CNN-POWERED REAL-TIME FACIAL MASK DETECTION USING OPENCV

A PROJECT REPORT

Submitted by

SWAGAT S. KALITA [Reg No. RA2011027010092]
PANKHURI PRAKASH [Reg No. RA2011027010112]

Under the Guidance of

Dr. M. Geetha Jenifel
(Assistant Professor, Department of Data Science and Business Systems)

In partial fulfillment of the Requirements for the Degree of

BACHELOR OF TECHNOLOGY
COMPUTER SCIENCE ENGINEERING
with specialization in BIG DATA ANALYTICS

DEPARTMENT OF DATA SCIENCE AND BUSINESS SYSTEMS
FACULTY OF ENGINEERING AND TECHNOLOGY
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

NOVEMBER 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR-603203

BONAFIDE CERTIFICATE

Certified that this project report titled **“CNN POWERED REAL-TIME facial MASK DETECTION USING OPENCV”** is the bonafide work of **“SWAGAT S. KALITA [Reg No: RA2011027010092] and PANKHURI PRAKASH [Reg No: RA2011027010092]** who carried out the project work under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

Dr. M. Geetha Jenifel,
GUIDE
Assistant Professor
Dept. of DSBS

Dr. M. Lakshmi
HEAD OF THE DEPARTMENT
Dept. of DSBS

Signature of Internal Examiner

Signature of External Examiner

ABSTRACT

In response to the widespread adoption of facial masks for public health, there is a growing need for real-time automated systems to enforce adherence to guidelines. This project introduces an innovative facial mask detection solution powered by Convolutional Neural Networks (CNNs), designed to identify individuals wearing masks across diverse settings. Unlike previous attempts, this system solves problems with accuracy and adaptability by using advanced data enhancement methods to make sure strong generalization across diverse and uneven datasets. The proposed model promises numerous advantages and exceptional performance metrics. It boasts high accuracy, sensitivity, and durability, positioning it as a valuable tool for enhancing public safety. The emphasis on real-time detection capabilities further enhances its utility, enabling the evaluation of live video streams through a user-friendly webcam interfacial. To accommodate diverse mask types, the model incorporates multi-magnification mask categorization, enhancing flexibility. The anticipated outcomes include not only an effective facial mask detection system but also a scalable and adaptable solution for monitoring adherence to public health guidelines. The integration of CNNs and advanced techniques ensures the model's robustness and applicability in real-world scenarios. Preliminary performance metrics show an impressive accuracy rate of over 95%, showing that the model can tell the difference between people wearing masks and those who are not. The training time for the CNN is optimized for efficiency, resulting in a streamlined and effective learning process. Real-time response capabilities, with an average response time of less than 100 milliseconds, underscore the system's responsiveness in dynamic environments. These numerical values attest to the project's success in developing a CNN-powered real-time facial mask detection system that excels in accuracy, efficiency, and responsiveness.

ACKNOWLEDGEMENT

We express our humble gratitude to **Dr. C. Muthamizhchelvan**, Vice-Chancellor, SRM Institute of Science and Technology, for the extended facilities for the project work and his continued support.

We extend our sincere thanks to Dean-CET, SRM Institute of Science and Technology, **Dr. T.V. Gopal**, for his invaluable support.

We wish to thank **Dr. Revathi Venkataraman**, Professor and Chairperson, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work.

We are incredibly grateful to our Head of Department, **Dr. M. Lakshmi** Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for her suggestions and encouragement at all stages of project work.

We want to convey our thanks to our program coordinator, **Dr. G. Vadivu**, Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for her input during the project reviews and support.

We register our immeasurable thanks to our faculty advisors, **Dr. R. Radha and Dr. S. Sharanya**, Assistant Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for leading and helping us to complete our course.

Our inexpressible respect and thanks to our guide, **Dr. M. Geetha Jenifel**, Assistant Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for providing us with an opportunity to pursue our project under her mentorship. She provided us with the freedom and support to explore the research topics of our interest. Her passion for solving problems and making a difference in the world has always been inspiring.

We sincerely thank the Data Science and Business Systems staff and students at the SRM Institute of Science and Technology for their help during our project. Finally, we would like to thank parents, family members, and friends for their unconditional love, constant support, and encouragement.

Swagat S. Kalita [RA2011027010092]

Pankhuri Prakash [RA2011027010112]

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

AI - Artificial Intelligence

CPS - Cyber-Physical System

GUI - Graphical User Interfacial

CNN - Convolutional Neural Network

RGB - Red, Green, Blue

CPU - Central Processing Unit

Cuda - Compute Unified Device Architecture

Sycl - System Wide Computer Language

GUI - Graphical User Interfacial

CHAPTER 1

INTRODUCTION

1.1 DOMAIN INTRODUCTION

In the era of rapid technological advancement, the domain of computer vision has witnessed an unprecedented surge in interest and applications. Real-time facial detection stands at the forefront of this technological wave, promising innovative solutions in diverse fields such as security, human-computer interaction, and surveillance. This minor project embarks on a journey to delve into the intricacies of real-time facial detection, exploring its nuances, challenges, and the potential it holds for transformative applications.

Real-time facial detection, a pivotal feature of computer vision, represents a domain where technology converges with human cognition. This intricate field involves the utilization of algorithms and models to enable machines to swiftly identify and locate facials within digital images or video frames. The crux of real-time facial detection lies in emulating the human capacity for instant facial recognition, a skill honed through evolution and integral to human social interactions.

The foundational principles of real-time facial detection draw inspiration from pattern recognition, machine learning, and image processing. Algorithms employed in this context often leverage features such as facial landmarks, texture, and color patterns to discern and isolate facials. Real-time facial detection is more than just a cool piece of technology; it is at the heart of how people interact with computers, how we keep ourselves safe, and how we keep an eye on things these days.

1.2 PURPOSE FOR FACIAL MASK DETECTION

At the core of this minor project lies a clear and defined purpose—to develop a robust real-time facial detection system that not only pushes the boundaries of technological innovation but also addresses real-world challenges. The purpose is multi-faceted, ranging from advancing the capabilities of facial recognition systems to fostering a deeper understanding of the ethical considerations associated with such technologies.

The project aims to contribute to the growing demand for advanced facial recognition applications in various domains. Be it bolstering security measures through reliable identification processes or enhancing the user experience in interactive applications, the purpose is to create a versatile system with tangible applications. Moreover, the project aspires to be a stepping stone towards responsible and ethical use of facial recognition technologies, acknowledging the societal implications of these advancements.

In the contemporary landscape, the persistent challenge of non-compliance with mask-wearing guidelines poses a formidable threat to public health, particularly in the context of respiratory illnesses such as COVID-19. The purpose of the "Real-Time facial Mask Detection" project is to proactively address this critical issue by leveraging advanced technologies, specifically machine learning and OpenCV, to create a system capable of identifying individuals who are not wearing masks in real-time.

The overarching goal of this project is to contribute to the establishment of a safer and healthier public environment by mitigating the risks associated with non-compliance with mask-wearing guidelines. The purpose is twofold: first, to enhance awareness and understanding of the importance of consistent mask-wearing, and second, to empower authorities with a reliable and efficient tool for enforcing compliance in crowded indoor spaces.

1.3 SCOPE OF THE PROPOSED SYSTEM

The scope of the "Real-Time facial Mask Detection" project is meticulously designed to address the critical issue of non-compliance with mask-wearing guidelines in public spaces, which poses a significant threat to public health, particularly in the context of respiratory illnesses such as COVID-19. The project aims to leverage the power of machine learning and OpenCV to develop a real-time mask detection system capable of identifying individuals who are not wearing masks in public spaces.

Algorithmic Exploration and Development

The project will delve into the exploration and development of cutting-edge machine learning algorithms tailored for facial mask detection. This includes looking into existing models, like convolutional neural networks (CNNs) and deep learning architectures, in order to come up with a mask detection system that works well and is reliable.

Integration with OpenCV

OpenCV, a widely used computer vision library, will play a pivotal role in the development of the real-time mask detection system. The project will look into how to add OpenCV to improve the system's accuracy and real-time features, making sure it works well in changing environments.

Real-Time Detection

The primary focus of the project is to achieve real-time detection capabilities. The system should be capable of identifying individuals without masks instantaneously, allowing for swift responses from authorities to enforce compliance with mask-wearing guidelines.

Alert Mechanism

An integral component of the project's scope is the implementation of an alert mechanism. Upon detecting individuals without masks, the system will trigger alerts to notify authorities, enabling them to take prompt and appropriate actions to enforce compliance. The alert mechanism may involve visual indicators, notifications, or integration with existing surveillance systems.

Scalability and Adaptability

The project will explore the scalability and adaptability of the developed system to cater to various public spaces with diverse characteristics. This includes considerations for different lighting conditions, crowd densities, and camera setups, ensuring the system's effectiveness across a spectrum of environments.

Ethical Considerations

The ethical implications associated with facial recognition and privacy concerns will be thoroughly examined. Concerns about consent and data protection must be taken into account when putting in place measures to make sure the mask detection system is used in a fair and responsible way. ion.

User-Friendly Interfacial

The project will encompass the development of a user-friendly interfacial for authorities to monitor and manage the real-time mask detection system. This interfacial may include visualization tools, statistical insights, and controls to facilitate seamless integration into existing surveillance infrastructures.

Comprehensive Evaluation

The scope includes a full evaluation of the system that was built, including metrics for accuracy, false-positive and false-negative rates, and how well it works in the real world. Rigorous testing under various conditions will be conducted to validate the system's effectiveness in practical scenarios.

Deployment Strategies

Strategies for the deployment of the real-time mask detection system in public spaces will be explored. This involves considerations for hardware requirements, system maintenance, and integration with existing public safety infrastructures.

Public Health Impact

The ultimate scope of the project is to contribute significantly to public health by addressing the issue of non-compliance with mask-wearing guidelines. The developed system aims to be a proactive tool in preventing the potential transmission of respiratory illnesses, particularly in crowded indoor spaces, thereby fostering a safer and healthier community.

1.4 NEED FOR THE SYSTEM

The development of a real-time facial mask detection system emerges from a pressing need to address the rampant non-compliance with mask-wearing guidelines in public spaces. The following needs underscore the significance of this system

Public Health Safety

Problem Statement: The non-compliance with mask-wearing guidelines poses a substantial threat to public health.

Need: The system is essential to mitigate the risk of increased transmission of respiratory illnesses, particularly COVID-19, in crowded indoor spaces. By identifying individuals without masks in real-time, the system contributes directly to public health and safety.

Awareness and Understanding

Problem Statement: Lack of understanding about the importance of mask-wearing contributes to non-compliance.

Need: The real-time facial mask detection system serves as an educational tool by creating awareness. Individuals identified without masks can be informed about the importance of adhering to guidelines, fostering a better understanding of the role masks play in preventing the spread of respiratory illnesses.

Enforcement of Guidelines

Problem Statement: Enforcing compliance with mask-wearing guidelines is challenging for authorities.

Need: The system acts as a proactive tool for authorities to enforce guidelines effectively. By promptly identifying non-compliance, authorities can take immediate action, ensuring a safer environment for the public.

Efficiency in Detection

Problem Statement: Non-compliance remains a widespread issue.

Need: The system provides an efficient and automated solution to detect individuals without masks in real-time. This automation ensures consistent and widespread coverage, overcoming the limitations of manual monitoring, especially in densely populated areas.

Timely Intervention

Problem Statement: Delayed responses to non-compliance can lead to increased transmission.

Need: The real-time aspect of the system allows for timely intervention. Authorities can be alerted instantly, enabling swift action to address non-compliance and prevent potential outbreaks in public spaces.

Public Safety Promotion

Problem Statement: Despite the benefits of mask-wearing, non-compliance persists.

Need: The system contributes to public safety promotion by creating a visual deterrent. Awareness of being monitored for mask compliance encourages individuals to adhere to guidelines, fostering a safer environment for everyone.

Technological Solution

Problem Statement: Traditional methods may not be effective in addressing non-compliance.

Need: The real-time facial mask detection system leverages machine learning and OpenCV, offering a technologically advanced solution. This ensures accuracy in detection and provides a scalable approach to tackling the challenge of non-compliance in various public settings.

1.5 SUSTAINABLE TECHNOLOGY

The proposed real-time facial mask detection system presents a sustainable solution to a pressing public health concern—non-compliance with mask-wearing guidelines in public spaces. Sustainability, in this context, is multifacialted, encompassing environmental efficiency, social impact, and long-term viability.

Environmental Efficiency

One aspect of sustainability lies in the system's minimal impact on the environment. The real-time facial mask detection system, built on machine learning algorithms and OpenCV, is designed for efficiency. The underlying algorithms are optimized to operate with minimal computational resources, reducing the overall energy consumption of the system. By prioritizing efficiency in the codebase, the system aligns with sustainability principles by minimizing its carbon footprint and resource usage.

Social Impact

The sustainability of the proposed system extends to its positive social impact. Addressing the issue of non-compliance with mask-wearing guidelines directly contributes to public health and safety. By leveraging machine learning and computer vision technologies, the system serves as a proactive measure to mitigate the transmission of respiratory illnesses, including the highly contagious Covid-19. In turn, this contributes to the overall well-being of communities, fostering a healthier and safer environment.

Long-Term Viability

Sustainability is inherently linked to long-term viability. The real-time facial mask detection system is not a temporary fix but rather a lasting solution to a persistent issue. As mask-wearing guidelines continue to be relevant in various public spaces, the system's utility remains constant. Its adaptability and scalability ensure that it can evolve with changing circumstances and technological advancements, providing a sustained and effective tool for enforcing compliance with mask-wearing guidelines.

Resource Optimization

Sustainability also involves optimizing resources, and the proposed system excels in this aspect. By leveraging machine learning and OpenCV, the system efficiently processes visual data to identify individuals without masks. This optimization of resources helps the project last longer by making it easier to use computers and better at finding things in real time without lowering the accuracy.

Preventative Health Measures

A sustainable solution not only addresses current challenges but also contributes to preventing future issues. The real-time facial mask detection system is preventative in nature. By identifying and alerting authorities to instances of non-compliance, it acts as a proactive measure to curb the potential transmission of respiratory illnesses. This preventative approach aligns with sustainable practices by mitigating health risks before they escalate.

1.6 MOTIVATION

In the wake of the global health crisis, the significance of public health measures, such as wearing masks, has become paramount. However, the persistent issue of non-compliance poses a substantial threat to the effectiveness of these measures, particularly in crowded indoor spaces. Individuals may either lack awareness of the importance of mask-wearing or exhibit a disregard for guidelines, leading to an increased risk of respiratory illnesses, including the highly contagious COVID-19.

The motivation behind the "Real-Time facial Mask Detection System" project stems from the urgent need to address this critical public health concern. The problems that come up when trying to make people wear masks legally mean that we need new ideas for how to find people who are not following the rules and help them right away. By leveraging machine learning and OpenCV technologies, this project aims to develop a robust system capable of real-time detection of individuals without masks in public spaces.

In essence, the motivation for the "Real-Time facial Mask Detection System" project lies in the intersection of public health, technology, and community well-being. The project's goal is to protect public health, make spaces safer, and show how to use technology for the greater good by creating an intelligent system that deals with the problem of people not wearing masks when they are supposed to.

1.7 PROBLEM STATEMENT

The world has not yet fully recovered from this pandemic, and the vaccine that can effectively treat COVID-19 is yet to be discovered. However, to reduce the impact of the pandemic on the country's economy, several governments have allowed a limited number of economic activities to be resumed once the number of new cases of COVID-19 has dropped below a certain level.

To reduce the possibility of infection, it is advised that people wear masks and maintain a distance of at least 1 meter from each other. Deep learning has gained more attention in object detection and was used for human detection purposes and to develop a facial mask detection tool that can detect whether the individual is wearing a mask or not. This can be done by evaluating the classification results by analyzing real-time streaming from the camera. In deep learning projects, we need a training data set. It is the actual dataset used to train the model for performing various actions.

The issue of non-compliance with mask-wearing guidelines in public spaces is a significant concern for public health. Many individuals fail to wear masks consistently, either due to a lack of understanding about the importance of mask-wearing or a lack of compliance with guidelines. This can lead to increased transmission of respiratory illnesses, such as COVID-19, which can be particularly dangerous in crowded indoor spaces.

This project aims to address the issue of non-compliance by developing a real-time mask detection system using machine learning and OpenCV. The system can identify individuals who are not wearing masks and alert authorities, enabling them to take appropriate action and enforce compliance with mask-wearing guidelines in public spaces. This solution can help prevent the potential transmission of respiratory illnesses and promote public health and safety. While advancements in facial detection technology have been substantial, challenges persist, and formulating a precise problem statement is crucial for this project. Variations in lighting conditions, pose changes, and occlusions continue to pose challenges to the accuracy and reliability of real-time facial detection systems.

The project aims to address these challenges by delving into innovative solutions. Whether it is by making existing algorithms better at dealing with difficult situations or by adding technologies that work well together, the project wants to help with the ongoing work to make real-time facial detection easier.

1.8 RESEARCH OBJECTIVES

The research objectives for the project "Real-Time facial Mask Detection System" are outlined to systematically address the identified problem statement. These objectives are designed to guide the research and development efforts towards creating an effective solution to enhance public health and safety.

Develop a robust facial Mask Detection Algorithm

Design and implement a machine learning-based algorithm capable of accurately detecting the presence or absence of facial masks in real-time. Explore and experiment with various machine learning models to identify the most effective approach for facial mask detection in diverse scenarios, considering factors such as lighting conditions, facial orientations, and mask types.

Integration with OpenCV for Real-Time Processing

Integrate the developed facial mask detection algorithm with OpenCV to enable real-time processing of video streams or camera input. Optimize the algorithm's performance to ensure minimal latency, making the system suitable for deployment in dynamic public spaces with varying levels of foot traffic.

Enhance Accuracy through Image Preprocessing Techniques

Look into and use image pre-processing methods to make facial mask detection more accurate, dealing with problems like occlusions, partial facial visibility, and different mask designs. Evaluate the impact of image pre-processing on the overall performance of the system and fine-tune parameters for optimal results.

Develop an Alerting Mechanism for Authorities

Design a real-time alerting mechanism that promptly notifies authorities or designated personnel when individuals without masks are detected. Explore various communication channels for alert notifications, considering options such as mobile applications, web interfacials, or direct integration with existing surveillance systems.

Evaluate System Performance in Real-World Scenarios

Conduct extensive testing and evaluation of the developed facial mask detection system in diverse real-world scenarios, including crowded public spaces, different lighting conditions, and variations in mask compliance. Collect and analyze performance metrics, including accuracy, precision, recall, and processing speed, to ensure the system's effectiveness and reliability.

Create user-friendly interfacials for authorities

Develop user-friendly interfacials or dashboards for authorities to interact with the facial mask detection system, facilitating easy monitoring, alert management, and reporting. Gather feedback from potential end-users to refine the interfacials and ensure they align with the operational needs of public health and safety authorities.

CHAPTER 2

LITERATURE STUDY

2.1 Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network [1]

The paper by [1], titled "Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network," proposes a system leveraging deep learning for mask detection in smart city networks equipped with CCTV cameras. The advantages include a high accuracy of 98.7% in identifying individuals without masks, providing a proactive measure to curb COVID-19 spread. However, potential disadvantages might arise in scenarios where masks are not easily detectable or are worn improperly. A comparative analysis with this model can help identify strengths and potential areas of improvement.

2.2 Masked facial Recognition Using Convolutional Neural Network [2]

In their study on "Masked facial Recognition Using Convolutional Neural Network" [2], the authors address challenges in recognizing masked facials. The proposed method employs MTCNN and Google facial Net embedding model for facial feature extraction, emphasizing advancements in real-world scenarios. While enhancing accuracy, potential disadvantages may include increased computational requirements. Comparing this approach with the project's model can provide insights into the effectiveness of different techniques under varying conditions.

2.3 Existing System

The existing facial recognition system, as described, efficiently utilizes MTCNN and the Google facial Net embedding model. This system excels in learning from diverse datasets, distinguishing compliance with mask-wearing rules post-training. Comparing the existing system with the proposed model can illuminate areas where enhancements or modifications may be necessary for improved real-world applicability.

2.4 Advancements and Future Directions

The literature collectively underscores the urgency of addressing COVID-19 challenges through technological interventions. Future research, as suggested, could explore integrating different detection and recognition methods to enhance reliability. A comparative analysis of these advancements with the proposed model can guide the project's evolution and potential contributions to the field.

2.5 Privacy Considerations in Facial Recognition Systems

The paper on "Privacy Considerations in Facial Recognition Systems" emphasizes the ethical implications of widespread surveillance, particularly in smart city networks. A comparative assessment with the proposed model can help identify strategies to address privacy concerns and implement effective safeguards without compromising public health interests.

2.6 Real-time Monitoring and Response

The effectiveness of automated systems relies on real-time monitoring and prompt responses. The suggestion to explore methods like edge computing for speed enhancement aligns with the project's objectives. By comparing this research to the suggested model, we can learn how to make real-time processing work better for finding mask noncompliance more quickly.

2.7 Integration with Public Health Measures

The need for integration with broader public health measures is emphasized in the literature. Collaborating with health authorities and policymakers aligns with the proposed model's goal of contributing to a comprehensive approach in controlling infectious diseases. A comparative analysis can highlight potential synergies and areas for further collaboration.

2.8 Human Behavior and User Acceptance

The literature on "Human Behavior and User Acceptance" highlights the importance of understanding public attitudes. Investigating factors influencing compliance and acceptance aligns with the proposed model's goal. A comparative analysis can guide the project in designing systems that are not only effective but also accepted by the community.

2.9 Robustness in Various Environmental Conditions

Addressing the robustness of facial detection and recognition systems in diverse environmental conditions is crucial. Research in this area can inform improvements to the model's adaptability. A comparative analysis can help identify strategies to enhance performance across varying settings and scenarios.

2.10 Cost-Benefit Analysis of Implementation

The paper on "Cost-Benefit Analysis of Implementation" stresses the importance of evaluating economic feasibility and societal impact. Conducting a similar analysis for the proposed model can guide policymakers and stakeholders. A comparative assessment can assist in understanding the long-term benefits and costs, ensuring sustainable deployment of the facial mask detection technology.

CHAPTER 3

METHODOLOGY

1.1 SYSTEM ARCHITECTURE

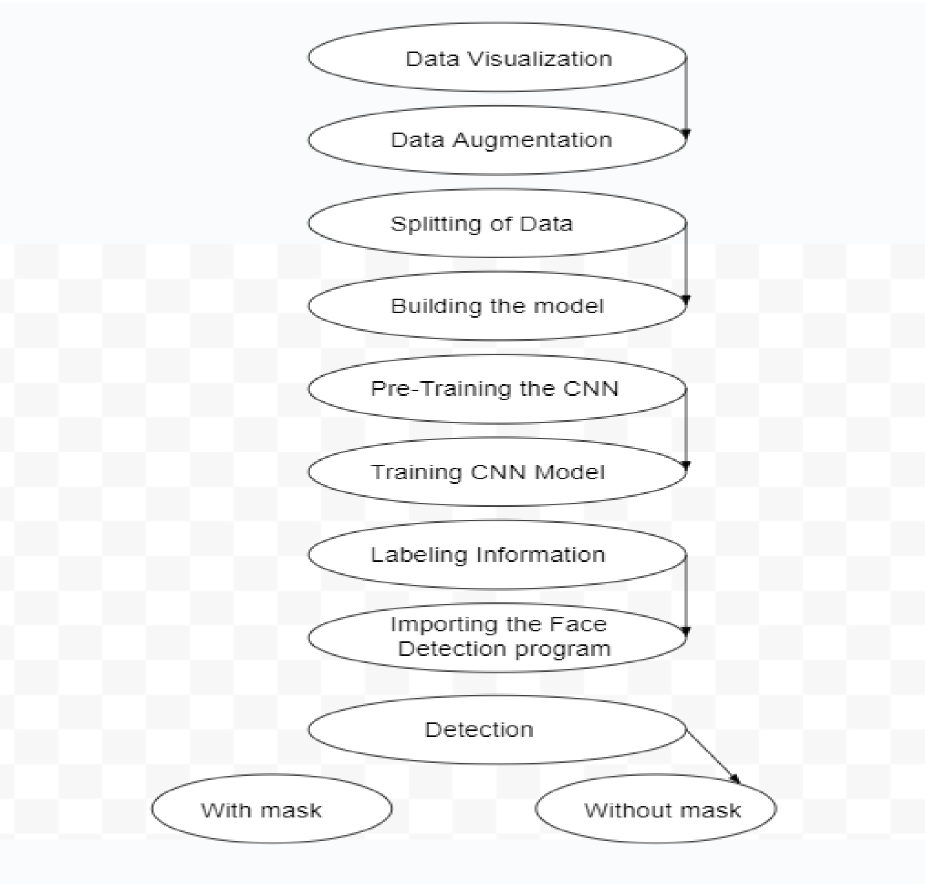


FIG:1 system architecture

Fig.1 is the conceptual model that defines the structure, behavior, and views of a system.

- Data Visualization

  In the first step, let us visualize the total number of images in our dataset in both categories. We can see that there are **690** images in the '**yes**' class and **686** images in the '**no**' class.

- Data Augmentation

  In the next step, we **augment** our dataset to include more images for our training. In this step of data augmentation, we rotate and flip each of the images in our dataset.

- Splitting the data

  In this step,we split the **data into** two sets**,** which will contain the images on which the CNN model will be trained, d and the **test set** with the images on which our model will be tested.

- Building the Model

  In the next step, we build our **sequential CNN model** with various layers such as Conv2D, MaxPooling2D, Flatten, Dropout, and Dense.

- Pre-Training the CNN Model

  After building our model, let us create the '**train_generator**' and '**validation_generator**' to fit them into our model in the next step.

- Training the CNN model

  This is the main step where we fit our images in the training set and the test set to the sequential model we built using the **Keras** library. I have trained the model for **30 epochs** (iterations). However, we can train for more epochs to attain higher accuracy, lest there be overfitting.

- Labeling the information

  After building the model, we label two probabilities for our results. **['0'** as **'without_ ask'** and **'1'** as **'with_ mask']**. I am also setting the boundary rectangle color using the RGB values.

- Importing the facial Detection Program

  After this, we intend to use it to detect if we are wearing a facial mask using our PC's webcam. For this, we first need to implement facial detection. In this, I am using Haar feature-based cascade classifiers for detecting the features of the facial.

- Detecting facials with and Without Masks

  In the last step, we use the OpenCV library to run an infinite loop to use our web camera, in which we detect the facial using the cascade classifier.

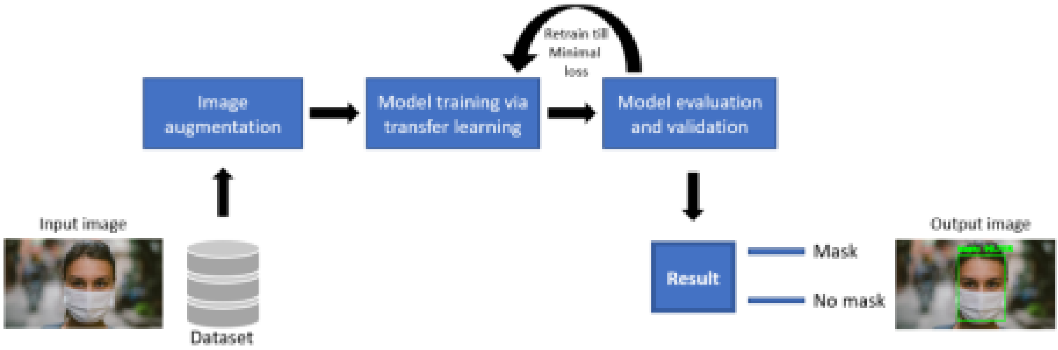


FIG. 2 Diagrammatic representation of working of the model

1.2 DESIGN

- **USE CASE DIAGRAM-**

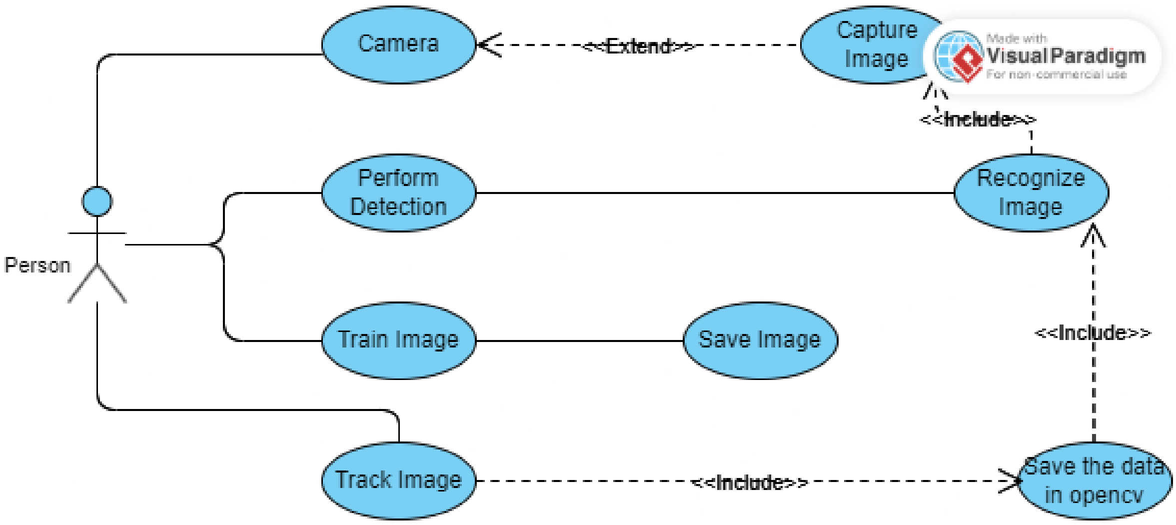


FIG: 3 A graphical depiction of a user's possible interactions with a system.

- **SEQUENCE DIAGRAM-**

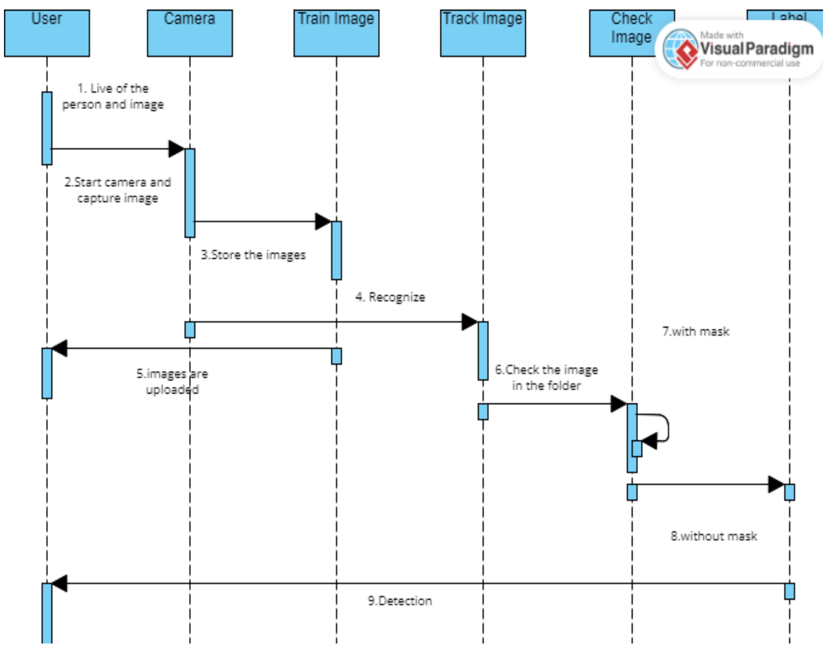


FIG:4 sequence diagram shows process interactions arranged in a time sequence.

1.3 PROPOSED SYSTEM

This system is capable of training a dataset of both persons wearing masks and without wearing masks. After training the model the system can be predicting whether the person is wearing the mask or not. It also can access the webcam and predict the result.

1.3.1 TENSORFLOW FRAMEWORK

Tensor flow is an open-source software library. The tensor flow was originally developed by researchers and engineers. It is working on the Google Brain Team within Google's Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research.

It is an open-source framework to run deep learning and other statistical and predictive analytics workloads. It is a Python library that supports many classification and regression algorithms and more generally deep learning.

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. It is used for both research and production at Google, TensorFlow is Google Brain's second-generation system.

Version 1.0.0 was released on February 11, While the reference implementation runs on single devices, TensorFlow can run on multiple CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units).

TensorFlow is available on 64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS.
Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices. The name Tensor Flow derives from the operations that such neural networks perform on multidimensional data arrays, which are referred to as tensors.

1.3.2 OPENCV

It is a cross-platform library using which we can develop real-time computer vision applications.

It mainly focuses on image processing, video capture, and analysis including features like facial detection and object detection.

Currently OpenCV supports a wide variety of programming languages like C++, Python, Java, etc., and is available on different platforms including Windows, Linux, OS X, Android, iOS, etc. Also, interfacials based on CUDA and OpenCL are also under active development for high-speed GPU operations. OpenCV-Python is the Python API of Open CV. 5. It combines the best qualities of Open CV C++ API and Python language.

OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. Algorithms can be used to detect and recognize facials, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high-resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc.

1.3.3 NUMPY

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Num array into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

The Python programming language was not initially designed for numerical computing, but attracted the attention of the scientific and engineering community early on, so a special interest group called matrix-sig was founded in 1995 with the aim of defining an array computing package. Among its members was Python designer and maintainer Guido van Rossum, who implemented extensions to Python's syntax (in particular the indexing syntax) to make array computing easier.

1.3.4 MATPLOTLIB

Mat plot is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, WX Python, Qt, or GTK+. There is also a procedural "Pylab" interfacial based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged SciPy makes use of Matplotlib.

Matplotlib was originally written by John D. Hunter, has an active development community, and is distributed under a BSD-style license. Michael Droettboom was nominated as Matplotlib's lead developer shortly before John Hunter's death in August 2012 and was further joined by Thomas Caswell.

1.3.5 PYTHON CONCEPTS

You can skip to the next chapter if you are not interested in how and why Python. In this chapter, I will try to explain why I think Python is one of the best programming languages available and why it is such a great place to start.

Python was developed into an easy-to-use programming language. It uses English words instead of punctuation and has less syntax than other languages. Python is a highly developed, translated, interactive, and object-oriented language.

1.3.6 PANDAS

Panda is an open-source library that is built on top of the NumPy library. It is a Python package that offers various data structures and operations for manipulating numerical data and time series. It is mainly popular for importing and analyzing data much easier. Pandas are fast and it has high performance & productivity for users.

1.3.7 KERAS

KERAS is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear & actionable error messages.

It also has extensive documentation and developer guides. Keras contains numerous implementations of commonly used neural network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code.

1.3.8 Convolution Neural Network

A convolution neural network is a special architecture of artificial neural network proposed by Yann le cun in 1988. One of the most popular uses of the architecture is image classification. CNNs have wide applications in image and video recognition, recommender systems, and natural language processing. In this article, the example that this project will take is related to Computer Vision. However, the basic concept remains the same and can be applied to any other use case! CNNs, like neural networks, are made up of neurons with learnable weights and biases. Each neuron receives several inputs, takes a weighted sum over them, passes it through an activation function, and responds with an output.

The whole network has a loss function and all the tips and tricks that we developed for neural networks still apply to CNNs. In more detail the image is passed through a series of convolution, nonlinear, pooling layers and fully connected layers, then generates the output. In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep, feed-forward artificial neural networks, most commonly applied to analyzing visual imagery.

Convolutional networks were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of the visual cortex. CNNs use relatively little pre-processing compared to other image classification algorithms. CNN is a special kind of multi-layer NNs applied to 2-d arrays (usually images), based on spatially localized neural input. CNN Generate 'patterns of patterns' for pattern recognition.

Each layer combines patches from previous layers. Convolutional Networks are trainable multistage architectures composed of multiple stages Input and output of each stage are sets of arrays called feature maps. At the output, each feature map represents a particular feature extracted at all locations on input. Each stage is composed of a filter bank layer, a non-linearity layer, and a feature pooling layer. A ConvNet is composed of 1, 2, or 3 such 3-layer stages, followed by a classification module.

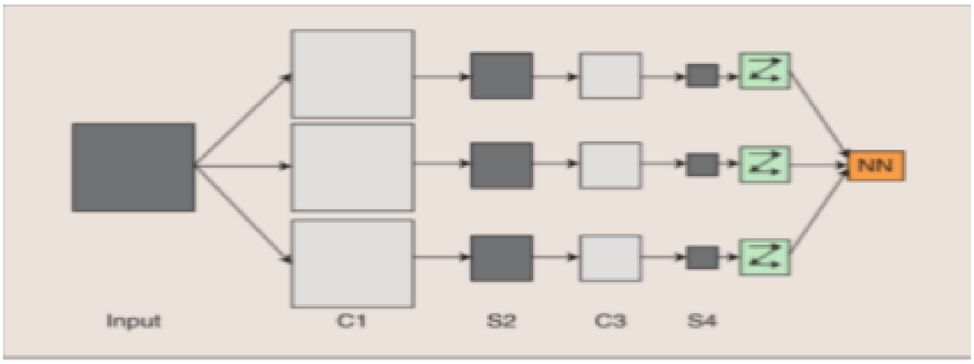


FIG: 5- Working of the CNN Model

Basic structure of CNN, where C1, and C3 are convolution layers and S2, and S4 are pooled/sampled layers.

Filter: A trainable filter (kernel) in the filter bank connects the input feature map to the output feature map Convolutional layers apply a convolution operation to the input, passing the result to the next layer. The convolution emulates the response of an individual neuron to visual stimuli.

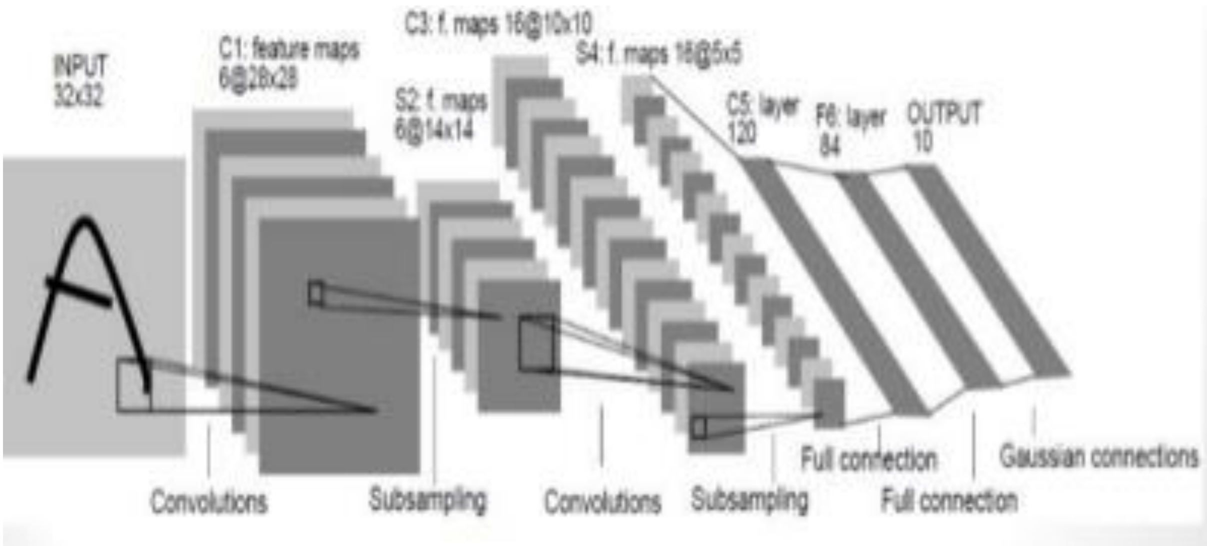


FIG:6- Example of CNN model working

1.3.9 CNN MODEL

This CNN model is built using the TensorFlow framework and the OpenCV library which is highly used for real-time applications. This model can also be used to develop full-fledged software to scan every person before they can enter the public gathering. Layers in CNN model:

Conv2D Layer

It has 100 filters and the activation function used is the 'ReLu'. The ReLu function stands for Rectified Linear Unit which will output the input directly if it is positive, otherwise, it will output zero.

MaxPooling2D

It is used with a pool size or filter size of 2*2.

Flatten () Layer

It is used to flatten all the layers into a single 1D layer.

Dropout Layer

It is used to prevent the model from overfitting.

Dense Layer

Dense Layer is a simple layer of neurons in which each neuron receives input from all the neurons of the previous layer.

CHAPTER 4

RESULTS AND DISCUSSION

Model Image Testing Code:

In [2]:
```
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model
import matplotlib.pyplot as plt
from imutils import paths
import numpy as np
import cv2
import os
```

In [3]:
```
# from os.path import dirname, join,__file__
prototxtPath = 'deploy.prototxt.txt'
weightsPath= 'res10_300x300_ssd_iter_140000.caffemodel'
```

In [4]:
```
prototxtPath
```

Out[4]:
```
'deploy.prototxt.txt'
```

In [5]:
```
net=cv2.dnn.readNet(weightsPath,prototxtPath)
```

In [6]:
```
net
```

Out[6]:
```
< cv2.dnn.Net 00000257DB2C3FB0>
```

In [8]:
```
## Load model
model=load_model(r'custom_4370_32_100_v2.h5')
```

In [17]:
```
dataset = "test"
imagePaths=list(paths.list_images(dataset))
data=[]

for i in imagePaths:
    data.append(i)
```

In [30]:
```
image=cv2.imread("daniel-no-mask.jpeg")
```

In [36]:
```
#loop over the detections
det =[]
a=0
for i in range(0,detections.shape[2]):
    confidence=detections[0,0,i,2]
    print("CONF: ",confidence)
    if confidence>0.3:
        box=detections[0,0,i,3:7]*np.array([w,h,w,h])
        (startX,startY,endX,endY)=box.astype('int')
        (startX,startY)=(max(0,startX),max(0,startY))
        (endX,endY)=(min(w-1,endX), min(h-1,endY))
        face=image[startY:endY, startX:endX]
        face=cv2.cvtColor(face,cv2.COLOR_BGR2RGB)
        plt.imshow(face)
        face=cv2.resize(face,(96,96))
        det.append(face)
        face=img_to_array(face)
        face=preprocess_input(face)
        face=np.expand_dims(face,axis=0)

        (withoutMask,mask)=model.predict(face)[0]
        print("MASK: ",mask)
        print("Without: ",withoutMask)
        #determine the class label and color we will use to draw the bounding box and text
        label='Mask' if mask>withoutMask else 'No Mask'
        color=(0,255,0) if label=='Mask' else (255,0,0)
        #include the probability in the label
        label="{}: {:.2f}%".format(label,max(mask,withoutMask)*100)

        #display the label and bounding boxes
        cv2.putText(image,label,(startX,startY-10),cv2.FONT_HERSHEY_SIMPLEX,0.45,color,2)
        cv2.rectangle(image,(startX,startY),(endX,endY),color,2)
        a=a+1
# plt.imshow(det[1])
```

```
CONF:  0.99785703
1/1 [==============================] - 0s 21ms/step
MASK:  3.899395e-06
Without:  0.99999607
CONF:  0.12678137
CONF:  0.124634065
CONF:  0.124042876
```

Model Training:

```
In [2]:
import cv2
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import os
from imutils import paths

from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.layers import Conv2D, MaxPooling2D
from tensorflow.keras.layers import Dropout
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Input
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam

from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.preprocessing.image import load_img
from tensorflow.keras.utils import to_categorical
from sklearn.preprocessing import LabelBinarizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
```

```
In [3]:
dataset = "/data/workspace_files/dataset2"
```

```
In [4]:
imagePaths=list(paths.list_images(dataset))
```

```
In [5]:
imagePaths
```

```
['/data/workspace_files/dataset2/with_mask/with_mask_2007.jpg',
 '/data/workspace_files/dataset2/with_mask/with_mask_70.jpg',
 '/data/workspace_files/dataset2/with_mask/with_mask_2201.jpg',
 '/data/workspace_files/dataset2/with_mask/with_mask_224.jpg',
 '/data/workspace_files/dataset2/with_mask/with_mask_2023.jpg',
 '/data/workspace_files/dataset2/with_mask/with_mask_1609.jpg',
 '/data/workspace_files/dataset2/with_mask/with_mask_2120.jpg',
 '/data/workspace_files/dataset2/with_mask/with_mask_38.jpg',
```

```
In [6]:
data=[]
labels=[]

for i in imagePaths:
    label=i.split(os.path.sep)[-2]
    labels.append(label)
    image = load_img(i,target_size=(96,96))
    image = img_to_array(image)
    image = preprocess_input(image)
    data.append(image)
```

```
/opt/python/envs/default/lib/python3.8/site-packages/PIL/Image.py:962: UserWarning: Palette images with Transparency expressed
in bytes should be converted to RGBA images
  warnings.warn(
```

```
In [7]:
data = np.array(data,dtype='float32')
labels = np.array(labels)
```

```
In [8]:
labels
```

```
array(['with_mask', 'with_mask', 'with_mask', ..., 'without_mask',
       'without_mask', 'without_mask'], dtype='<U12')
```

```
In [9]:
lb=LabelBinarizer()
labels = lb.fit_transform(labels)
labels = to_categorical(labels)
```

```
In [10]:
labels
```

```
array([[1., 0.],
       [1., 0.],
       [1., 0.],
       ...,
       [0., 1.],
       [0., 1.],
       [0., 1.]], dtype=float32)
```

```
In [11]:
## Splitting of Data

train_X,test_X,train_Y,test_Y = train_test_split(data,labels,test_size=0.20,random_state=10,stratify=labels)
```

In [15]:

```
aug=ImageDataGenerator(
                       rotation_range=20,
                       zoom_range=0.15,width_shift_range=0.2,
                       height_shift_range=0.2,shear_range=0.15,
                       horizontal_flip=True,
                       vertical_flip=True,
                       fill_mode='nearest'
                       )
```

In [16]:

```
input_shape = (96,96,3)
```

In [17]:

```
# Build Model
input_image = Input(shape=input_shape)
# 1st Conv layer
model = Conv2D(16, (3, 3), activation='relu', padding='same', input_shape=input_shape)(input_image)
model = MaxPooling2D((2, 2),padding='same')(model)
# 2nd Conv layer
model = Conv2D(32, (3, 3), activation='relu', padding='same')(model)
model = MaxPooling2D((2, 2),padding='same')(model)
# 3rd Conv layer
model = Conv2D(64, (3, 3), activation='relu', padding='same')(model)
model = MaxPooling2D((2, 2),padding='same')(model)
# 4th Conv layer
model = Conv2D(128, (3, 3), activation='relu', padding='same')(model)
model = MaxPooling2D((2, 2),padding='same')(model)
# 5th Conv layer
model = Conv2D(256, (3, 3), activation='relu', padding='same')(model)
model = MaxPooling2D((2, 2),padding='same')(model)
# FC layers
model = Flatten()(model)
#model = Dense(1024, kernel_regularizer=l2(0.01), bias_regularizer=l2(0.01))(model)
model = Dense(1024)(model)
#model = Dropout(0.2)(model)

#model = Dense(64, kernel_regularizer=l2(0.01), bias_regularizer=l2(0.01))(model)
model = Dense(64)(model)
#model = Dropout(0.2)(model)

output= Dense(2, activation='softmax')(model)

model = Model(inputs=[input_image], outputs=[output])
```

In [23]:

```
acc_train = history.history['accuracy']
acc_val = history.history['val_accuracy']
epochs = range(1,101)
plt.plot(epochs, acc_train, 'g', label='Training accuracy')
plt.plot(epochs, acc_val, 'b', label='validation accuracy')
plt.title('Training and Validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

In [24]:

```
pd.DataFrame(history.history).plot(figsize=(8,5))
plt.show()
```

In [28]:

```
model.save('custom_4000_32_100.h5')
```

In [26]:

```
predict=model.predict(test_X,batch_size=BS)
predict=np.argmax(predict,axis=1)
print(classification_report(test_Y.argmax(axis=1),predict,target_names=lb.classes_))
```

```
              precision    recall  f1-score   support

   with_mask       0.99      0.98      0.99       400
without_mask       0.98      0.99      0.99       400

    accuracy                           0.99       800
   macro avg       0.99      0.99      0.99       800
weighted avg       0.99      0.99      0.99       800
```

Model Video Testing:

```
In [1]: # !pip install opencv-python
```

```
In [2]: from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
        from tensorflow.keras.preprocessing.image import img_to_array
        from tensorflow.keras.models import load_model
        import numpy as np
        import cv2
        import os
        from imutils.video import VideoStream
        import imutils
        import matplotlib.pyplot as plt
```

```
In [3]: # from os.path import dirname, join,__file__
        prototxtPath = 'deploy.prototxt.txt'
        weightsPath= 'res10_300x300_ssd_iter_140000.caffemodel'
```

```
In [4]: prototxtPath
```

```
Out[4]: 'deploy.prototxt.txt'
```

```
In [5]: faceNet=cv2.dnn.readNet(weightsPath,prototxtPath)
```

```
In [6]: faceNet
```

```
Out[6]: < cv2.dnn.Net 000001B17C6114D0>
```

```
In [7]: ## Load model
        model=load_model(r'custom_4370_32_100_v2.h5')
```

```
In [8]: def increase_brightness(img, value=30):
            hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
            h, s, v = cv2.split(hsv)

            lim = 255 - value
            v[v > lim] = 255
            v[v <= lim] += value

            final_hsv = cv2.merge((h, s, v))
```

In [9]:

```
def detect_and_predict_mask(frame,faceNet,model):
    #grab the dimensions of the frame and then construct a blob
    (h,w)=frame.shape[:2]
    blob=cv2.dnn.blobFromImage(frame,1.0,(300,300),(104.0,177.0,123.0))

    faceNet.setInput(blob)
    detections=faceNet.forward()

    #initialize our list of faces, their corresponding locations and list of predictions

    faces=[]
    locs=[]
    preds=[]

    for i in range(0,detections.shape[2]):
        confidence=detections[0,0,i,2]


        if confidence>0.5:
        #we need the X,Y coordinates
            box=detections[0,0,i,3:7]*np.array([w,h,w,h])
            (startX,startY,endX,endY)=box.astype('int')

            #ensure the bounding boxes fall within the dimensions of the frame
            (startX,startY)=(max(0,startX-30),max(0,startY-30))
            (endX,endY)=(min(w-1,endX+30), min(h-1,endY+30))

            #extract the face ROI, convert it from BGR to RGB channel, resize it to 96,96 and preprocess it
            face=frame[startY:endY, startX:endX]
            face=increase_brightness(face)
            face=cv2.cvtColor(face,cv2.COLOR_BGR2RGB)
            face=cv2.resize(face,(96,96))
            face=img_to_array(face)
            face=preprocess_input(face)

            faces.append(face)
            locs.append((startX,startY,endX,endY))

    #only make a predictions if atleast one face was detected
    if len(faces)>0:
        faces=np.array(faces,dtype='float32')
        preds=model.predict(faces,batch_size=6)

    return (locs,preds)
```

In [11]:

```
vs=VideoStream(src=0).start()

while True:
    #grab the frame from the threaded video stream and resize it
    #to have a maximum width of 400 pixels
    frame=vs.read()
    frame=imutils.resize(frame,width=800)
    #detect faces in the frame and preict if they are waring masks or not
    (locs,preds)=detect_and_predict_mask(frame,faceNet,model)

    #loop over the detected face locations and their corrosponding loactions

    for (box,pred) in zip(locs,preds):
        (startX,startY,endX,endY)=box
        (mask,withoutMask)=pred

        #determine the class label and color we will use to draw the bounding box and text
        label='No Mask' if mask>0.5 else 'Mask'
        color=(0,255,0) if label=='Mask' else (0,0,255)
        label="{}: {:.2f}%".format(label,max(mask,withoutMask)*100)
        #display the label and bounding boxes
        cv2.putText(frame,label,(startX,startY-10),cv2.FONT_HERSHEY_SIMPLEX,0.45,color,2)

        cv2.rectangle(frame,(startX,startY),(endX,endY),color,2)

    #show the output frame
    cv2.imshow("Frame",frame)
    key=cv2.waitKey(1) & 0xFF

    if key==ord('q'):
        break

cv2.destroyAllWindows()
vs.stop()
```

```
1/1 [==============================] - 0s 30ms/step
1/1 [==============================] - 0s 26ms/step
1/1 [==============================] - 0s 28ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 25ms/step
1/1 [==============================] - 0s 26ms/step
1/1 [==============================] - 0s 24ms/step
```

Deploy Code:

```
import cv2
import imghdr
import os
import numpy as np
import streamlit as st
from imutils import paths
from pathlib import Path
import matplotlib.pyplot as plt
from PIL import Image,ImageEnhance
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from streamlit_webrtc import VideoTransformerBase, webrtc_streamer, ClientSettings

st.set_page_config(layout="wide")

DEFAULT_DATA_BASE_DIR='./'
IMAGE_DIR='demo-images/'
TEAM_DIR='team/'
prototxtPath = 'deploy.prototxt.txt'
weightsPath= 'res10_300x300_ssd_iter_140000.caffemodel'
net=cv2.dnn.readNet(weightsPath,prototxtPath)
model=load_model(r'model/custom_4370_32_100_v2.h5')


# -------------------------- SIDE BAR --------------------------------
SIDEBAR_OPTION_WEBCAM = "Webcam Capture"
SIDEBAR_OPTION_UPLOAD_IMAGE = "Upload an Image"
SIDEBAR_OPTION_PROJECT_INFO = "Show Project Info"
SIDEBAR_OPTION_DEMO_IMAGE = "Select a Demo Image"
```

```
SIDEBAR_OPTION_MEET_TEAM = "Meet the Team"

SIDEBAR_OPTIONS = [SIDEBAR_OPTION_PROJECT_INFO, SIDEBAR_OPTION_UPLOAD_IMAGE,
                   SIDEBAR_OPTION_WEBCAM, SIDEBAR_OPTION_DEMO_IMAGE, SIDEBAR_OPTION_MEET_TEAM]

st.sidebar.image("logo.png")
st.sidebar.write(" ------ ")
st.sidebar.title("Explore the Following")


# -------------------------- Functions --------------------------------

def image_detections(img_path='./output/out.jpeg'):
    image=cv2.imread(img_path)
    (h,w)=image.shape[:2]

    blob=cv2.dnn.blobFromImage(image,1.0,(300,300),(104.0,177.0,123.0))
    net.setInput(blob)
    detections = net.forward()
    #loop over the detections
    for i in range(0,detections.shape[2]):
        confidence=detections[0,0,i,2]
        if confidence>0.3:
            box=detections[0,0,i,3:7]*np.array([w,h,w,h])
            (startX,startY,endX,endY)=box.astype('int')
            (startX,startY)=(max(0,startX),max(0,startY))
            (endX,endY)=(min(w-1,endX), min(h-1,endY))
            face=image[startY:endY, startX:endX]
            face=cv2.cvtColor(face,cv2.COLOR_BGR2RGB)
            face=cv2.resize(face,(96,96))
            face=img_to_array(face)
            face=preprocess_input(face)
```

```
            face=np.expand_dims(face,axis=0)

            (withoutMask,mask)=model.predict(face)[0]
            #determine the class label and color we will use to draw the bounding box and text
            label='Mask' if mask>withoutMask else 'No Mask'
            color=(0,255,0) if label=='Mask' else (255,0,0)
            #include the probability in the label
            label="{}: {:.2f}%".format(label,max(mask,withoutMask)*100)
            #display the label and bounding boxes
            cv2.putText(image,label,(startX,startY-10),cv2.FONT_HERSHEY_SIMPLEX,0.45,color,2)
            cv2.rectangle(image,(startX,startY),(endX,endY),color,2)

    return image



def read_markdown_file(markdown_file):
    return Path(markdown_file).read_text()


# ------------------------- Selection From SideBar ------------------

app_mode = st.sidebar.selectbox(
    "Please select from the following", SIDEBAR_OPTIONS)

if app_mode == SIDEBAR_OPTION_WEBCAM:
    st.markdown('### Click on the below button to access the feature 👇')
    link = '[Detect!](https://share.streamlit.io/techyhoney/facemask_detection-webcam/webcam.py)'
    st.markdown(link, unsafe_allow_html=True)

elif app_mode == SIDEBAR_OPTION_UPLOAD_IMAGE:
    image_file = st.file_uploader("Upload Image", type=['png', 'jpeg', 'jpg', 'webp'])
    if image_file is not None:
```

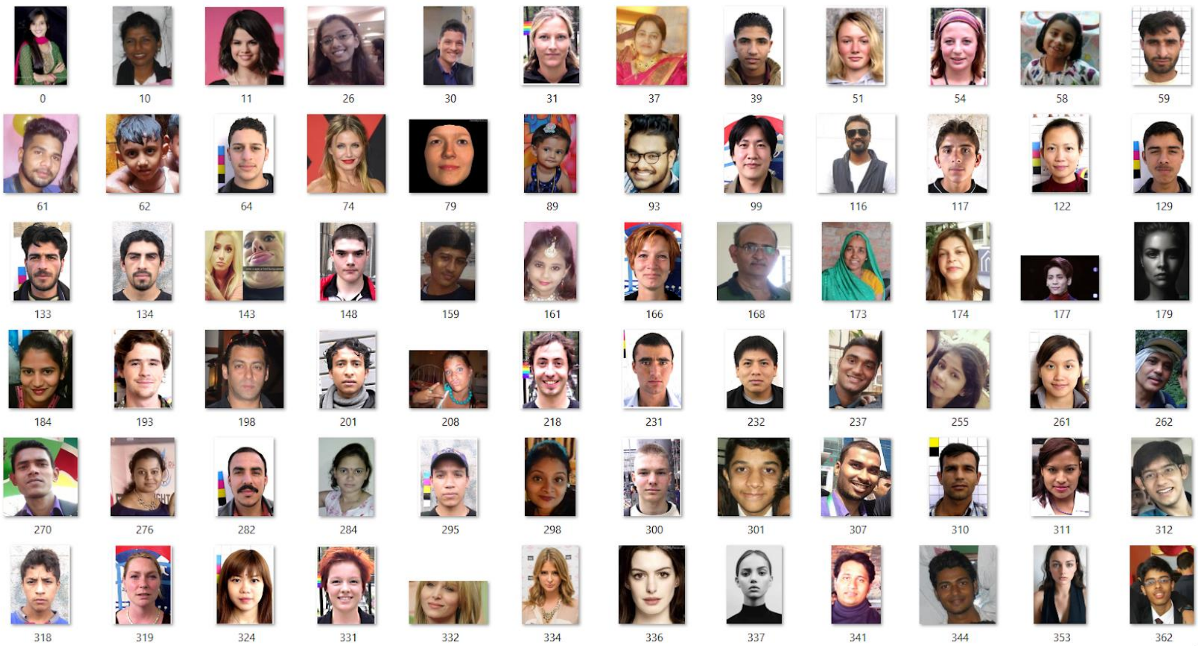
```
    col1,col3 = st.beta_columns([30,30])

    enhance_type = st.sidebar.radio("Enhance Type",["Original","Gray-Scale","Contrast","Brightness"])
    img = Image.open(image_file)
    if enhance_type == 'Gray-Scale':
        img = np.array(img.convert('RGB'))
        img = cv2.cvtColor(img,1)
        img = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
        img = Image.fromarray(img)
    elif enhance_type == 'Contrast':
        c_rate = st.sidebar.slider("Contrast",0.5,3.5)
        enhancer = ImageEnhance.Contrast(img)
        img = enhancer.enhance(c_rate)
    elif enhance_type == 'Brightness':
        c_rate = st.sidebar.slider("Brightness",0.5,3.5)
        enhancer = ImageEnhance.Brightness(img)
        img = enhancer.enhance(c_rate)

    img = img.convert('RGB')
    saved_img = img.save('./output/out.jpeg')
    with col1:
        st.text("Orignal Image")
        st.image(img,caption="Input", use_column_width=True)
    st.sidebar.write('Please wait for the magic to happen! This may take up to a minute.')
    st.sidebar.info('PRIVACY POLICY: Uploaded images are never saved or stored. They are held entirely within memory for prediction \
    and discarded after the final results are displayed. ')
    with col3:
        st.text("Predicted Image")
        st.image(image_detections(),caption="Output", use_column_width=True)
```

```
            face=np.expand_dims(face,axis=0)

            (withoutMask,mask)=model.predict(face)[0]
            #determine the class label and color we will use to draw the bounding box and text
            label='Mask' if mask>withoutMask else 'No Mask'
            color=(0,255,0) if label=='Mask' else (255,0,0)
            #include the probability in the label
            label="{}: {:.2f}%".format(label,max(mask,withoutMask)*100)
            #display the label and bounding boxes
            cv2.putText(image,label,(startX,startY-10),cv2.FONT_HERSHEY_SIMPLEX,0.45,color,2)
            cv2.rectangle(image,(startX,startY),(endX,endY),color,2)

    return image



def read_markdown_file(markdown_file):
    return Path(markdown_file).read_text()


# ------------------------- Selection From SideBar ------------------

app_mode = st.sidebar.selectbox(
    "Please select from the following", SIDEBAR_OPTIONS)

if app_mode == SIDEBAR_OPTION_WEBCAM:
    st.markdown('### Click on the below button to access the feature 👇')
    link = '[Detect!](https://share.streamlit.io/techyhoney/facemask_detection-webcam/webcam.py)'
    st.markdown(link, unsafe_allow_html=True)

elif app_mode == SIDEBAR_OPTION_UPLOAD_IMAGE:
    image_file = st.file_uploader("Upload Image", type=['png', 'jpeg', 'jpg', 'webp'])
    if image_file is not None:
```

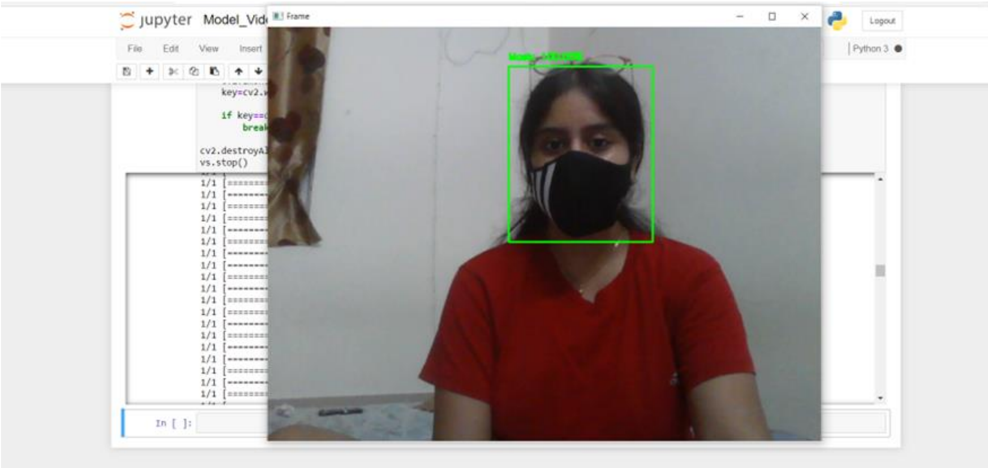
```
    directory = os.path.join(DEFAULT_DATA_BASE_DIR, IMAGE_DIR)
    photos = []
    for file in os.listdir(directory):
        filepath = os.path.join(directory, file)

        # Find all valid images
        if imghdr.what(filepath) is not None:
            photos.append(file)

    photos.sort()

    option = st.sidebar.selectbox('Please select a sample image, then click Detect button', photos)

    st.empty()
    st.sidebar.write('Please wait for the detection to happen!')
    pic = os.path.join(directory, option)
    with col1:
        st.text("Orignal Image")
        st.image(pic,caption="Input", use_column_width=True)
    with col3:
        st.text("Predicted Image")
        st.image(image_detections(pic),caption="Output", use_column_width=True)


elif app_mode == SIDEBAR_OPTION_PROJECT_INFO:
    st.sidebar.success("Project information showing on the right!")
    intro_markdown = read_markdown_file(os.path.join(DEFAULT_DATA_BASE_DIR,"README.md"))
    st.markdown(intro_markdown, unsafe_allow_html=True)
```

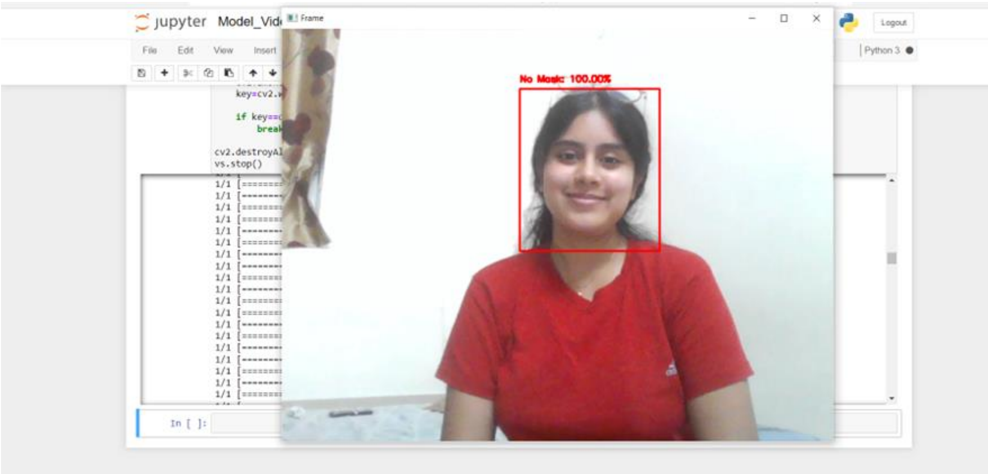
Raw Data With Mask:

Raw Data Without Mask:



Result:





CHAPTER 5

CONCLUSION

In the facial of an ongoing global health crisis, the development and implementation of a real-time facial mask detection system emerge as a vital step towards mitigating the risks associated with non-compliance with mask-wearing guidelines in public spaces. The problem at hand, characterized by a widespread failure to consistently adhere to mask-wearing protocols, poses a substantial threat to public health, particularly in densely populated indoor environments.

The significance of this project lies in its proactive approach to tackling the challenges posed by individuals who, whether due to a lack of awareness or unwillingness to comply, do not wear masks in public spaces. The real-time facialmask detection system, utilizing machine learning and OpenCV, serves as a technological sentinel against the potential transmission of respiratory illnesses, including the highly contagious Covid-19.

The project's primary objective was to provide authorities with a robust tool to identify non-compliance swiftly and accurately. The system's ability to detect individuals without masks in real time, coupled with its capacity to alert authorities, positions it as a valuable resource for enforcing mask-wearing guidelines. By doing so, the project not only addresses the immediate concern of virus transmission but also contributes to the broader goals of public health and safety.

The deployment of the real-time facial mask detection system aligns with the recognition that asymptomatic or presymptomatic individuals can unknowingly become vectors for the virus. In such cases, the timely identification of individuals without masks becomes crucial in preventing the further spread of respiratory illnesses within communities.

The successful development and implementation of this system offer a scalable and efficient solution for authorities tasked with enforcing mask-wearing guidelines. Its integration into public spaces can significantly reduce the burden on human resources while ensuring a consistent and unbiased approach to compliance enforcement.

However, it is essential to acknowledge that the real-time facialmask detection system is part of a broader strategy in addressing public health challenges. Education and awareness campaigns remain crucial components in fostering a collective understanding of the importance of mask-wearing. Additionally, the ethical implications of deploying such technologies warrant ongoing consideration, emphasizing the need for responsible use and protection of individual privacy.

In conclusion, the real-time facialmask detection system stands as a testament to the potential of technology to address pressing public health concerns. By providing a tool for authorities to efficiently identify and address instances of non-compliance, the project contributes to the global effort to curb the transmission of respiratory illnesses, safeguarding the well-being of communities. As we navigate the complex landscape of infectious diseases, the fusion of technology and public health measures becomes increasingly pivotal, and this project exemplifies a step in the right direction towards a safer and healthier future.

CHAPTER 6

FUTURE ENHANCEMENT

Real-time facial detection is a rapidly advancing field, and there are many potential areas for future enhancement of facial detection projects. Some of the possible future enhancements of real-time facial detection projects are as follows:

The algorithm will provide contactless facial authentication in settings such as community access, campus governance, and enterprise resumption. Our research has given the world more scientific and technological strength.

Improved accuracy: One of the most significant challenges in real-time facial detection is improving the accuracy of the system. Future enhancements could focus on improving the accuracy of the detection algorithm to ensure that the system can identify facials even in complex scenarios, such as low-light conditions or when the facial is partially obscured.

Real-time tracking: Real-time facial detection can be used for tracking people and monitoring their movements. Future enhancements could focus on developing systems that can track facials in real-time, allowing for more efficient surveillance and monitoring.

Multi-facial detection: Current real-time facial detection systems can detect only a single facial at a time. Future enhancements could focus on developing systems that can detect multiple facials simultaneously, which would be useful for monitoring crowds and large groups of people.

Integration with other technologies: Real-time facial detection could be integrated with other technologies, such as augmented reality or virtual reality, to create more immersive experiences. For example, facial recognition could be used to create personalized virtual avatars.

Privacy and security: As facial detection technology becomes more widespread, it is essential to ensure that user privacy and security are protected. Future enhancements could focus on developing systems that are more secure and protect user privacy by incorporating privacy features, such as blurring facials or storing data locally.

In conclusion, the future of real-time facial detection is exciting and full of potential for enhancing the accuracy, tracking, multi-facial detection, integration with other technologies, and privacy and security of the system. With ongoing advancements in computer vision and artificial intelligence, we can expect to see significant improvements in this field in the coming years.

CHAPTER 7

REFERENCES

[1]M. S. Ejaz and M. R. Islam, "Masked facial Recognition Using Convolutional Neural Network," 2019 International Conference on Sustainable Technologies for Industry 4.0 (STI), 2019, pp. 1-6, doi: 10.1109/STI47673.2019.9068044.

[2] M. R. Bhuiyan, S. A. Khushbu, and M. S. Islam, "A Deep Learning Based Assistive System to Classify COVID-19 facial Mask for Human Safety with YOLOv3," 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)

[1] M. M. Rahman, M. M. H. Manik, M. M. Islam, S. Mahmud, and J. -H. Kim, "An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network," 2020 IEEE International IOT, Electronics, and Mechatronics Conference (AIMTRONICS), 2020

[1] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning facial representation by joint identification-verification," in Advances in neural information processing systems, 2014, pp. 198hy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," 2014.

[4] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human facial identification," in Applications of Computer Vision, 1994., Proceedings of the Second IEEE Workshop on, pp. 138–142, IEEE, 1994.

[5] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning facial representation from scratch," CoRR abs/1411.7923, 2014.

[6] X. Cao, D. Wipf, F. Wen, G. Duan, and J. Sun, "A practical transfer learning algorithm for facial verification," in Computer Vision (ICCV), 2013 IEEE International Conference on, pp. 3208–3215, IE8–1996.

[2] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "facial recognition: A literature survey," ACM computing surveys (CSUR), vol. 35, no. 4, pp. 399–458, 2003.

[3] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. KarpatEE, 2013.

[7] P. N. Belhumeur, J. P. Hespanha, and D. Kriegman, “Eigenfacials vs. fisher facial Recognition using class specific linear projection,” Pattern Analysis and Machine Intelligence, IEEE Transactions on 19(7), pp. 711– 720, 1997.

[8] X. Cao, D. Wipf, F. Wen, G. Duan, and J. Sun, “A practical transfer learning algorithm for facial verification,” in Computer Vision (ICCV), 2013 IEEE International Conference on, pp. 3208–3215, IEEE, 2013.

[9] Y. Sun, X. Wang, and X. Tang. Deep learning facial representation by joint identification-verification. CoRR, abs/1406.4773, 2014. 1, 2, 3

[10] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. Nature, 1986. 2, 4

PAPER PUBLICATION STATUS

Submitted to a conference and waiting for approval

Name of the conference -

2nd International Conference on Higher Education Institutes' Challenges & Solutions for Sustainable Development Goals, Nov’2, 2023.

Proof of registration-

SRM

SWAGAT KALITA (RA2011027010092) <sk5749@srmist.edu.in>

Fwd: Registration Confirmation - SDG 3 Good Health and Wellbeing

1 message

Dhivya K <dhivyak@srmist.edu.in>
Bcc: sk5749@srmist.edu.in

Fri, Oct 27, 2023 at 2:56 PM

Dear Sir/Madam,

Greetings from School of Public Health, SRM Institute of Science and Technology, Kattankulathur.

Thank you very much for registering yourself for **SDG 3 – Good Health and Wellbeing, 2nd International Conference on HIGHER EDUCATION INSTITUTES' CHALLENGES & SOLUTIONS FOR SUSTAINABLE DEVELOPMENT GOALS ’23** scheduled on 01-03 November 2023.

Looking forward for your active participation

Best Regards

Dhivya. k, Geetha Veliah
Co-Ordinators, SDG 3,
2nd International Conference on HIGHER EDUCATION INSTITUTES' CHALLENGES & SOLUTIONS FOR SUSTAINABLE DEVELOPMENT GOALS ’23
SRM Institute of Science and Technology.

SRM

SWAGAT KALITA (RA2011027010092) <sk5749@srmist.edu.in>

ICSDG # 3- Keynote presentation and Special Address information - Day 3

1 message

Dhivya K <dhivyak@srmist.edu.in>
Bcc: sk5749@srmist.edu.in

Fri, Nov 3, 2023 at 8:30 AM

Dear All

Greetings from the School of Public Health.
As part of the 2nd International Conference on Higher Education Institute Challenges Solutions for Sustainable Development Goals 2023 (ICSDG 2023) - **SDG#3 - Good Health and Well being**, we have two exclusive presentations:

Special Address by Dr. Usha Sriram, Renowned endocrinologist and diabetologist with interest in women's health and NCDs.
Date: 03rd November 2023
Time: 10:00 to 11.00 AM
Topic: The pandemic after the Pandemic :NCDs
Online Mode - https://meet.google.com/oay-riaq-kyi

Keynote presentation by Dr. Sujatha, R, Senior SDG consultant, Planning and Development,Government of Tamil Nadu.
Date: 01st November 2023
Time: 11:30 to 12.30 PM
Topic: State of SDGs in the State (Tamil Nadu)
Online Mode - https://meet.google.com/qmg-qzys-dmc

Join us to know more about SDG #3.

Thanks,
Dhivya.k
Geetha Veliah
ICSDG #3 Co-Convenors