SIGN LANGUAGE DETECTION USING MACHINE LEARNING

Bhanu Prakash Vasina[1], Umar Janu Shaik[2], Sukesh Reddy Vinnamala[3], Sadhik Basha Shaik[4]
Mastanbi Shaik[5]
[1-4]*Undergraduate Students, Department of ECE, Vasireddy Venkatadri Institute of Technology, Nambur, Guntur, Andhra Pradesh, India*
[5]*Assistant Professor, Department of ECE, Vasireddy Venkatadri Institute of Technology, Nambur, Guntur, Andhra Pradesh, India*

ABSTRACT

Sign language serves as a crucial mode of communication for the Deaf and Hard of Hearing community. However, bridging the communication gap between sign language users and those unfamiliar with it remains a significant challenge. This final year project focuses on the development and implementation of a machine learning-based sign language detection system. The project aims to create a robust and real-time system capable of recognizing and translating sign language gestures into text or speech. Leveraging machine learning techniques, including computer vision and deep learning. The primary objectives of the project include data collection, model development, and user interface design. Data collection involves gathering a comprehensive dataset of sign language gestures performed by a diverse group of individuals to ensure the system's adaptability and inclusivity.

Keyword: - *Machine Learning, Sign Language Detection, and User-friendly interface*

1. INTRODUCTION

A key element of human reach is communication, that may encompass numerous shapes, including speech, writing, or visual expression. Text and visual clues are frequently used in communication amongst deaf and mute adults. Secure and intimate delivery between thoughts and emotions is made much easier by gestural communication, particularly when compared to hand and face expressions. The primary method of communication for the deaf and mute is sign language, that utilizes an intricate framework of gestures and movements. On the opposing hand, this makes communication challenging between those who are difficult to hearing and individuals who are not literate in sign language.

For communication to be made simpler and more inclusive, the barrier between the hearing impaired and the public at large must be eliminated. A sign language recognition system addresses this challenge by taking input expressions from hearing-impaired individuals and converting them into text or voice output for those who do not understand sign language. By recognizing and interpreting sign language gestures, this system enables more seamless and effective communication between hearing-impaired individuals and the rest of society. This technology not only enhances communication but also promotes inclusivity and understanding, fostering a more connected and accessible world for all.

2. EXISTING SYSTEM

In real-time applications, the K-Nearest Neighbors (KNN) method provides a rapid and simple solution for evaluating tiny or well-structured datasets with continuous characteristics, making it perfect for tasks such as gesture detection. For example, in a classroom, a system may be deployed to analyze live video feeds, extracting and interpreting hand signals to improve communication. By processing the video progressively, including motion detection and color conversion to HSV format, the system can decrease noise and focus on important details. The KNN algorithm then uses these extracted parameters to identify and interpret hand gestures, allowing for real-time comprehension of the user's intents or requests. This technique not only exhibits the algorithm's success in real situations, but also emphasizes its potential to improve communication in many interactive environments.

2.1 Disadvantages in existing system

There are a number of issues with the K-Nearest Neighbors (KNN) model when it comes to hand sign language recognition. Its sensitivity to distracting or loud characteristics is one of its main drawbacks, as it might result in incorrect gesture classification, particularly in intricate or dynamic situations. Furthermore, KNN is less effective for big datasets or real-time applications since it needs to save and compare every training instance for every prediction, which can be computationally costly. Because the model uses basic distance-based categorization, it is also incapable of learning from the data. Moreover, KNN has trouble processing high-dimensional data since its accuracy decreases as the distance calculation loses significance in higher dimensions. The selection of the "k" value is critical to the model's success, although it can be difficult to ascertain.

3. PROPOSED SYSTEM

In this advanced Hand sign-language detection model, Random Forest is a robust and advanced algorithm compared to K-Nearest Neighbors (KNN), particularly adept at handling high-dimensional datasets and noisy or irrelevant features by amalgamating multiple decision trees. It offers the advantage of providing feature importance, aiding in the selection of relevant features for improved model performance. However, due to its complexity, Random Forest can be computationally intensive and has the potential to overfit if not properly tuned.

In contrast, KNN is a simpler algorithm that may be more suitable for small or well-structured datasets. Its straightforward approach makes it easy to understand and implement, but it lacks the sophistication and flexibility of Random Forest. When deciding between the two algorithms, it is essential to consider the nature of the dataset and the specific requirements of the problem at hand. Conducting experiments with both algorithms and evaluating their performance on the given dataset can help determine which approach is more effective for the task.

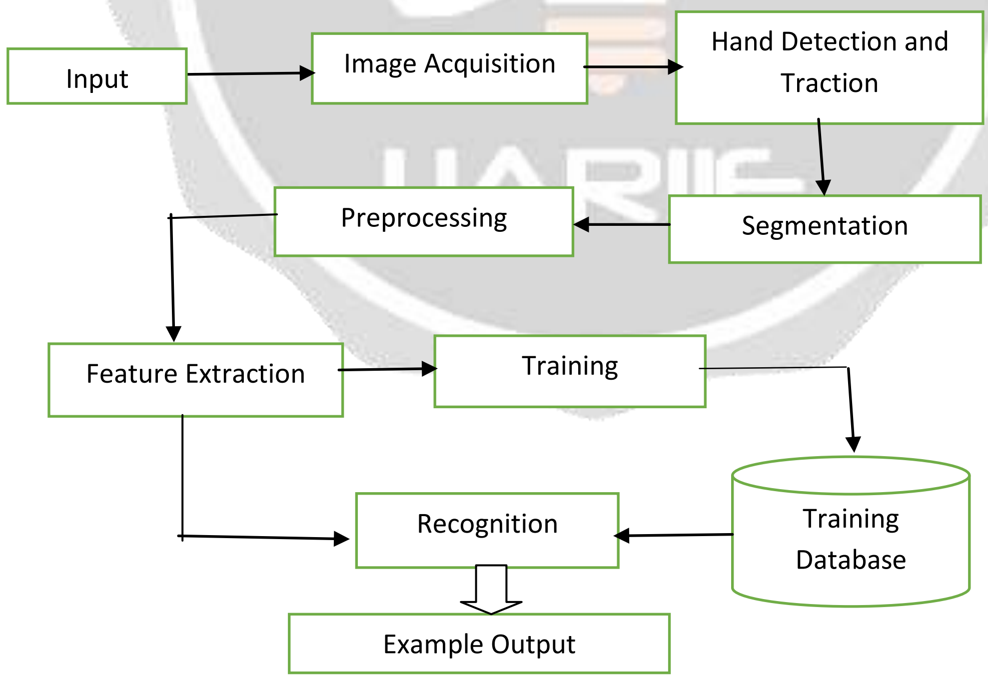

Fig-1: Block Diagram representing the process

3.1 Prerequisites:

- **Visual Studio:** Visual Studio is a powerful developer tool that you can use to complete the entire development cycle in one place. It is a comprehensive integrated development environment (IDE) that you can use to write, edit, debug, and build code, and then deploy your app.
- **Anaconda:** Anaconda is a distribution of Python and R languages tailored for data science and machine learning. It comes with pre-installed packages like Jupyter notebooks and Spyder IDE, simplifying setup and management. Anaconda streamlines workflows for developers and data scientists, providing a robust platform for building data-driven applications.
- **Camera:** A camera is an instrument used to capture and store images and videos, either digitally via an electronic image sensor, or chemically via a light-sensitive material such as photographic film. As a pivotal technology in the fields of photography and videography, cameras have played a significant role in the progression of visual arts, media, entertainment, surveillance, and scientific research. The invention of the camera dates back to the 19th century and has since evolved with advancements in technology, leading to a vast array of types and models in the 21st century.

1.2 Libraries:

- **numpy:** It is a powerful Python library for numerical computing, providing support for arrays, matrices, and mathematical functions. It offers efficient operations on large datasets, making it essential for scientific computing, data analysis, and machine learning tasks. NumPy's array-oriented computing capabilities enable faster execution of mathematical operations compared to traditional Python lists, making it a cornerstone library in the Python ecosystem for numerical computing tasks.
- **MediaPipe:** MediaPipe is an open-source framework for building pipelines to perform computer vision inference over arbitrary sensory data such as video or audio. Using MediaPipe, such a perception pipeline can be built as a graph of modular components. MediaPipe is currently in alpha at v0. You can see this task in action by viewing the web demo.
- **Scikit learn:** It is a widely-used Python library for machine learning, offering a simple and efficient interface for various classification, regression, clustering, and dimensionality reduction algorithms. It provides tools for data preprocessing, model selection, and evaluation, making it an essential tool for building and deploying machine learning models. With scikit-learn, users can easily implement and experiment with different machine learning techniques to solve a wide range of real-world problems.
- **OpenCV:** OpenCV is a Python library that allows you to perform image processing and computer vision tasks. It provides a wide range of features, including object detection, face recognition, and tracking. In this OpenCV Tutorial in Python, we'll be learning more about the library. Python is a general purpose programming language started by Guido van Rossum that became very popular very quickly, mainly because of its simplicity and code readability.
- **Decision tree:** A decision tree is a decision support hierarchical model that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements their possible consequences, including chance events outcomes, resource costs, and utility. It is one way to display an algorithm that only contain conditional control statements.
- **Random forest algorithm:** Random forests or random decision forests is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned.

3.3 Methodology:

In this prediction model the methodology of hand sign language detection using the Random Forest algorithm involves several key steps. First, a dataset of hand sign language images or videos is collected and preprocessed to extract relevant features. These features may include hand shape, movement, and position. Next, the dataset is split into training and testing sets. The Random Forest algorithm is then trained on the training set, where multiple decision trees are built on different subsets of the data. During training, the algorithm learns to classify hand sign gestures based on the extracted features.

Once trained, the model is evaluated on the testing set to assess its performance in recognizing hand sign gestures. The model's hyperparameters, such as the number of trees and maximum depth, can be tuned to improve its performance. Finally, the trained model can be deployed in a real-world application to recognize hand sign gestures in real-time or on new unseen data. Overall, this methodology leverages the power of Random Forest to accurately classify hand sign language gestures, enabling effective communication for the hearing-impaired.

Steps to implement are as follows:

.

- Install the necessary prerequisites on your system to begin the setup process.
- Create a GitHub account using your email address to facilitate version control and collaboration.
- Employ the .sav files generated earlier to integrate model functionalities into a web interface using Spyder.
- Access Download all required libraries in Visual Studio for the project.
- Open Visual Studio and create four new files for the code: “collect_imgs.py”, “create_datasets.py”, train_classifier.py”, and inference_classifier.py”.
- Write the code files one by one. Start with “collect_images.py”, which activates the camera through OpenCV for input.
- Run “creat_datasets.py” to store input data into datasets in a folder called “Date”.
- Execute “train_classifier.py” to train the model using 80% of the data for the training purpose and 20% for the testing.
- Finally, run “inference_classifier.py” to open the camera and detect hand sign language and displaying the meaning in text and audio form.

4. RESULT

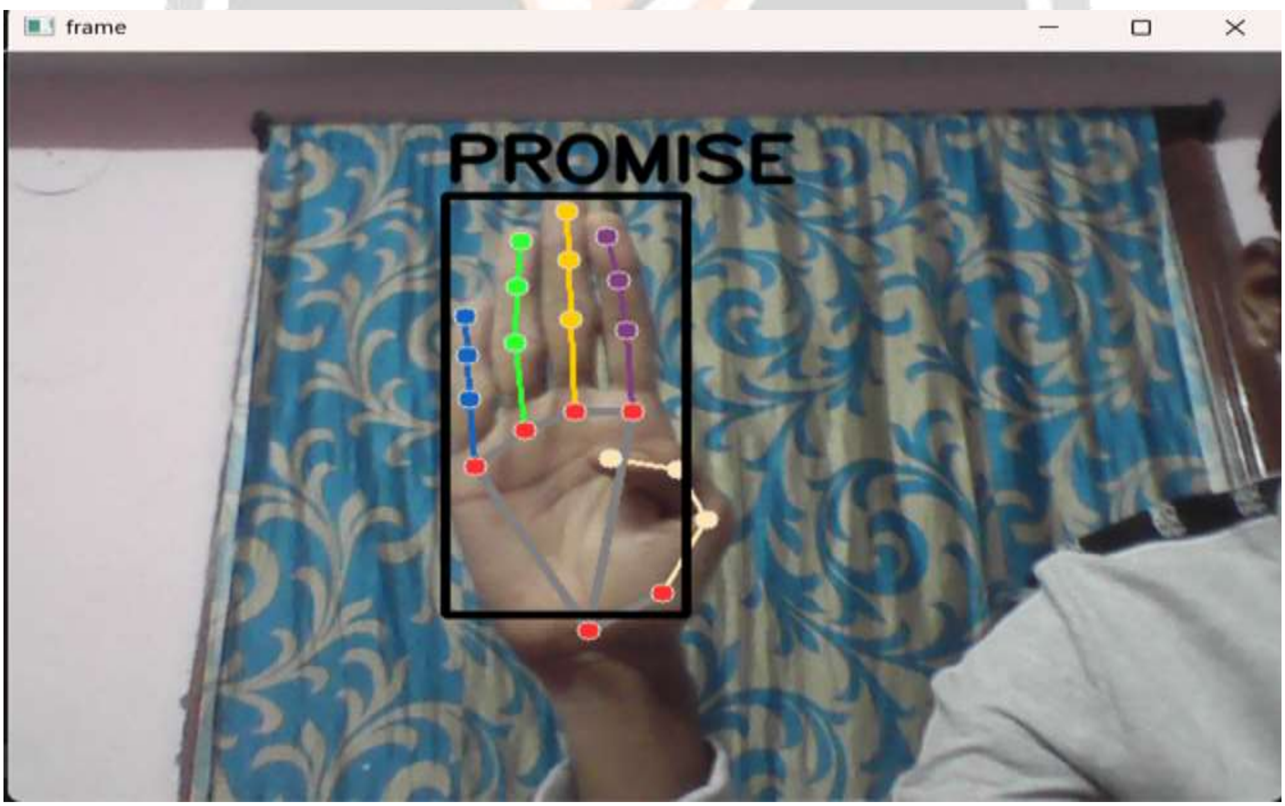

Fig-2: Promise gesture

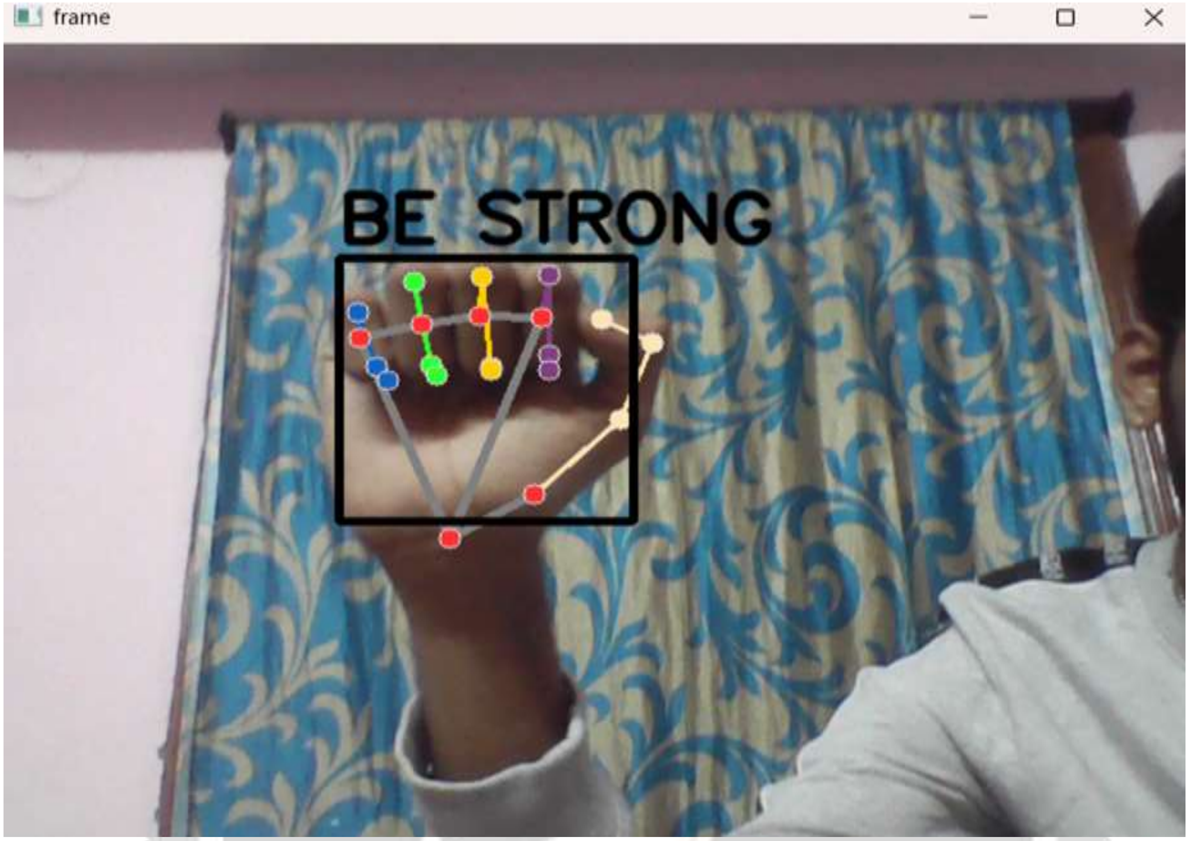

Fig-3: Be strong gesture

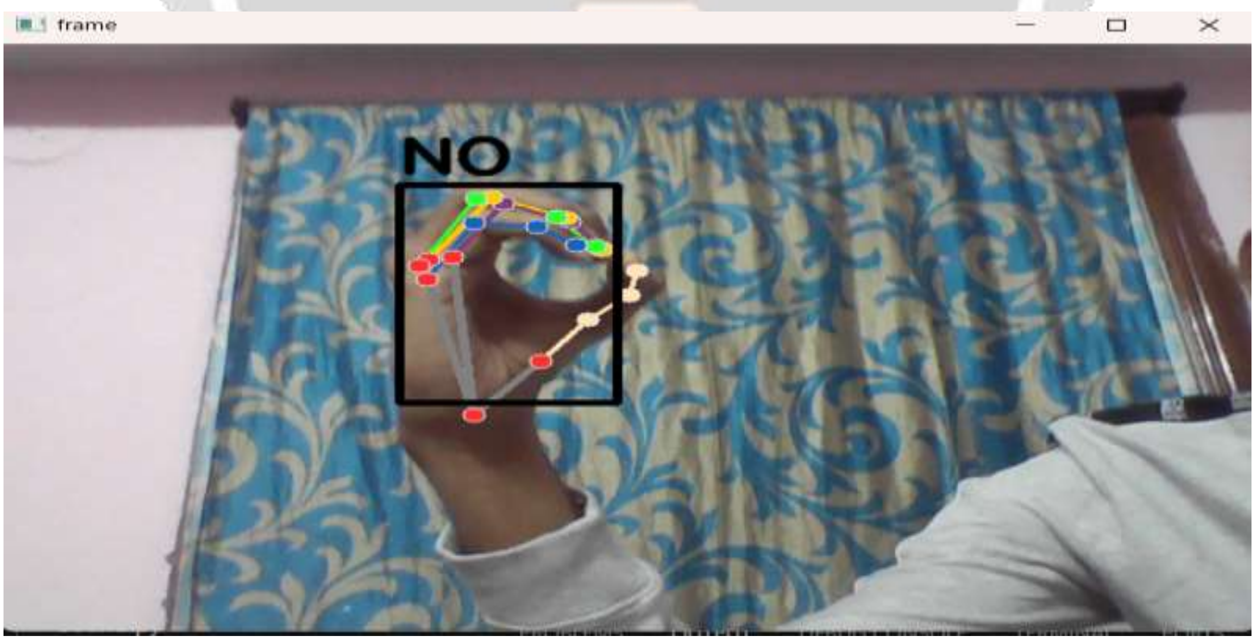

Fig-4: No gesture

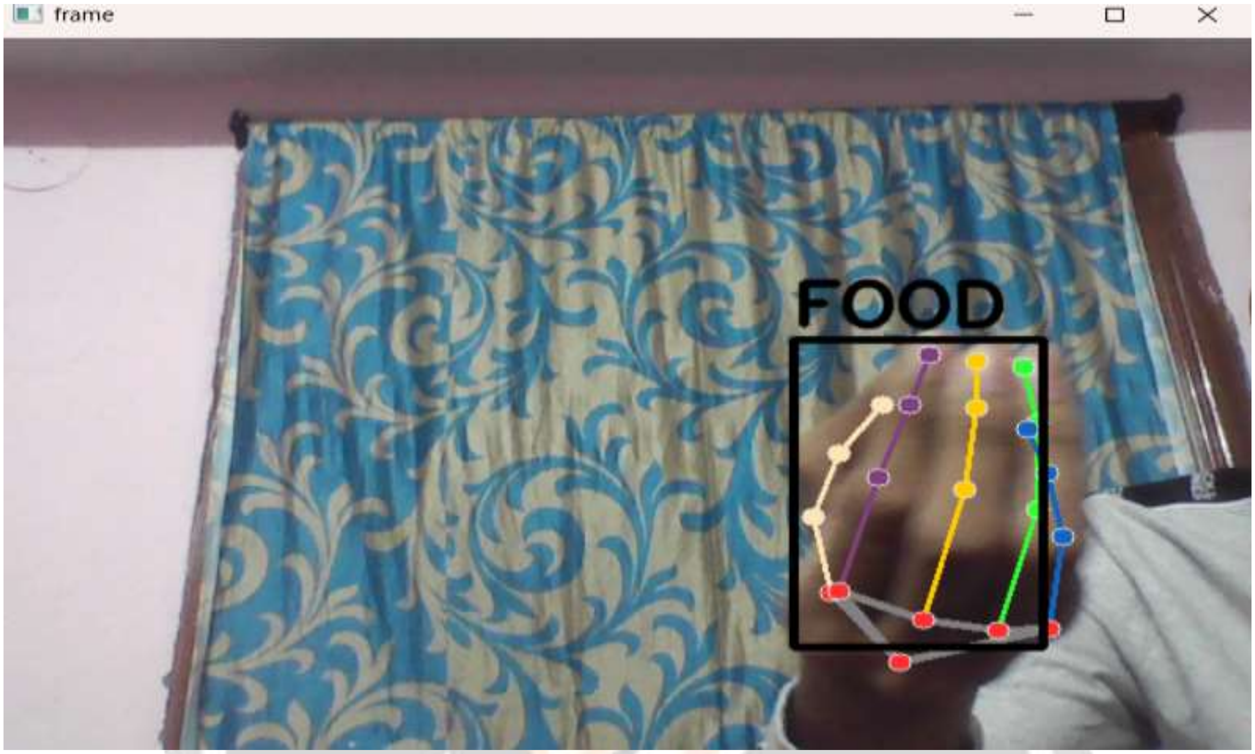

Fig-5: Food gesture

4. CONCLUSION

The project on hand sign language recognition using machine learning focuses on bridging the communication gap between the deaf and hard of hearing community and those unfamiliar with sign language. By leveraging machine learning algorithms like Random Forest, the system aims to accurately detect and translate sign language gestures into text or speech in real-time. The development process involves data collection, preprocessing, model training, and user interface design to create a robust and user-friendly system.

Key components of the project include the use of computer vision techniques to extract features from sign language gestures and the implementation of a Random Forest algorithm for gesture classification. The system's ability to recognize a diverse range of sign language gestures is crucial for its effectiveness in real-world scenarios. Additionally, the integration of audio output enhances the user experience by providing both visual and auditory feedback.

The project's methodology emphasizes the importance of dataset diversity and model tuning to ensure accurate and reliable gesture recognition. By training the model on a comprehensive dataset and optimizing its hyperparameters, the system aims to achieve high accuracy and robustness. The implementation of a user-friendly interface further enhances the system's usability, making it accessible to a wider audience.

In conclusion, the project on hand sign language recognition using machine learning presents a promising solution to facilitate communication between sign language users and non-signers. By combining advanced machine learning techniques with user-friendly design, the system has the potential to improve inclusivity and accessibility for the deaf and hard of hearing community in various communication settings.

6. REFERENCES

[1] Hema B, Sania Anjum, Umme Hani, Vanaja P, Akshatha M "Survey on sign language and gesture recognition system" IRJET -V6-I3 - 2019

[2] Data Flair "Sign language recognition using python and OpenCV"

[3] Math Works "Convolutional neural networks"

[4] J. S. Raikwal, Kanak Saxena "Performance evaluation of SVM and K-Nearest Neighbor Algorithm over medical data set" International Journal of Computer Applications (0975 – 8887) Vol 50 No 14 July 2012

[5] B. Srinivas, G. Sasibhushana Rao "A Hybrid CNN-KNN Model for MRI brain Tumor Classification" International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-8 Issue-2, July 2019