without her wise counsel and able guidance, it would have been impossible to complete the project in this manner. We express gratitude to other faculty members of the Information Technology Department of GNDEC for their intellectual support throughout the course of this work. Finally, we are indebted to all whosoever have contributed in this report work.

Sukhmeet Singh(1905431)

Prikshat Kumar(1905426)

Sonu Pandit(1805917)

Contents

1 Introduction

1.1 Introduction to Project Page

Sign language is a language through which communication is possible without the means of acoustic sounds. Instead, sign language relies on sign patterns, i.e., body language, orientation and movements of the arm to facilitate understanding between people. There are perhaps around two hundred sign languages in use around the world today. It has been estimated that there are between 0.9 and 14 million hearing impaired in India and perhaps "one of every five people who are deaf in the world, lives in India", making it the country with the largest number of Deaf, and perhaps also the largest number of sign language users.

1.2 Objectives

The main objective of this project was to build a model able to classify hand signs, given an image of a signing hand. This project is a first step towards building a possible sign language translator/interpreter, which can take communications in sign language and translate them into written language. Such a translator would greatly lower the barrier for many deaf and mute individuals to be able to better communicate with others in day to day interactions.

1.3 Identification/Reorganization of Need

Communication is one of the basic requirements for survival in society. Deaf and dumb people communicate among themselves using sign language but normal people find it difficult to understand their language. Our project aims at taking the basic step in bridging the communication gap between normal people and deaf and dumb people.

1.4 Existing System

Most research implementations for this task have used depth maps generated by depth camera and high resolution images which can consume a lot of processing power to work.

1.5 Proposed System

This project aims to train the model using images taken from a normal camera like laptop webcam and then detect hand signs using OpenCV on a normal windows machine.

2 Requirement Analysis and System Specification

2.1 Feasibility Study (Technical, Economical, Operational)

Depending on the results of the initial investigation the survey is now expanded to a more detailed feasibility study. âFEASIBILITY STUDYâ is a test of system proposal according to its work-ability, impact of the organization, ability to meet needs and effective use of the resources.

2.2 Software Requirement Specification Document which must include the following: (Data Requirement, Functional Requirement, Performance Requirement, Dependability Requirement, Maintainability requirement, Security Requirement, Look and feel requirement)

1. Python
2. OpenCV
3. Jupyter Notebook
4. Visual Studio
5. Tensorflow
6. LabelImg

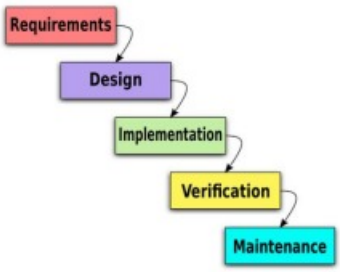
2.3 Expected hurdles

Sometimes the detection doesnât work, we have to restart the project to make it work.Accuray is low.Model takes too long to train.

2.4 SDLC Model to be used

We used a concept called the systems development life cycle (SDLC) to plan and manage the system development process.

The SDLC model contains the following steps: System Planning. System Analysis. System Design. System Implementation. System Operation, Support, and Security.



3 System Design

3.1 Design Approach

Function Oriented Design - Function Oriented design is a method of software design where the model is decomposed into a set of interacting units or modules where each unit or module has a clearly defined function. Thus, the system is designed from a functional viewpoint. We have divided our project into three modules - Image Processing Training model Object Detection

All the modules should be working in order for the Sign Recognition to work.

3.2 Data Processing

- First at least 15 images of each hand signs are collected, then they are labeled using label Img tool, they are divided into two parts - training and testing. We have to also create TF records. TF record is Tensor flowâs own binary storage format If you are working with large data sets, using a binary file format for storage of your data can have a significant impact on the performance of your import pipeline and as a consequence on the training time of your model.

3.3 Training

First we have to download Pretrained models from Tensorflow model Zoo. Then copy the model config to the training folder. We are using SSD Mobilenet because itâs fast, lightweight and can also detect multiple objects at once.Then we can train out model using the following command - python Tensorflow/models/research/object$_detection$
$-model_dir = Tensorflow/workspace/models/my_ssd_mobnet --pipeline_config_path = Tensorflow/workspace/model$
$-num_train_steps = 5000$

3.4 Classify Gesture

After a model has been trained, it can be used to detect hand signs from a live video camera feed using OpenCv and Python.

3.5 Methodology

Having collected the data set, we divided our approach to tackle the classification problem into three stages.
âc¢ The first stage is to segment the skin part from the image, as the remaining part can be regarded as noise w.r.t the character classification problem.
âc¢ The second stage is to extract relevant features from the skin segmented images which can prove significant for the next stage i.e learning and classification.
âc¢ The third stage as mentioned above is to use the extracted features as input into various supervised learning models for training and then finally use the trained models for classification.

4 Implementation, Testing, and Maintenance

4.1 Introduction to Languages, IDE's, Tools and Technologies used for Implementation

Python - Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Object Detection - Object detection can be defined as a branch of computer vision which deals with the localization and the identification of an object. Object localization and identification are two different tasks that are put together to achieve this singular goal of object detection. Object localization deals with specifying the location of an object in an image or a video stream, while object identification deals with assigning the object to a specific label, class, or description. With computer vision, developers can flexibly do things like embed surveillance tracking systems for security enhancement, real-time crop prediction, real-time disease identification/ tracking in the human cells, etc.

Computer Vision - Computer vision is a process by which we can understand the images and videos how they are stored and how we can manipulate and retrieve data from them.

OpenCV - OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in todayâs systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When it integrated with various libraries, such as Numpuy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

Tensorflow - TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art

in ML and developers easily build and deploy ML powered applications.

TensorFlow Object Detection API - Creating accurate machine learning models capable of localizing and identifying multiple objects in a single image remains a core challenge in computer vision. The TensorFlow Object Detection API is an open source framework built on top of TensorFlow that makes it easy to construct, train and deploy object detection models.

TensorFlow Model Zoo for Object Detection - The TensorFlow Model Zoo is a collection of pre-trained object detection architectures that have performed tremendously well on the COCO dataset. The model architectures include: 1. CenterNet 2. EfficientDet 3. MobileNet 4. ResNet 5. R-CNN 6. ExtremeNet

1. CenterNet (2019) is an object detection architecture based on a deep convolution neural network trained to detect each object as a triplet (rather than a pair) of keypoints, so as to improve both precision and recall.

2. EfficientDet (2019) is an object detection architecture built to scale up model efficiency in computer vision. This architecture achieves much better efficiency than prior architectures across a wide spectrum of resource constraints.

3. MobileNet is an object detector released in 2017 as an efficient CNN architecture designed for mobile and embedded vision application. This architecture uses proven depth-wise separable convolutions to build lightweight deep neural networks.

4. RetinaNet is an architecture developed by the Facebook research team in 2018. RetinaNet uses a Feature Pyramid Network (FPN) backbone on top of a feed-forward ResNet architecture to generate a rich, multi-scale convolutional feature pyramid. It is a one-staged detector (that is, a single network, unlike R-CNN, which is 2-staged).

5. R-CNN (2014) is a 2-stage object detection architecture. It is a region-based CNN that uses a Region Proposal Network to generate regions of interests in the first stage, and then sends the region proposal down the pipeline for object classification and bounding box regression.

6. ExtremeNet (2019) is a bottom-up object detection framework that detects four extreme points (top-most, left-most, bottom-most, right-most) of an object to find extreme points, by predicting four multi-peak heatmaps for each object category.

4.2 Coding standards of Language used (Python)

Use single quotes - Use single-quotes for string literals, e.g. 'my-identifier', but use double-quotes for strings that are likely to contain single-quote characters as part of the string itself (such as error messages, or any strings containing natural language), e.g. "You've got an error!".

Imports - Avoid creating circular imports by only importing modules more specialized than the one you are editing.

String formatting - Donât use the old Use the new .format() method instead, and give meaningful names

to each replacement field, for example:

$'(...foo...bar...').format(foo =' foo - value', bar =' bar - value')$

4.3 Testing Techniques and Test Plans

Functional vs. Non-functional Testing - In this testing we just test if our project is functional/running or not.

Unit Testing - Unit testing is the first level of testing and is often performed by the developers themselves. It is the process of ensuring individual components of a piece of software at the code level are functional and work as they were designed to. In our Project we test each step if it is working or not and remove any errors that occur.

Performance Testing - Performance testing is a non-functional testing technique used to determine how an application will behave under various conditions. In this we make sure that object detection is running smoothly and not lagging or overheating the machine.

Usability Testing - Usability testing is a testing method that measures an applicationâs ease-of-use from the end-user perspective.

Compatibility Testing - Compatibility testing is used to gauge how an application or piece of software will work in different environments. In this we make sure that our project is running on all environments like windows and linux, also on low end machines.

5 Results and Discussions

5.1 Brief Description of Various Modules of the system

Brief Description of Various Modules of the system - Data Processing - First at least 15 images of each hand signs are collected, then they are labeled using label Img tool, they are divided into two parts - training and testing. Training - First we have to download Pre trained models from Tensor flow model Zoo. Then copy the model config to the training folder. We are using SSD Mobile net because itâs fast, lightweight and can also detect multiple objects at once. Classify Gesture - After a model has been trained, it can be used to detect hand signs from a live video camera feed using OpenCv and Python.

5.2 Snapshots of system with brief detail of each

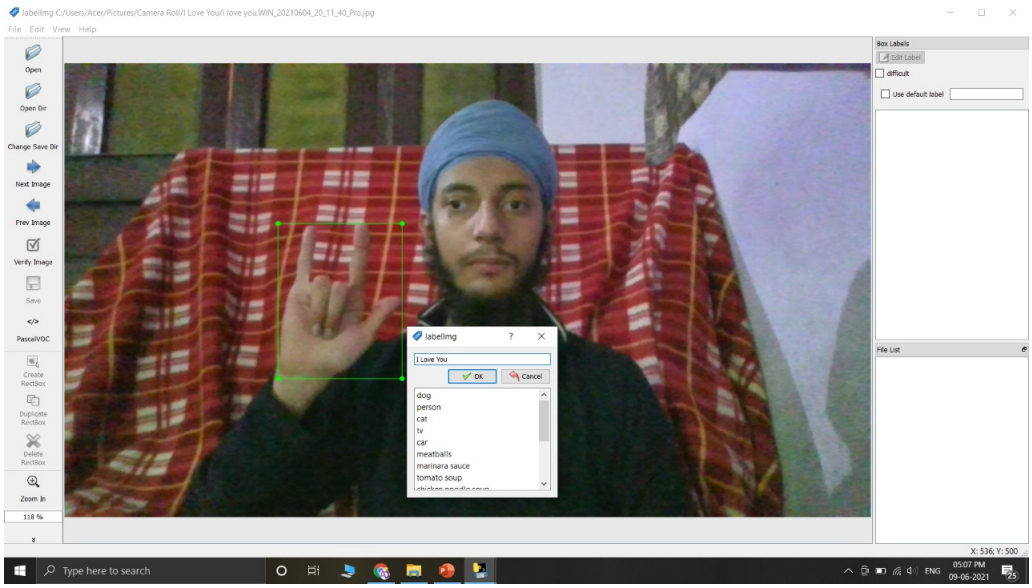


Figure 1: Prepare Images using Labelimg

```
I1105 12:11:02.423585  2492 cross_device_ops.py:443] Reduce to /job:localhost/replica:0/task:0/device:CPU:0 then broadcast to ('/job:localhost/r
plica:0/task:0/device:CPU:0',).
WARNING:tensorflow:From C:\ProgramData\Anaconda\lib\site-packages\tensorflow\python\util\deprecation.py:574: calling map_fn_v2 (from tensorflow.
ython.ops.map_fn) with dtype is deprecated and will be removed in a future version.
Instructions for updating:
Use fn_output_signature instead
W1105 12:11:07.136454 15600 deprecation.py:506] From C:\ProgramData\Anaconda\lib\site-packages\tensorflow\python\util\deprecation.py:574: callin
 map_fn_v2 (from tensorflow.python.ops.map_fn) with dtype is deprecated and will be removed in a future version.
Instructions for updating:
Use fn_output_signature instead
INFO:tensorflow:Step 100 per-step time 0.129s loss=0.752
I1105 12:11:29.458861  2492 model_lib_v2.py:652] Step 100 per-step time 0.129s loss=0.752
```

Figure 2: Training the object detection model

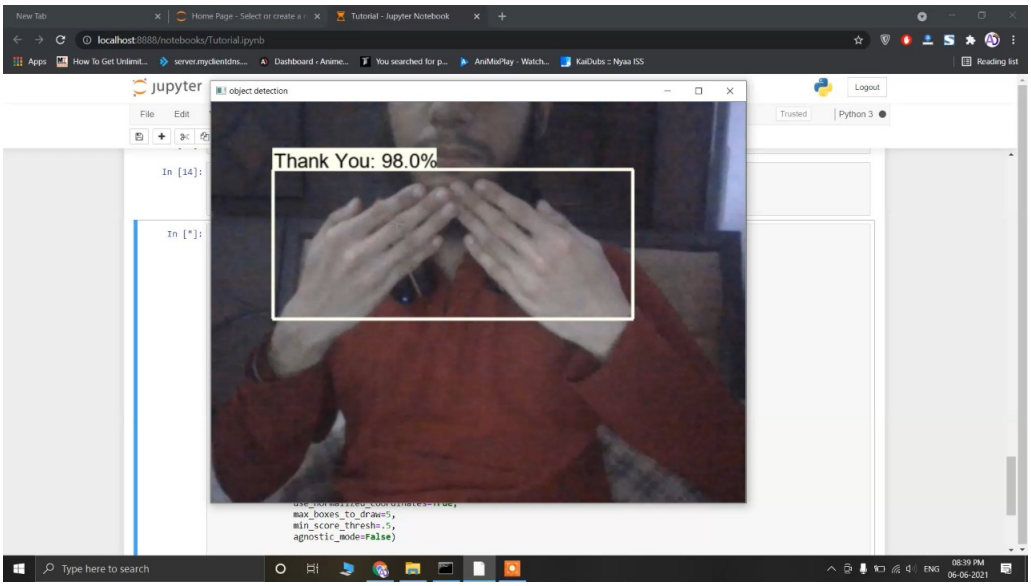


Figure 3: Real time detection using opencv and webcam

6 Conclusion

We conclude that Tensorflow, SSD Mobile net, OpenCv and python can be used to train a model and detect hand signs in real time from live video camera feed. Itâs lightweight and fast but accuracy can be quite low and training the model can take a lot of time, so thereâs still room for improvement.

7 Future Scope

Create a front end for the project, windows or mobile application. Add more hand signs. Increase accuracy of the detection.

References/Bibliography

https://opencv.org/

https://www.tensorflow.org/

https://github.com/tensorflow/models/tree/master/research/object$_detection$

$https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md$

$https://heartbeat.fritz.ai/real-time-object-detection-using-ssd-mobilenet-v2-on-video-streams-3bfc1577399c$