Parichay Maharaja Surajmal Institute Journal of Applied Research

ML-Based Hand Sign Detection System for Deaf-Mute People

Dr. Neetu Narwal*[1], Ayush Tanwar[2]

ABSTRACT

Communication is imperative to human existence and pervasive in all aspects of our lives. It is also an important right to everyone. This paper focuses on solving the communication problem with the deaf and mute population of the world. This is done using real-time machine learning (ML) for sign language detection along with certain APIs (Application Programming Interfaces). The discussed system detects the signs in the region of interest (ROI) and then converts them into the appropriate format. Computer vision has been used for detection and API function for many other functionalities. This idea here is focused on reducing the communication gap within the deaf-mute population and bringing a sense of normal in their daily life.

KEYWORDS: Hand Signs, Computer Vision, Object Detection, Machine Learning, Transfer Learning, Cloud Bucket

INTRODUCTION

Communication is a way of exchanging information, opinions, and feelings among individuals in order to reach a common understanding. People with disabilities may require assistance due to complex communication needs, but they are still left behind in communicating their ideas to the general public to some extent because the majority of people are not educated enough to understand their first language, resulting in one-way communication. It's easy to be scared by the thought of communicating with someone who has a disability, particularly if you're not sure what to say or how to say it. As a result, we hope to use artificial intelligence and other technologies in our project to bridge these gaps and bring everyone, special or not, on the same level, resulting in a better world. For the general public also this application will be a very useful platform as it will completely demolish the different prerequisites that were required earlier to communicate to people with different special needs.

The transmitter and the receiver are the two basic components of any communication. The sender expresses a sentiment or emotion, seeks information, or transmits an idea or notion, and the recipient receives the message. To summarize, each communication entails a sender and a receiver, a message, and both sides' interpretations of the message's meaning [1]. We interact with individuals on a daily basis and it's crucial to remember to treat each person with a disability as an individual while talking with them. To communicate responsibly and respectfully with and about an individual, people-first language is utilized which emphasizes the person first, not the disability.

When it is impossible or undesirable to speak orally, sign language is any form of communication involving physical movements, mainly hand gestures. It is possible that the practice precedes speech. Sign language can be as crude as frowns, shoulder shrugs, or gesticulation or it can be a fine combination of manually coded signals complemented by face expressions or spelled out words. When voice communication is impossible, such as between speakers of mutually incomprehensible languages or when one or more potential communicators are deaf, sign language can be employed to bridge the gap. The deaf and hard-of-hearing community uses sign language as their primary mode of communication, but it can also be useful for other groups.

The BSL (British Sign Language) alphabet is fingerspelling. There is a symbol for each letter of the alphabet. On your hand, you can use these letter signs to spell out words – most common names and places – and sentences. If you don't know or can't remember some BSL signs, you can use fingerspelling to communicate.

With the help of deep learning method, computer vision, we plan to read the hand sign of the user and translate it to the appropriate format. For example, a deaf person who wants to communicate with a person with no disability can do so using our website in which we read the hand sign of the person with a disability and convert it to audio for the other person to hear.

The key features implemented in this paper are:

- Taking video inputs of sign language used by differently abled people.
- With the use of artificial intelligence, convert these video inputs of various signs used by the user, interpret them to their corresponding speech and text, making it more comprehensible for the respective auditor.

[1] *Associate Professor, Department of Computer Applications, Maharaja Surajmal Institute, Affiliated to Guru Gobind Singh Indraprastha University, New Delhi*

[2] *Student, Department of Computer Applications, Maharaja Surajmal Institute, Affiliated to Guru Gobind Singh Indraprastha University New Delhi*

- Personalize the application for each user as per their preference, if needed.

BACKGROUND STUDY

OBJECT DETECTION

Object detection is a method of identifying objects and locating them in images or videos using computer vision. It draws boundary boxes around the objects that are identified in the image or video so we can see their position and movement across a scene. Object detection is different from picture recognition. Image recognition is used to label a picture. In a photograph of a dog, the term "dog" is used. A photograph of two dogs still bears the label "dog." On the other hand, object detection draws a box around each dog with the word "dog" written on it. The model predicts the location of each object as well as the label that should be attached to it [2].

OPENCV

OpenCV is a free and open-source computer vision and machine learning library, developed to provide a general infrastructure for computer vision based applications. A wide-range of optimized algorithms are included that cover machine learning techniques and integration of machine perception. These can be used for detecting or identifying objects, recognizing faces or classifying actions in videos, tracking and following movements, set markers or even extract 3D models of objects and stitch images together to create a high-resolution image of an entire scene. There are several key domains involved with computer vision, including image processing, video capture and analysis, face detection, and object detection, but developing real-time applications requires a cross-platform library. This is where OpenCV, a C++ based program that was later followed by a Java-based version, comes in [3].

TENSORFLOW

TensorFlow is an open-source machine learning platform that may be utilized from start to end. It's a symbolic math toolbox for deep neural network training and inference that solves a wide range of issues using data flow and differentiable programming. It allows programmers to develop machine learning applications using several tools, libraries, and open-source resources [4]. By receiving inputs as a multidimensional array (referred to as Tensor), TensorFlow allows you to make structures that govern the flow of data through a graph. The input enters at one end, passes through the complex actions of preprocessing, creating model and its training and estimation, and exits at the other end as output. [10].

SINGLE SHOT DETECTOR (SSD)

The SSD (Single Shot Detection) technique is used for object-detection in real-time. The SSD architecture comprises a single convolutional neural network (CNN) which learns to predict and classify the bounding box positions for object detection in a single pass. Therefore, it can be trained from start to the end. MobileNet, the basic SSD network architecture used in this work consists of numerous convolution layers. It implements multi-scale features and other enhancements to improve accuracy while utilizing images of lower resolution, and thus further increasing the speed [8] [9].

TENSORFLOW DETECTION MODEL ZOO

Similar to Facebook's Detectron2 computer vision library, the TensorFlow Object Detection API model zoo includes a wide range of object detection models that you can deploy to your custom dataset and build from.

The TensorFlow Object Detection API lets you to quickly try out novel architectures in the TensorFlow ecosystem and provides deployment solutions for developers by offering model export scripts to .pb protobuf files that contain the inference graph description. These models can then be exported to venues like TF Lite or TFJS [5].

These models are evaluated on the basis of speed per step, mAP (Mean Average Precision), and even frame style.

For each class, one can calculate the

- True Positive TP(c): There was a proposal for class c, and there was a class c object.
- False Positive FP(c): Class c has been proposed, yet there is no class c object.
- Average Precision for class c, as shown through equation 1.

$$\frac{\#TP(c)}{\#TP(c)+\#FP(c)} \quad (1)$$

Then mAP is calculated as shown in equation 2 [6]:

$$mAP = \frac{1}{|classes|} \sum_{c \in classes} \frac{\#TP(c)}{\#TP(c) + \#FP(c)} \quad (2)$$

CONCEPT OF TRANSFER LEARNING

Transfer learning is a machine learning technique in which a model developed for one task is used as the foundation for another task's model. It is a common deep learning approach that employs pre-trained models. Given the massive compute and temporal resources required, as a starting point for research into computer vision and natural language processing challenges construct neural network models for these issues, and from the massive amount of data leaps in a skill that they offer in connection with a problem. A pre-trained model is a previously trained network, typically on a huge dataset, preserved on a large-scale picture classification assignment. This pre-trained model can be used as it is or transfer learning can be used to adapt it to a specific task. For image classification, transfer learning involves training the model on a general huge dataset such that it may serve as a generic model of the visual world. One is then able to use these feature maps that they have learned without having to re-learn them [7].

WORKING AND IMPLEMENTATION

SYSTEM DESIGN

In this work, we will be using the concept of transfer learning to train our model to reduce the training time. We are going to extract a model from the tensor flow model zoo for the same.

First, we will collect images to create a dataset of our own. For this, we made an automated system to capture the images. These images are then labeled using a library called *labelImg*. Labeling each image creates an XML file that contains data about the image such as its label name, height, width, dimensions on the x-y axis, etc. After this, all the images with their respective XML files are divided into two segments where ¾ images go to the training folder and the rest to the testing folder. This creates data for the model to train itself and then test itself. Then we provide the data and the training process begins, where we use SSD Mobilenet V2 FPNLite 320x320 to train our model. After a successful training session, the model is tested for efficiency and after receiving satisfactory results, the model is frozen and converted into TensorFlowJS format to be stored inside a cloud bucket to be accessible anywhere.

This bucket is then accessed in our web app which takes video inputs and then provides them to the model which then detects and provides the output in form of text. This text is accessed by a text to speech API which also provides the audio outputs simultaneously. Figure 1 represents the architecture we discussed.

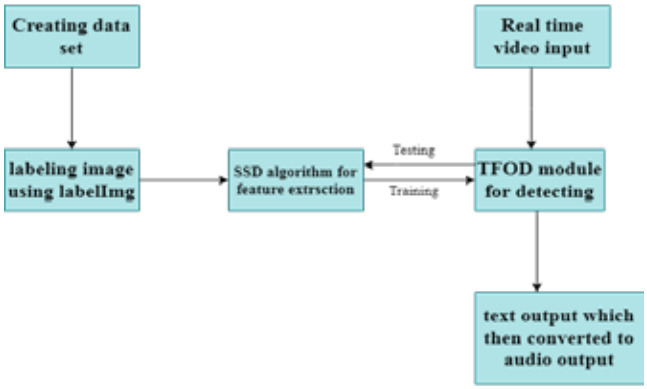

Figure 1 : System Architecture

DATASET CREATION

The LabelImg software is used to graphically label the images, which is then utilized to recognize the images. We must remember that labeling must be done correctly, i.e., the gesture must be labeled with the correct label in order for the movements to be recognized correctly later with the correct label. After the photographs have been labeled and saved, an XML file is created for each one. During the training phase, this XML file contains information on where the model should look in the image. Figure 2 shows the file structure of one such XML file.

Figure 2 : Label data for "Hello"

TRAINING AND TESTING

As discussed, we used ¾ of our collected images along with their XML files to train the model and the rest to test the model.

For the training, we used the SSD Mobilenet V2 FPNLite 320x320 model. It had a speed of 19ms per step and mAP of 20.5. For testing, we used the TensorFlow object detection API.

Figure 3 shows the learning rate at 10,000 steps.

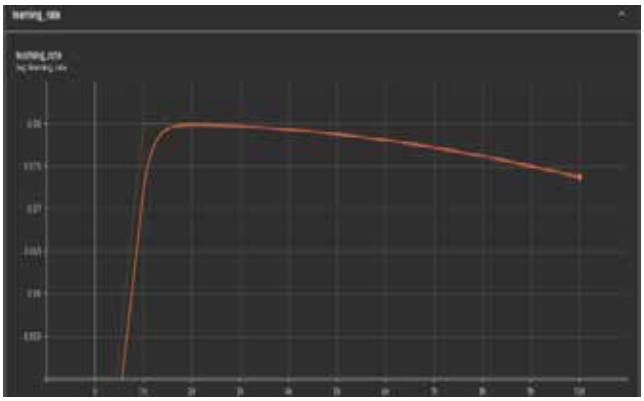

Figure 3 : Learning rate at 10,000 steps

Loss function is used to optimize the ML algorithm. The loss is estimated based on the model's performance in both training and testing, and its interpretation is based on the model's performance in both training and testing sets, i.e. the total number of errors committed in each. The value of this loss function specifies the mode's performance after every optimization iteration. Our machine learning model's loss has been lowering with each iteration, indicating that the model's detection accuracy has improved. Figure 4 shows the loss of the model.

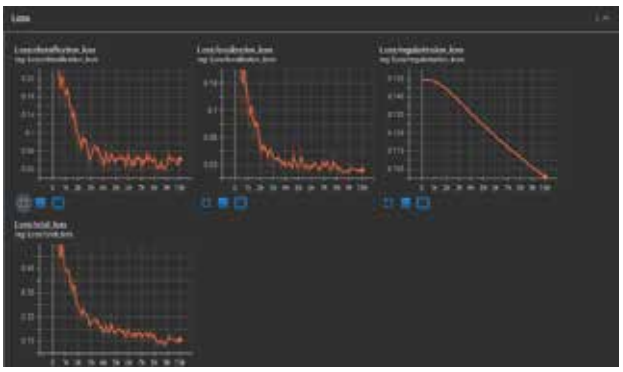

Figure 4 : Loss graph on various parameters

The mismatch between the ground truth box and the anticipated boundary box is known as the localization loss. Only forecasts with positive matches are penalized by SSD. To get closer to the ground reality, only the predictions from the good matches are required. Negative matches aren't worth paying attention to. The predicted boundary box is the box that is predicted by the model while testing the photos, while the ground truth box is the box that is constructed in the LabelImg software while making the labels. Our model has a localization loss of 0.05.

STORING THE MODEL IN CLOUD BUCKET AND WEBSITE INTEGRATION

After a successful training session, we freeze the model and store the model in a cloud bucket. We used IBM cloud storage bucket for this project. Then we extract the bucket link and integrate it with our web application. The web app is specially designed to make the user experience easy and smooth. The web app also has a text to speech API that converts the text results that we receive from the model and converts them to audio. Figure 5 shows a working example of one such gesture i.e. "hello". The output text also displays the accuracy percentage, like in Fig.3 the accuracy is 95%.

Figure 5 : Sign detection from the web app

FINDINGS

After evaluation, the precision rate of the model came out to be 69% and the recall rate came out to be 70%, which is not a great result but still it's a start. And also it was observed that the increase in the number of images and adding a more and more different variety of data from various angles also increased the stats significantly. Even after that, as seen in figure 5, the detection is being performed with a high accuracy rate.

CONCLUSION

In this age where nothing seems to be impossible, leaving a section of our society out just because of their physical disability is unfair to them. Thus our project is a small step to bridge this gap. Although we have a lot of bases to cover, still this will hopefully start a process that might go on further ahead. With the use of object detection, we have made the model very easy to recreate and expand or improve wherever required.

FUTURE SCOPE

We are still capturing still images, but in the future, we can train our model to capture more complex live hand signs that require more than one gesture. Also, we plan to add many other features such as performing certain daily life actions using hand gestures or voice commands. And also this can be also incorporated into a mobile application which would make it easier to use.

ACKNOWLEDGEMENTS

We are very thankful to the head of our department and the institute and its management, for giving us the inspiration, training, and tools to carry out the research work. Despite the constant spread and emergence of covid-19 pandemic, it was possible to continue with the research work and even request for online suggestions at any time.

REFERENCES

[1]. Nordquist, R. (2019). Definition and Examples of Senders in Communication. ThoughtCo. Available at https://www.thoughtco.com/sender-communication-1691943

[2]. What is Object detection? Mathworks. Available at https://www.mathworks.com/discovery/object-detection.html

[3]. OpenCV documentation. OpenCV Team. Available at https://opencv.org/about/

[4]. Tensorflow documentation. Available at https://www.tensorflow.org/

[5]. Solawetz, J. (2020). The TensorFlow 2 Object Detection Library is Here. Roboflow. Available at https://blog.roboflow.com/the-tensorflow2-object-detection-library-is-here/

[6]. What is mAP metric and how is it calculated? Stackoverflow. Available at https://stackoverflow.com/questions/36274638/what-is-the-map-metric-and-how-is-it-calculated

[7]. Transfer Learning. Wikipedia. Available at https://en.wikipedia.org/wiki/Transfer_learning

[8]. Rastgoo, R., Kiani, K., & Escalera, S. (2020). Video-based isolated hand sign language recognition using a deep cascaded model. Multimedia Tools and Applications, 79, 22965-22987.

[9]. Sanmitra, P. R., Sowmya, V. S., & Lalithanjana, K. (2021). Machine Learning Based Real Time Sign Language Detection. International Journal of Research in Engineering, Science and Management, 4(6), 137-141.

[10]. Arcos-Garcia, A., Alvarez-Garcia, J. A., & Soria-Morillo, L. M. (2018). Evaluation of deep neural networks for traffic sign detection systems. Neurocomputing, 316, 332-344.