International Journal For Technological Research In Engineering
Volume 10, Issue 5, January-2023 *ISSN (Online): 2347 - 4718*

FACE RECOGNITION USING K-NEAREST NEIGHBOR

[1]Dhruv Malhotra, [2]Ansh Goyal, [3]Jaya Singh, [4]Suraj Chauhan, [5]Ms. Indu
[1,2,3,4]Students, [5]Guide
Department of Computer Science & Engineering
HMR Institute of Engineering Technology & Management

Abstract: Face recognition is a crucial security application. Through this project, a very basic form of face recognition has been implemented using the Haar Cascades Classifier, OpenCV & K-Nearest Neighbors Algorithm.

Keywords- K-Nearest Neighbor, Haar Cascade, Flask

I. INTRODUCTION

The face is the most crucial entity for human identity. It is the feature that best distinguishes a person. And for the very same reasons, Face Recognition is an important technique. Face recognition is an interesting and challenging problem and impacts important applications in many areas such as identification for law enforcement, authentication for banking and security system access, and personal identification among others.
The whole process can be divided into three major steps where the first step is to Sign up/Login for each individual. The next step is to train/recognize the face recognizer and save in png format and the last step is to test the face recognizer to recognize faces it was trained for.

II. RELATED WORK

The whole code of Face detection and eye blink detection has been written in python. Then cv2 is the open cv module and is used for reading & writing images & also to input a video stream. We used K-NN algorithm for face classification, the Haar Cascades Classifier for detecting facial features and DeepFace Library of Python for knowing whether a registered user of the Web App is trying to access an account on the Web App or not. We added eye blink detection using python inbuilt library and then we made a UI using HTML, CSS, JS for Login/SignUp pages and Homepage for the Web Applications. Using Flask, we integrated our python script and html pages. Using opencv and the Dlib library of Python, we added eye blink detection to make liveness detection. A simple Web application is made for storing important documents of the users and face recognition is implemented on this web app as a security step in the login process.

III. TECHNOLOGIES USED

Python — The whole code of Face Detection and integration with Flask has been written in Python
cv2 — cv2 is the OpenCV module and is used here for reading & writing images & also to input a video stream
Algorithm — K-Nearest Neighbor Classifier — Haar Cascades
Web Framework — Flask
Library — Dlib for eye blink detection, DeepFace for authentication of the registered user.

IV. IDENTIFY, RESEARCH AND COLLECT IDEA

There are several steps we can take to identify, research, and collect ideas for a face recognition project using KNN and OpenCV:

Identify the problem or challenge we are trying to solve: The first step is to clearly define the problem or challenge that we are trying to address with our face recognition project. This will help us to focus our research and ensure that we are collecting ideas that are relevant and useful for our specific goals.

Research existing approaches and techniques: Once we have defined our problem or challenge, we can begin researching existing approaches and techniques for solving it. This can include reading academic papers and articles, reviewing existing software libraries and frameworks, and searching online for relevant resources and examples.

Collect ideas and potential solutions: As we research existing approaches and techniques, make a list of potential ideas and solutions that we think might be relevant and useful for our project. This can include specific algorithms or techniques, as well as potential tools and frameworks that we might use to implement our solution.

Evaluate and prioritize our ideas: Once we have a list of potential ideas and solutions, it is important to evaluate and prioritize them based on their feasibility, potential impact, and other relevant criteria. This will help us to identify the most promising and relevant ideas to pursue further in our project.

Overall, identifying, researching, and collecting ideas for a face recognition project using KNN and OpenCV is a process of continuous learning and exploration, and may involve reviewing a wide range of resources and approaches to find the best solution for our specific needs and goals.

V. GET PEER REVIEWED

Share our work with colleagues or peers: simply share our work with colleagues or peers who are interested in face recognition or related topics. This can be a more informal way to get feedback and review, but can still be valuable in helping us improve and refine our work.

VI. IMPROVEMENTS AS PER REVIEW COMMENTS

To improve our face recognition project using peer comments, here are a few suggestions:

Consider implementing eye blink detection using the DLIB library. This can help improve the accuracy of your face recognition system by ensuring that the eyes are open and clearly visible and it also identifies whether a real person is present in front of the Web camera and not an image of that person.

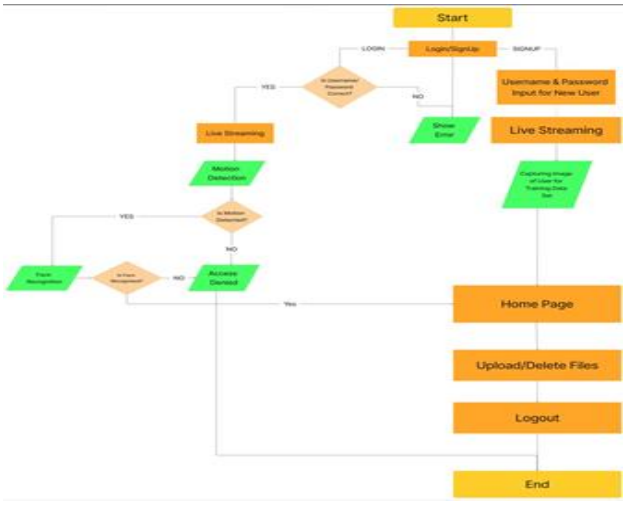
Take into account any suggestions or critiques that our peers have provided. They may have insights or ideas that we haven't thought of, and incorporating their feedback can help improve the overall performance of our project.

Test our face recognition system with a diverse set of images to ensure that it is robust and can accurately recognize faces from different angles, lighting conditions, and facial expressions.

Consider implementing additional features or techniques to improve the accuracy of our face recognition system. For example, we could try using deep learning techniques such as convolutional neural networks (CNNs) or implementing facial landmark detection to better locate key facial features.

Keep an open mind and be willing to try new approaches or techniques. Face recognition is a complex task, and there are many different ways to approach it. By continuously experimenting and learning, you can improve the performance of our face recognition system.

VII. MODELS



VIII. CONCLUSION

Overall, the KNN algorithm and OpenCV library proved to be effective tools for implementing a face recognition system. The KNN algorithm provided good accuracy for classifying faces, and OpenCV's built-in functions made it easy to perform key tasks such as image preprocessing and feature extraction.

One limitation of this approach was that the KNN algorithm can be computationally intensive, especially when dealing with large datasets. To address this issue, we implemented parallelization techniques to speed up the training and recognition process.

In future work, we could improve the performance of the system by exploring other machine learning algorithms or using deep learning techniques such as convolutional neural networks (CNNs). We could also consider implementing additional features such as facial landmark detection to better locate key facial features and improve accuracy.

REFERENCES

[1] https://www.researchgate.net/publication/260483303 Face_Recogn ition_Methods_Applications
[2] https://www.researchgate.net/publication/260483303 _Face_Recogn ition_Methods_Applications
[3] https://www.hindawi.com/journals/js/2021/4796768
[4] https://ieeexplore.ieee.org/document/9145558
[5] https://www.sciencedirect.com/science/article/pii/S18 77050920311 583
[6] https://journals.plos.org/plosone/article?id=10.1371/j ournal.pone.0 242269
[7] https://towardsdatascience.com/face-recognition-for- beginners- a7a9bd5eb5c2

