ABSTRACT

Sign language is the means of communication for hearing-impaired people. There is a challenge for ordinary people to communicate with deaf people, which makes this system helpful in assisting them.

This project aims at implementing computer vision, which can take the signs from the users and convert them into text in real time.

The proposed system contains four modules such as: image capturing, preprocessing, classification, and prediction. By using image processing, the segmentation can be done. Sign gestures are captured and processed using the OpenCV python library. The captured gesture is resized, converted to a greyscale image, and the noise is filtered to achieve prediction with high accuracy. The classification and predication are done using a convolution neural network (CNN).

The converted real-time text appears in the text box. Generated text can be edited if necessary, and there is an option to save it on a clipboard, which can be pasted on any application we want.

The main features of this project are:

- It allows for all people who depend on sign language for effective communication.
- It helps to learn sign language for those who wish to learn sign language.
- It provides accurate sign language conversions in real-time.
- The converted text can be copied to any application.

REFERENCES

Sign Language Translation
Published in: 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)
Conducted by: IEEE
Date of Conference: 06-07 March 2020
Conference Location: Coimbatore, India

CHAPTER 1
INTRODUCTION

1.1 GENERAL INTRODUCTION

The most recent WHO estimate suggests that approximately 466 million people (or 6.1% of the world's population) were living with disabling hearing loss in 2018. This estimate is projected to rise to 630 million by 2030 and to over 900 million by 2050. In India, there are approximately 63 million people who are suffering from significant hearing impairment; this places the estimated prevalence at 6.3% of the Indian population.

This proposed system is a python application using packages like Tkinter, OpenCV, and TensorFlow. Where sign language is captured in real-time from the end user and these signs are evaluated through a CNN deep learning neural network, which was trained using machine learning mechanisms to predict each word and appear in the text field as output, which can be used for various applications.

1.2 GOAL OF THE PROJECT

The main means of communication for deaf people are through text and sign language. So, I am trying to demonstrate how effective it is when we integrate these sources of communication by making a sign-to-text transcript, which is beneficial to deaf people because it allows conversation between people and even modern technologies like voice assistants through sign language regardless of whether or not they are deaf, with the help of computer vision and deep learning CNN model.

Here I am using American Sign Language (ASL) since it is one of the most widely used sign languages in the world.

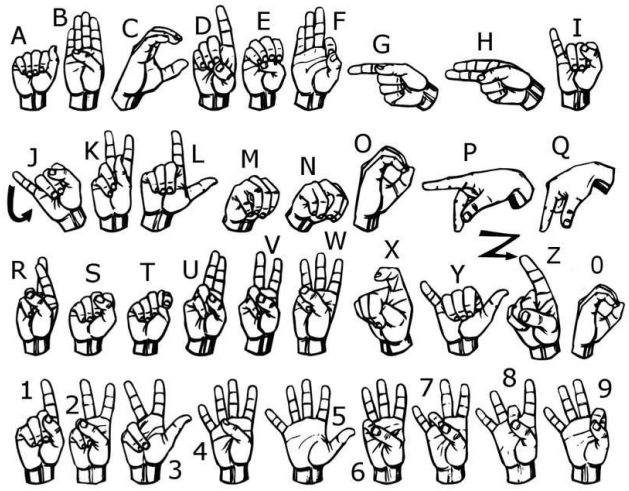

Figure 1: American Sign Language (ASL)

CHAPTER 2
LITERATURE SURVEY

2.1 STUDY OF SIMILAR WORK

There are many types of advanced and modern transcription applications that are available today on various platforms like desktops, smartphones, etc., web applications, hardware devices, or even as an embedded future on various applications like voice assistant systems.

The main leader in this field is Google Translate, which has lots of options and features like selecting translations according to the end user, predicting the language, and translating it through both voice and text transcript.

But there is a lack of a sign to corresponding text translation system even though there are lots of people dependent on it and different sign languages like American Sign Language (ASL), British Australian and New Zealand Sign Language (BANZSL), Indian Sign Language(ISL), etc. are some of the sign languages commonly used.

2.1.1 EXISTING SYSTEM

Existing Sign to text transcript systems are commonly available as prototypes that can't be commercially used with minimal and sometimes no user interface and we can't extract any resulting text for other applications just because it labels on the display view itself.
These Existing systems need to be manually trained whenever we try to use this system and it is not feasible for common users with minimal or no programming skills.
Lack of providing training or adding new training data to the existing system is a concern but there is a risk factor of overfitting and miss judge the outliers on the system which can affect the accuracy. So, research on overcoming this problem is just evolving now may be in the future there will be an optimal solution for this.

2.1.2 DRAWBACK OF EXISTING SYSTEM

- Lack of a good Graphical User interface
- Only prototypes are available
- Absence of full-fledged application
- Lack of availability for common people
- Absence of this feature as an integrated unit in modern technologies
- Language support is minimal
- Dependent on external environment for sign capture
- Not produce accurate results on some existing system according to external environments
- Some existing system is purely dependent on training data and it always need to be trained manually

CHAPTER 3
OVERALL DESCRIPTION

3.1 PROPOSED SYSTEM

The research paper entitled "Sign Language Translation" (2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)) conducted by IEEE at Coimbatore, India on March 06-07, 2020) presents a good general idea of how to implement one of the methods, i.e., by using live sign image capture, reducing background noise, evaluating using a conventional neural network(CNN) and generating results, which is one of the basic and easy ways to implement on various platforms.

By inspiring this paper, which basically focuses on implementation, I am trying to implement how this system can be commercialised to the end user as a fully-fledged application which can be accessed by ordinary people.

The process complexity of this proposed system is hidden from the end user. Only the required transcript text will appear as the output.

3.2 FEATURES OF PROPOSED SYSTEM

The aim of the proposed system is to translate American Sign Language (ASL) to text by using CNN and deep learning algorithms with the help of Tensorflow and Keras. Apart from that It shows real-time identification of which character is provided according to text on display itself as a label using OpenCV and this character forms into a text message on the text field provided. This text field can be further edited and can be copied using the copy to clipboard option using TKinter provided in this application itself, which can be pasted on various applications.

3.3 FUNCTIONS OF PROPOSED SYSTEM

There are basically four main functions for this proposed system:

1. **Image capturing**
   Python OpenCV library can be used to capture sign gestures from computer's internal camera. The dataset for various signs are collected. To predict gestures with high accuracy, around 25 images are collected for each sign.

2. **Pre-processing**
   The primary focus of the system is to support detecting gestures in dynamic background condition. To achieve this, the frames are pre-processed and converted to gray scale image and then background subtraction algorithm is used using Gaussian blur to reduce noise. The camera first captures around 10 frames to identify the background and compares current frame with previous frame. If a hand is detected, then the background is subtracted and only the hand gesture is converted to gray scale and transfers to model for classification and prediction.

3. **Classification**
   After collecting and processing the image dataset, they have to be classified. Convolutional neural network is used to analyse and classify visual imagery. It is

widely applied in image recognition and classification, natural language processing, etc. CNN is regularized versions of multilayer perceptron.

4. **Prediction**
   The CNN model evaluates the real time input from end user and predicts the output.

3.4 REQUIREMENTS SPECIFICATION

1. **Accuracy**
   The proposed system should be accurate on generating results based on given inputs.

2. **Speed**
   The proposed system should be in real time for generating results.

3. **Flexible**
   The proposed system should be flexible to new updates and patches in near future.

4. **Good Interface**
   The proposed system should maintain good interface even after upgradations.

3.5 FEASIBILITY STUDY

Feasibility Study in Software Engineering is a study to evaluate feasibility of proposed project or system. Feasibility study is one of stage among important four stages of Software Project Management Process. As name suggests feasibility study is the feasibility analysis or it is a measure of the software product in terms of how much beneficial product development will be for the organization in a practical point of view. Feasibility study is carried out based on many purposes to analyse whether software product will be right in terms of development, implantation, contribution of project to the organization etc.
In our proposed system the product is feasibility can be achieved in all four aspects Technical Operational, Economical and Behavioural.

3.5.1 TECHNICAL FEASIBILITY

In Technical Feasibility current resources both hardware software along with required technology are analysed/assessed to develop project. This technical feasibility study gives report whether there exists correct required resources and technologies which will be used for project development. Along with this, feasibility study also analyses technical skills and capabilities of technical team, existing technology can be used or not, maintenance and up-gradation is easy or not for chosen technology etc. In this proposed system technical feasibility is achieved according to above criteria.

3.5.2 OPERATIONAL FEASIBILITY

In Operational Feasibility degree of providing service to requirements is analysed along with how much easy product will be to operate and maintenance after deployment. Along with this other operational scopes are determining usability of product, determining suggested

solution by software development team is acceptable or not etc. The Operational feasibility can be ensured by the proposed system.

3.5.3 ECONOMICAL FEASIBILITY

Economic feasibility the most important and frequently used method for evaluating the effectiveness of the proposed system. It is very essential because the main goal of the proposed system is to have economically better results along with increased efficiency. Cost benefit analysis is usually performed for the expected from the proposed system. Since the organization is well equipped with the required hardware, the project was found to be economically feasible and the users who possess a device supports Windows operating system can easily use it.

3.5.4 BEHAVIORAL FEASIBILITY

The proposed system satisfies behavioral feasibility because the system is providing with good and minimalistic GUI which can easily be understand for any end users and its encapsulates the conversion procedure from the users. Hence it's easier to operate the system with ease.

CHAPTER 4

OPERATING ENVIRONMENT

4.1 HARDWARE REQUIREMENTS

1. Processor: Dual Core 1.60 GHz or higher
2. Hard disk: 500 GB
3. RAM: 4GB
4. Monitor: 17" Color Monitor
5. Mouse: Microsoft
6. Keyboard: Microsoft multimedia keyboard

4.2 SOFTWARE REQUIREMENTS

1. Operating System: Windows 8.1 Pro or higher
2. Framework: Microsoft .Net Framework
3. Environment: PyCharm Community Edition 2022.2
4. Language: Python 3.9, Open CV, and Tensor Flow
5. Documentation: Microsoft Word 2010 or higher

4.3 TOOLS AND PLATFORMS

4.3.1 PYCHARM

PyCharm is a dedicated Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers, tightly integrated to create a convenient environment for productive Python, web, and data science development.

4.3.2 PYTHON 3.9

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

4.3.3 TKINTER

The tkinter package ("Tk interface") is the standard Python interface to the Tcl/Tk GUI toolkit. Both Tk and tkinter are available on most Unix platforms, including macOS, as well as on Windows systems.
Running python -m tkinter from the command line should open a window demonstrating a simple Tk interface, letting you know that tkinter is properly installed on your system, and also showing what version of Tcl/Tk is installed, so you can read the Tcl/Tk documentation specific to that version.
Tkinter supports a range of Tcl/Tk versions, built either with or without thread support. The official Python binary release bundles Tcl/Tk 8.6 threaded.

Tkinter is not a thin wrapper, but adds a fair amount of its own logic to make the experience more pythonic

4.3.4 OPEN CV

Open CV (*Open Source Computer Vision Library*) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel). The library is cross-platform and free for use under the open-source Apache 2 License. Starting with 2011, OpenCV features GPU acceleration for real-time operations.

4.3.5 DEEP LEARNING

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example. Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost. It is the key to voice control in consumer devices like phones, tablets, TVs, and hands-free speakers. Deep learning is getting lots of attention lately and for good reason. It's achieving results that were not possible before.
In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. Models are trained by using a large set of labeled data and neural network architectures that contain many layers.

4.3.6 CNN

In deep learning, a **convolutional neural network** (**CNN/ConvNet**) is a class of deep neural networks, most commonly applied to analyze visual imagery. Now when we think of a neural network we think about matrix multiplications but that is not the case with ConvNet. It uses a special technique called Convolution. Now in mathematics **convolution** is a mathematical operation on two functions that produces a third function that expresses how the shape of one is modified by the other.

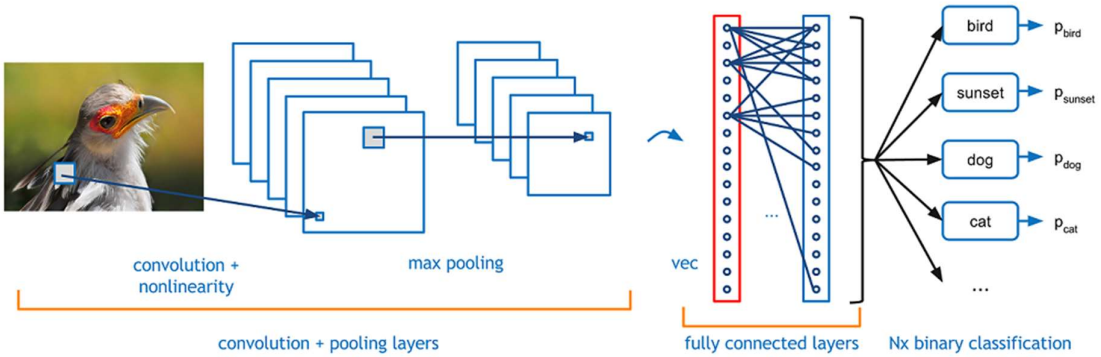

Figure 2: CNN

Convolutional neural networks are composed of multiple layers of artificial neurons. Artificial neurons, a rough imitation of their biological counterparts, are mathematical functions that calculate the weighted sum of multiple inputs and outputs an activation value. When you input an image in a ConvNet, each layer generates several activation functions that are passed on to the next layer.
The first layer usually extracts basic features such as horizontal or diagonal edges. This output is passed on to the next layer which detects more complex features such as corners or combinational edges. As we move deeper into the network it can identify even more complex features such as objects, faces, etc.

4.3.7 TENSORFLOW 2 AND KERAS

TensorFlow 2 is an end-to-end, open-source machine learning platform. You can think of it as an infrastructure layer for differentiable programming. It combines four key abilities:

- Efficiently executing low-level tensor operations on CPU, GPU, or TPU.
- Computing the gradient of arbitrary differentiable expressions.
- Scaling computation to many devices, such as clusters of hundreds of GPUs.
- Exporting programs ("graphs") to external runtimes such as servers, browsers, mobile and embedded devices.

Keras is the high-level API of TensorFlow 2: an approachable, highly-productive interface for solving machine learning problems, with a focus on modern deep learning. It provides essential abstractions and building blocks for developing and shipping machine learning solutions with high iteration velocity.

Keras empowers engineers and researchers to take full advantage of the scalability and cross-platform capabilities of TensorFlow 2: you can run Keras on TPU or on large clusters of GPUs, and you can export your Keras models to run in the browser or on a mobile device.