Identity Recognition

A Project Report

Presented to

The Faculty of the Computer Engineering Department

San Jose State University

In Partial Fulfillment

Of the Requirements for the Degree

Bachelor of Science in Software Engineering

By

Dean Guardanapo, Hewan Mekuria, Madelyne Taligato, Hieu Vo

11/2021

APPROVED FOR THE COLLEGE OF ENGINEERING

Kaikai Liu, Project Advisor

Dr. Wencen Wu, Instructor

Dr. Rod Fatoohi, Computer Engineering Department Chair

ABSTRACT

Identity Recognition

By Dean Guardanapo, Hewan Mekuria, Madelyne Taligato, Hieu Vo

Identity recognition software and products are becoming ever-present around the globe today. In its current technological state, facial recognition software allows for the human face to be detected and matched from either a digital frame or a video frame that coexists with a database. With the growing need for facial recognition software worldwide, advanced platforms will be developed to increase the accuracy and functionality of these products.

One of the leading challenges in the Computer Vision (CV) and Machine Learning (ML) industry for identity recognition software is being able to identify individuals with a mask on. Although there are current products that are in the market for facial recognition, they generally do not contain the advancements to successfully recognize humans with a face mask on.

In this report, an investigation on the methods to recognize/identify individuals with a face mask on is presented in depth. These abstractions provide AI and ML developers with information on how to manipulate existing facial recognition software to increase the capabilities of the program to identify humans who are wearing face masks. In conclusion, these advancements will hopefully allow for facial recognition software platforms to identify individuals with a face mask on or allow developers to further advance the technology.

Acknowledgments

The Identity Recognition team would like to show appreciation to our project advisor Kaikai Liu for being our guide throughout the project. Professor Liu provided the team with great advice and knowledge.

In addition to that, we would like to thank the Software Engineering department at SJSU for allowing us to represent ourselves through this project.

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction (Madelyne Taligato, Hieu Vo)

Facial recognition software takes in the spatial geometry of distinguishing facial features including the cheekbones. eye-bones, lips, eyes. jaw-line and more. There are different methods that can be used for facial recognition, however, every single method is needed to focus on the facial features regardless. This application will focus on capturing facial features by a camera from a distance x away from the user. The goal of this software is to attempt to recognize an individual *even when they are wearing a mask/face covering*.

Traditionally, biometric facial recognition or authentication systems have high levels of accuracy by training on a large amount of facial data. As stated in the article "Biometrics - A Look at Facial Recognition" by Woodward, Horn, Gatune, and Thomas, "Facial recognition is able to leverage existing databases in many cases. For example, there are high-quality mugshots of criminals readily available to law enforcement. Similarly, facial recognition is often able to leverage existing surveillance systems such as surveillance cameras or closed-circuit television (CCTV)." As an example, these traditionally facial recognition systems use the following steps for their process:

1. A captured image of an individual's face will need to be provided by either evaluating a pre-existing image of the person or a new image that must be captured via a camera.
2. Next, the program will come into effect to detect and analyze the pre-existing or the newly captured image. Analyzing the image consists of locating the face from the image based on landmark patterns that define a face. For example, facial features such as cheekbones, lips, eyebrows, and jawline will be used to verify a user's face.
3. After the software has recognized that there is a face in the image, the software can begin to further analyze the image by taking the spatial geometry of the individual's facial features. The method that will be used in this software is the

Principle Component Analysis. This is defined in the article "Biometrics - A Look at Facial Recognition" as follows, "The most popular method is called Principal Components Analysis (PCA), which is commonly referred to as the eigenface method. PCA has also been combined with neural networks and local feature analysis in efforts to enhance its performance. Template generation is the result of the feature extraction process. A template is a reduced set of data that represents the unique features of an enrollee's face. It is important to note that because the systems use spatial geometry to distinguish facial features."

4. This step is to compare the facial features that have been extracted from the user's pre-existing image or newly taken image and compare them with those in the verified dataset. This process will allow for the software to identify the user and give a score indicating whether or not there is a verified match within the dataset.
5. The last step of this process is for the software to take the identification score that was acquired from the user's image compared throughout the dataset to determine if there is a match for the user or not. This match is declared for both users wearing and not wearing a face mask.

1.1 **Project Goals and Objectives** (Madelyne Taligato, Hieu Vo)

Following are the project goals and objectives.

Goals	Objectives
Accurate face recognition/authentication program	Trained program with a widely variable database including faces of different ages, emotional states, etc.
Identity Accuracy	Program can accurately identify individuals regardless if there is a face mask covering or not.

Table 1.100 Goals and Objectives

1.2 Problem and Motivation (Madelyne Taligato)

Currently, there are many facial recognition software systems that are used in different areas of government and businesses. Facial recognition software is used to increase security at airports or border crossings, help identify child victims of abuse or even identify suspects in addition to other uses in phone applications for biometric authentication. However, there are issues facing existing recognition software systems including the inability to recognize an individual without the whole facial ratio. This project focuses on working on a system that will have the potential to recognize an individual by only using the upper region of a person's face. While currently, we are in the process of understanding all the internal workings of existing facial recognition software, we plan to incorporate all the necessary features into our new project. Furthermore, we plan to improve inclusivity by diversifying the data in the dataset since the existing systems have shown bias in the dataset and sometimes result in mismatch which can result in a significant impact on a person's life.

1.3 Project Application and Impact (Hieu Vo)

The results and knowledge gained behind this project can be applied to current implementation and systems for general facial recognition or biometric authentication. Our initial goal with this project was to explore its use in biometric authentication with face coverings due to COVID-19. However, we acknowledge that there are potential negative consequences behind this project including its potential application in surveillance at any level; casual or even government surveillance. However, there also exists broader positive applications of this project outside of authentication. One example is being able to locate a lost individual with face coverings on; Perhaps they were kidnapped and the kidnappers had facial coverings on them.

1.4 **Project Results and Deliverables** (Hieu Vo)

The end goal of the project is to have a program with great accuracy in identifying individuals based on the upper region landmarks of the face. The system's main intention is to be able to recognize and identify individuals regardless of whether they have a face mask/covering on or not. The program will be built using a modification of existing libraries such as Yolov3 and Yolov5 along with a dataset for training to improve accuracy.

1.5 **Project Report Structure** (Madelyne Taligato)

Chapter 2 Background and Related Work:

- 2.1 Background and Used Technologies: Discusses background on identity recognition, including other technologies, programs, and knowledge used for the project.
- 2.2 Literature Search: Discusses published works and literature about identity recognition technology.
- 2.3 State-of-the-art Summary: Describes the current state of identity recognition technology, including current leading technologies, research, and current challenges and issues being addressed.

Chapter 3 Project Requirements:

- 3.1 Domain and Business Requirements: Activity, domain class, and state machine diagrams to show processes, domain, and key business classes of the project system.
- 3.2 System (or Component) Functional Requirements: Statements of what the system should do.

- 3.3 Non-functional Requirements: Statements describing the measurable requirements on the system that include the performance, capacity, availability, compliance to standards, security, etc.
- 3.4 Context and Interface Requirements: Specify the environments supporting the project and the interface for the components and system.
- 3.5 Technology and Resource Requirements: Requirements for hardware and software technologies for the project.

Chapter 4 System Design:

- 4.1 Architecture Design: Architectural design for the project system the team has created.
- 4.2 Interface and Component Design: Interface and component design for the project system the team has created.
- 4.3 Structure and Logic Design: Structural and logical design of the components and processes of the project, including descriptions of how the components interact with one another for the system logic.
- 4.4 Design Constraints, Problems, Trade-offs, and Solutions:
    - 4.4.1 Design Constraints and Challenges: Constraints and challenges on the project system, including economic, resources, environment, hardware, software, safety, reliability, etc.
    - 4.4.2 Design Solutions and Trade-offs: Solutions and/or trade-offs to the design constraints and challenges presented in the previous section.

Chapter 5 System Implementation:

- 5.1 Implementation Overview: Describes the group's implementation, including scope, used platform and language, dependent software, and implementation dependencies.
- 5.2 Implementation of Developed Solutions: Describes the implementation of the improvement on identity recognition accommodating face masks, including techniques, methods, and algorithms used.

5.3 Implementation Problems, Challenges, and Lessons Learned: Describes major implementation problems, challenges encountered while developing, and lessons learned from these encounters.

Chapter 2 Background and Related Work

2.1 Background and Used Technologies (Dean Guardanapo)

2.1.1 Background

Our application, Identity Recognition, is similar to most facial recognition systems in addition to one new feature. The new feature is believed to be an upgrade and noticeable improvement over most facial recognition software because most of these current applications are not able to recognize individuals with a face mask on. Our application will hopefully provide an advanced feature that will allow for users to be identified/recognized while wearing a facemask. With the help of utilizing a trained database, the users are able to achieve a more accurate identity reading which will allow individuals to save time by not having to remove their facemask to be recognized by the software.

2.1.2 Technology

Facial recognition software is built around many concepts and pre-existing applications. One of the first major concepts that need to be understood for the development of this application is biometrics. According to an article on biometrics, what biological measurements are considered to be biometrics depends on the properties of:

- **Universality**- which means that each individual should have a characteristic
- **Uniqueness**- no two personas should have the same characteristics
- **Permanence**- the characteristic should be unchanging over time
- **Collectability**- the characteristic can be measured quantitatively
- **Performance**- refers to the resource requirements in order to achieve identification accuracy

- **Acceptability**- the extent to which the biometric system is accepted by the population
- **Circumvention**- refers to how easy it is to manipulate the system.

Even though no single biometrics is known to be 100% effective in meeting the expectations of all identification and authentication applications, each biometrics has its strength and weakness. According to the same article, it is also stated that Biometrics Can be done using different characteristics including voice, face, fingerprints, iris, ear, and so on (Annil, Ruud et al). However, for our project, we depend on the identification of an individual based on the face.

Biometrics is essentially what allows for a person to be uniquely identified and authenticated based on verifiable data. The verifiable data, in this case, focuses on a preexisting image of the individual. Furthermore, the biometric authentication will compare the data from the unique individual's biometric "template" (their facial features in this case) to the preexisting image to determine the resemblance.

Building on the previous technology of biometrics, machine learning goes hand in hand with this. The facial recognition software will use deep learning algorithms to compare the captured biometric image of the unique individual's face and compare it to the stored image in the database to verify the individual's identity. It has also been found that face recognition techniques that are based on machine learning models exceeded that of the traditional methods (Sharma and Kumar).

The main reason that machine learning needs to be understood is that the software will use image processing (a form of machine learning) to break down the image and identify that the image does indeed possess a face. The machine learning algorithm will consist of recognizing the height/width of the face, the color of the face, and the width/height of other parts of the facial landmarks including but not limited to lips, nose, cheekbones, eyebrows, and more.

Artificial intelligence may be one of the most important concepts to understand when it comes to facial recognition software. More specifically, deep vision AI is what is most widely used when it comes to the development of such applications. Deep Vision AI allows developers to use a “plug and play” platform which allows for real-time updates and quick responses based on the cameras being used to capture the biometric images of the face.

2.2 Literature Search (Dean Guardanapo)

“Facial Recognition Software with Python” - Philipp Wagner

The article “Face Recognition with Python” by Philipp Wagner provides excellent documentation for a successful facial recognition software using Python.

- This article provides a step-by-step process to explain the fundamentals of facial recognition software using the Python programming language to utilize the Eigenfaces method. This method does not only take in the facial features but also illuminates the faces as shown in Figure.2.002 taken from the article.

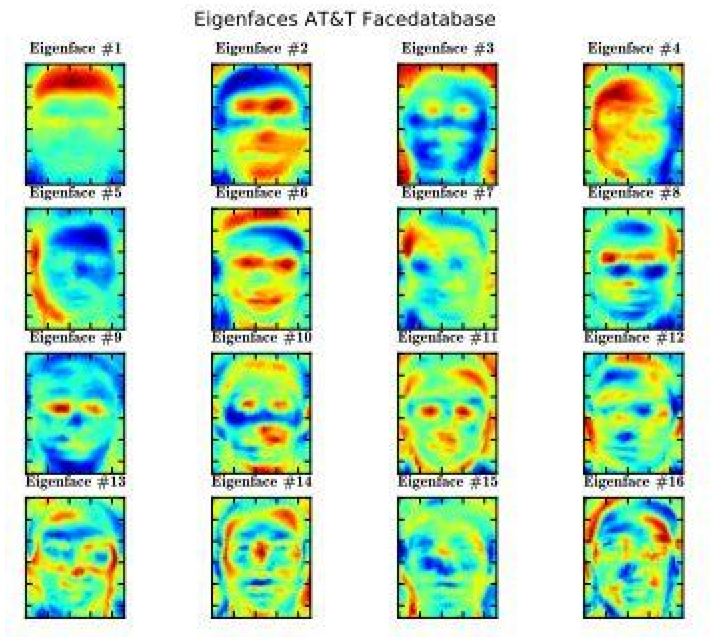


Figure.2.002 Eigenfaces AT&T face database

Referencing Chapter 1 under the detailed description, Principal Component Analysis is used here for Eigenfaces methods. This model works well to successfully complete facial recognition software because it is object-oriented and so is Python. In addition to that, Python has an "EigenfaceModel" class that contains a promising algorithm to be utilized for the completion of the project.

Development of Real-Time Face Recognition

OpenCV has been proven to be used in successful facial recognition software deployments. The article "Development of Real-Time Face Recognition System Using OpenCV" in the IRJET Journal provides documentation of a successful project.

This article demonstrates how the OpenCV library can be used to determine the identity of an individual. For example, the article shows how it is used through a script of code. It is stated, "Here in case of testing, whenever we click on the test button it will take the test image by using a webcam. Then for the following code will run for recognition of people coming in front of the camera:

```
def test_cap_Image(self):

i=1

camera=cv2.VideoCapture(0)

while(i<2):

if not os.path.exists('./test-images'):

print 'hello'

os.mkdir('./test-images')

name=self.lineEdit.text()

if not os.path.exists('./test-images/'):

os.mkdir('./test-images/' + name)

ret, image=self.camera.read()

imagename='./test-images/' + str(i) + '.jpg'

print imagename

## cv2.imshow('test',image)

cv2.imwrite(str(imagename),image)
```

After clicking on the test button, the system will recognize the person's name with confidence. Otherwise displayed as an Unknown Person." In addition to this, the article further goes on to prove the proficiency of facial recognition using OpenCV.

2.3 State-of-the-art Summary (Dean Guardanapo)

Facial recognition software has grown throughout the years since its creation. The best measure of the current state of any facial recognition technology would be its accuracy and error rate. Most, if not all, of the leading brands that are involved in facial recognition technology, have improved over the years, reaching around 98% accuracy (Anderson, 2020). Some leading technologies have higher accuracy rates than facial recognition by humans. However, the current situation of the world has exposed a possible issue with facial recognition technology, and that is the accommodation of face masks.

Network Connect, or NEC, has taken the lead in improving facial recognition technology that can accommodate face masks. NEC, a Japanese company, claims that its software can verify the identity of individuals wearing face masks with an accuracy rate of more than 99.9% (NEC, 2020). NEC's software is an amazing improvement to facial recognition software, where the accuracy of identification with face masks before the pandemic was roughly 20% - 50%. In a post on NEC's website, a breakdown of the process the software goes through to verify the identity of an individual is shown (NEC, 2020). The software would first check if the individual is wearing a mask, then there are two different facial reconciliation processes that would identify the individual based on the presence of a mask. NEC has made a huge breakthrough in facial recognition technology. However, compatibility with their software is limited, with only three of

NEC products being compatible, two out of the three products being in the Japanese market only.

Facial recognition is used in many smartphones and devices, leading to the need to remove your mask to gain access to those devices. Currently, there is no safe way to go through Face ID or phone facial recognition biometrics without removing your mask.

Chapter 3 Project Requirements

According to IEEE standard 729, a requirement is defined as a condition or capability needed by a user to solve a problem or achieve an objective, a capability that must be met or possessed by a system to satisfy a contract, standard, or is a documented representation of a condition or capability. As such, our facial recognition system has the following 5 requirements as a system. Each requirement with a detailed explanation is as follows.

3.1 Domain and Business Requirements (Hewan Mekuria)

Domain requirements are the requirements that involve the character of a particular domain of a project.

R1. Authentication- This authentication feature lets biometric systems use this technology to assist in authenticating users. For example for systems such as Apple faceID.

R2. Payments- Facial recognition is used in electronic payment methods such as Apple pay, Samsung pay, etc. Digital identification methods increase security and privacy in the financial services sector.

R3. Photo tagging- social media sites such as Facebook use facial recognition to identify the users and in photo tagging.

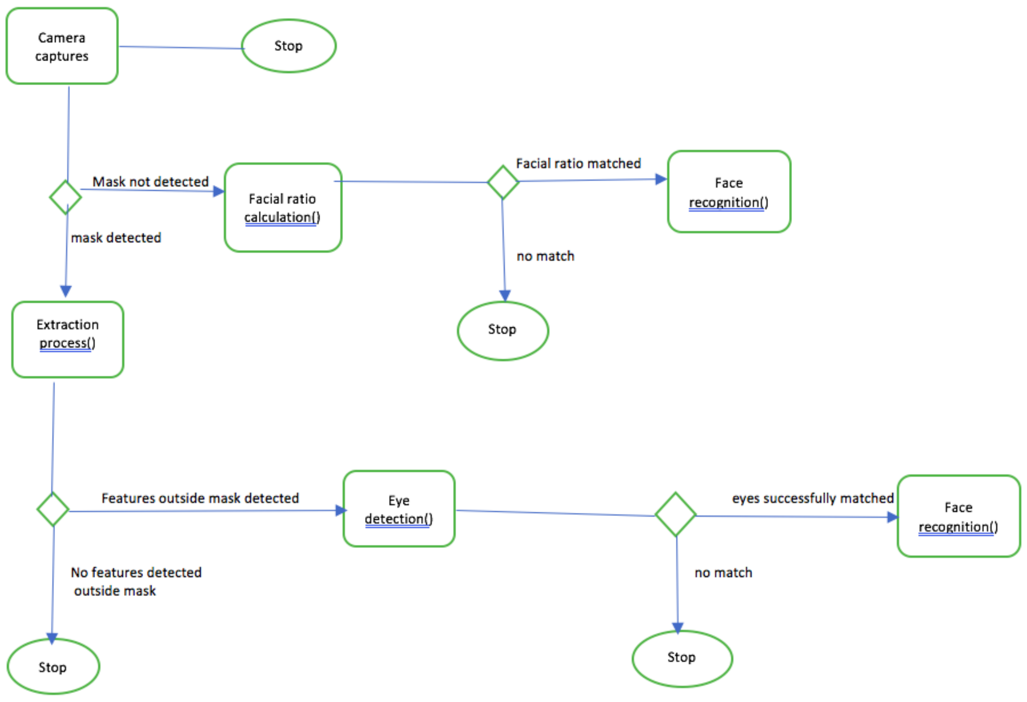


Figure.3.001 State Machine Diagram

3.2 System (or Component) Functional Requirements (Hieu Vo, Hewan Mekuria)

Functional requirements are requirements that the user demands as a basic facility that the system should offer. For this system, we will divide the functional requirements into two separate groups. The first is the dataset and the second is the recognition/identification program

Dataset

R1- The system shall contain an inclusive dataset to avoid bias, such as multiple skin colors, sexes, ages, etc.

R2- The dataset shall not use the same image used in the training and testing phases.

R3- The testing dataset shall only contain facial images with masks

Program/System

R1- the system shall be able to detect faces in images.

R2- the system shall be able to extract each face's region of interest

R3- The system shall be trained on the upper face region

3.3 Non-functional Requirements (Hewan Mekuria)

Non-functional/non-behavioral requirements constitute the quality constraints that the system must satisfy according to the project contract.

R1. capacity -The system must be portable and can be applied to devices with limited computational capacity.

R2. performance- the system output response must be reasonably fast and all other features must execute instantaneously.

R3. availability - the system must be available on different devices and also on the internet in order for users to download and access it at any time

R4. usability- The system should be platform-independent, and users don't need to read the user manual to operate the system. Simply following along the instructions and responding to modal dialogues on the UI should be enough to fully operate the system.

3.4 Context and Interface Requirements (Hewan Mekuria)

R1. The system must be able to detect all faces present in the frame of the system.
R2. The system must be able to ignore the background and focus on the facial marks.
R3. The user should not move their face out of the frame in order to get accurate results.

3.5 Technology and Resource Requirements (Hewan Mekuria, Hieu Vo)

R1. The system will be implemented using python script /jupyter notebook.
R2. The system will be operable on desktop platforms.
R3. There should not be any RAM or storage limitations to use the system. A working web on any eligible operating system should be enough.

Chapter 4 System Design (Dean Guardanapo)

The identity recognition application is implemented utilizing the Python programming language in the Yolov3 and Yolov5 libraries to access real-time computer vision algorithms and classes. The user interface of this application will be similar to a regular camera function. The application will start with an open camera and consist of a red or green box that verifies whether or not a human face is recognized (green indicates recognized and red indicates not recognized) in addition to the percent accurate tagged in for each recognized individual.
This application does not have many user functions available to those who are using the application but rather just to be able to be identified even with a face mask on. The

algorithm embedded in the application will be able to identify individuals regardless of they are wearing a face mask or not

4.1 Architecture Design (Dean Guardanapo)

UML Sequence Diagram

- The image below is a design pattern in the form of a UML sequence diagram. Analyzing the pattern will describe how the facial recognition software will work. The user interacts with the user interface (whether there is an actual interface or it is just running the software) and/or upload an image into the database, the software will then open the camera which performs a biometric face scan and put it into the database, the database then responds back to the software with both images where machine learning will be used to compare the images for facial matches, and the final step will be the software giving a match message back to the user.

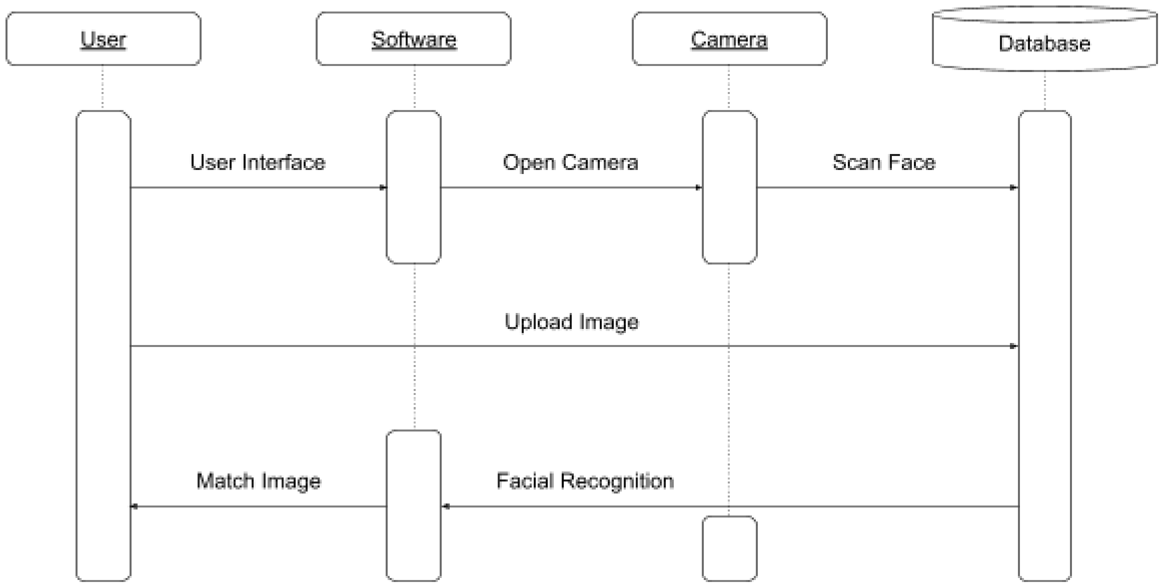


Figure.4.001 UML sequence diagram

Facial Recognition Architecture

- Facial recognition software has simple yet unique architecture as shown in Figure.4.002. This design and implementation of our facial recognition architecture have four extremely important components.

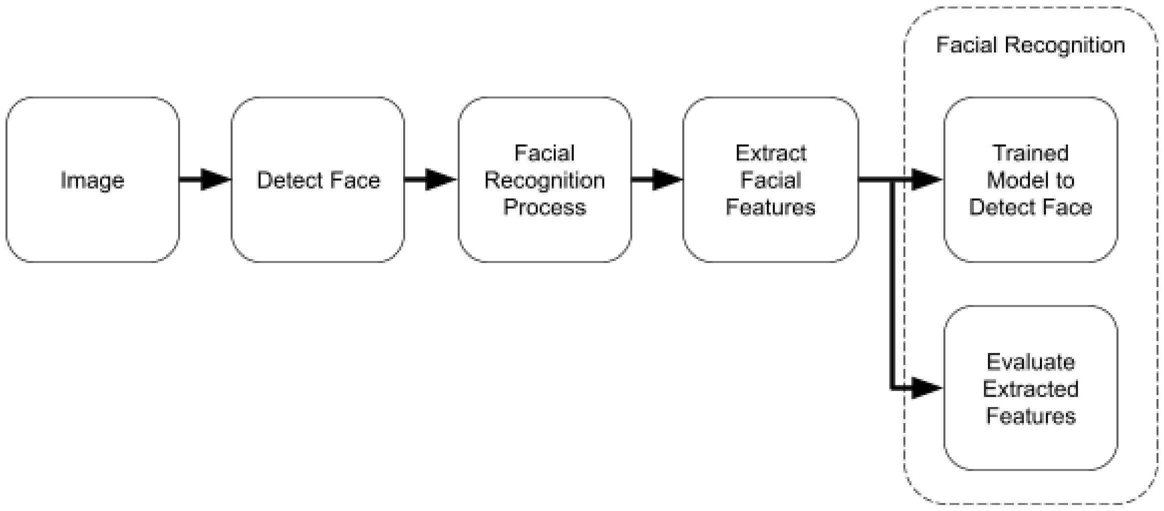


Image
Detect Face
Facial Recognition Process
Extract Facial Features
Facial Recognition
Trained Model to Detect Face
Evaluate Extracted Features

Figure.4.002 Facial recognition architecture

The first component is the "Detect Face" block. This component is the first stage of our facial recognition software and allows for the system to perform real-time recognition. A captured image of an individual's face will need to be provided by either evaluating a pre-existing image of the person or a new image must be captured via a camera. This step must be precise because it moves us into the next component which is arguably the most important.

The next step is the “Facial Recognition Process” which is going to be used to determine the accuracy of the software. The software will come into effect to detect and analyze the pre-existing or newly captured image. Analyzing the image consists of locating the face from the image based on geometric patterns that define a face. The techniques that will be used in this step will be modified pre-existing facial recognition algorithms to better identify an individual with a mask on. This is done here by removing the portion of the face from the nose down. Through this, the algorithm can attempt to do facial recognition using the data input around the eyes and the forehead of a person. Therefore, having a large and accurately trained dataset model will be important here. The challenge with the data set is that there are only a limited number of pre-existing datasets to reference from. Therefore this will potentially necessitate creating our own manually inputted data.

Moving forward, we have the “Extract Facial Features” component. This component will be once again determined by the facial recognition algorithm. This is to compare the facial features that have been extracted from the user’s pre-existing image or newly taken image and compared with those in the verified dataset. This process will allow for the software to identify the user and give a score indicating whether or not there is a verified match within the dataset.

Once all of these components have been successfully completed, the architecture flows into the “Facial Recognition” block which consists of two stages. The first stage, “Trained Model to Detect Face”, is where the algorithm analyzes the dataset to understand the distinct differences between an individual's face with and without a mask on. The second stage, “Evaluate Extracted Features”, is where the evaluation process will take place and compare the “new” face to other individuals within the trained dataset in order to determine the output of the program. The output will be able to give the user a message letting them if they are detected with and/or without a face mask.

4.2 Interface and Component Design (Madelyne Taligato)

The following component diagram shows which component of the program interacts with which component. The input image would interact with the Face Detection component. The Face Recognition component would interact with the Face Detection component and the Database. The Face Recognition component would then send back the identification of the individual in the inputted image.

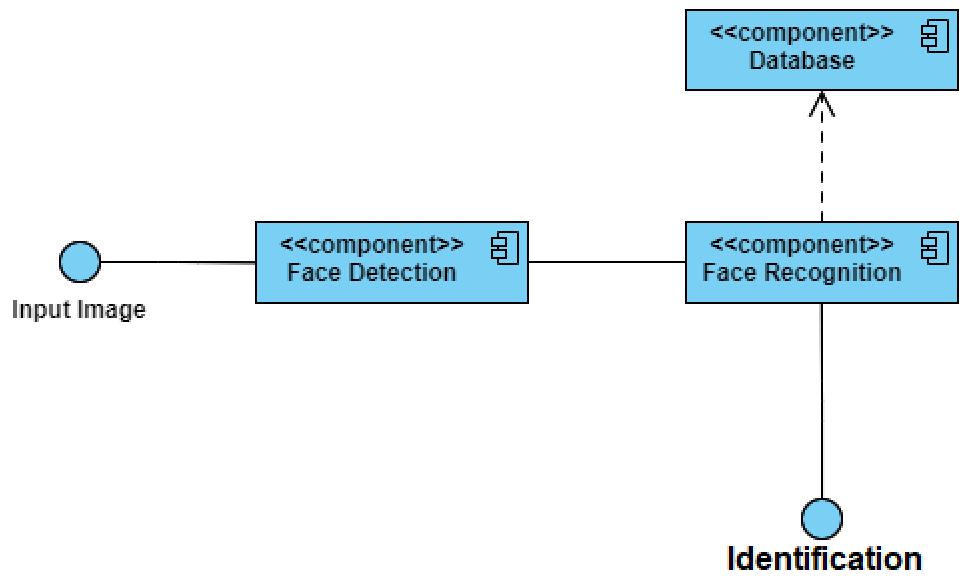


Figure.4.003 Component Diagram

4.3 Structure and Logic Design (Madelyne Taligato)

Figure.4.003 shown above shows which components interact with each other. The process of identifying the face in the input image first starts with the Face Detection component. The Face Detection component would process the image, identifying where in the image the face, regardless of if a face mask is on or not, is located and boxing the face in a rectangle. The processed image is then sent to the Face Recognition component. The Face Recognition component would then reference the database to find a face that matches the face on the image. The Face Recognition component would return the info on the identified face.

4.4 Design Constraints, Problems, Trade-offs, and Solutions

4.4.1 Design Constraints and Challenges (Dean Guardanapo)

Design constraints we encountered while designing this project were finding a free to use the database to train the model, the accuracy of the software in identifying an individual with a face mask, what facial features the software should focus on if a face mask is detected, and how to identify the individual based on those facial features not hidden by the face mask.

Some challenges we also faced were time constraints for each team member as well as learning how to use new software we have not used before to test mask detection. Our team had limited options in the knowledge of machine learning and artificial intelligence programming structures needed to build the facial recognition software. Due to this challenge/constraint, our team was not able to fully develop a far more advanced application and fulfill all of our prototype deliverables.

4.4.2 Design Solutions and Trade-offs (Dean Guardanapo)

According to Harvard.edu, facial recognition software algorithms claim a high classification accuracy however, these are not entirely true. A number of researchers have found significant bias and error rates in existing recognition software. As a result of our recognition software, we plan to spend a tremendous amount of time improving the algorithm and increasing the accuracy rate rather than having an appealing user interface. Even though the software will have an easy navigation tool and reasonably engaging and appealing front end for the users, the main effort will be put into developing a better algorithm that enhances recognition ability and lessens inaccuracies from existing recognition software systems. Solutions to reduce the bias in our algorithms include modifying testing cohorts and improving data collection, creating more diverse datasets, and establishing standardized image quality.

One solution that our team was able to come up with was utilizing open-source code to our advantage to learn and build our skills. Being able to successfully utilize these codes, we were able to learn enough to increase the functionality and accuracy of the current code.

Chapter 5 System Implementation

5.1 Implementation Overview (Hieu Vo)

This program is implemented within Jupyter Notebook and is meant to be run under Google's Colaboratory environment. The user either uploads the demonstrated jupyter notebook file onto Google Colaboratory or modify files. The user then uploads an image of the user's face or the default file. This default file is an image of a person that was used to train on the model.

Once uploaded, the program checks if the image of the person has a mask on or not. If there is no mask, the program then proceeds with regular facial recognition. If however the program detects that the image has a mask, it will attempt to identify the user based on a previous trained model of their entire face with the upper region of their face.

Once the above steps are completed, the program will output if it is the identified user or someone else.

5.2 Implementation of Developed Solutions (Hieu Vo)

The current implementation is a jupyter notebook through colab in order to utilize free resources provided by Google. The initial step is to prepare a training dataset for the model. We will use a jupyter notebook script that takes an input of a video of a person's face unmasked. This script will parse the video into separate frames and will detect labels using Local Binary Histograms built into OpenCV. This script can then either label where the face is in the frame, label only the top portion of person's face, or even take in the eye region of the face. Another prepared dataset containing masks will be used to train a mask detector.

Once prepared, the dataset(s) are trained through a machine learning algorithm called YOLO or You Look Only Once. Once the models are trained, the models will be loaded to a detector notebook. This notebook will use a user provided image or a default image of the person that was trained upon.

If the detector sees that a person does not have a mask, it will run the image through a traditional face recognition algorithm built into OpenCV or use a Yolo trained model. If however the person does have a mask on, it will attempt to identify the person based on the upper region of their face.

5.3 Implementation Problems, Challenges, and Lesson Learned (Hieu Vo)

The main challenge related to implementation is understanding how the original facial recognition/identification algorithms worked. Once this was done we were able to modify it and create our own algorithms for the project. Outside of this, most challenges with implementation were simply understanding the material in order to implement it.

There were several challenges with the implementation. The initial challenge was understanding how facial recognition and identification worked to reimplement it for our purposes. Once this was done we stumbled upon the issue of labeling the data of a person's face that we wanted to train on. This resulted in needing to create a specialized script in order to take in a video input and parse the video into separate frames that can be used to train. This also necessitated using openCV's facial detector in order to determine where the face was in the frames.

This itself also raised another challenge on converting OpenCV's face detection label boxes into Yolo Format which required extended research. Once this was figured out, we were able to modify the labeling data to capture specific portions of the user's face.

The biggest challenge with this project was working in the limits of Google's Colaboratory environment. Google places many restrictions on the free tier of their service which makes it ***extremely*** difficult to train the models. Initially we had multiple labels on a single model to use to identify a person. However this would end up necessitating an estimated 48 hours to train the model. Colab's free tier meant that we had to keep the session active and stay active every 5 minutes or so or else Google would shut down the instance.

Therefore it was necessary to change this into several separate models with single labels for each. This would then reduce the amount of training time for each model although it was still difficult.

Contributing to the challenge was that we had to investigate multiple ways of recognizing a person underneath the mask therefore increasing the amount of time to train. However it was realized that despite our efforts, the model's accuracy was not accurate enough. This is an extremely difficult problem to solve for the industry; recognizing a person underneath a face mask.

We theorize that potentially adding other hardware data inputs such as depth detection might help with recognizing a person or possibly cheat by using a face mask as part of their identity. This is similar how we observe another person based on how they look such as the clothes they wear and that style of clothes.

Chapter 6 Tools and Standards

6.1. Tools Used (Hewan Mekuria and Dean Guardanapo)

For this project we have used the following tools:

- **Github**
  - open-source version control used for real-time collaboration
- **Google Colaboratory**
  - Used to write and execute python code
- **Tensorflow with GPU**
  - open-source platform for machine learning
- **Python**
  - object oriented programming language
- **Yolov3**
  - Facial recognition
- **Yolov5**
  - object oriented architectures and models pre-trained on the COCO dataset
  - Mask detection
- **OpenCV**
  - Frame script parsing and traditional facial recognition.
- **Visual Studio Code**
  - source code editor

6.2. Standards

- Hardware/Software components

    - The system is required to work on both Mac OS and Windows operating systems. There are also no limitations on browser selection and users can use any browser such as chrome, firefox, or safari to use this system.

- Requirements
    - No user account is required to use the system.
    - A valid operating system and a web browser are required.

- Design
    - The system is designed to identify if an individual is wearing a mask or not.
    - Design based upon an explicit understanding of users, tasks, and environments
    - The design addresses the whole user experience

- Interface
    - The interface is to be easy to use and navigate between pages

- Accessibility standard
    - The system should respond to different changes such as changes in the size, brightness, and orientation of the browser,
    - The system should display a language readable to the user.
    - The system should be available to be used by a screen reader.

- Usability standard
    - Pages (if multiple) should have consistent components across all pages.

- Testing
    - Enough test cases are to be conducted to fully test out each component.
    - Interface testing should ensure no errors occur when navigating through the interface using any type of interaction.
    - Each feature should be tested.

- Documentation
    - A clear description of processes and components used.
    - Gantt and PERT Charts should be created and updated to reflect progress on the project.

Chapter 7 Testing and Experiment

7.1 Testing and Experiment Scope

Test process for the project aims to test the functionality of each component of the hosting website for the project. The project will host a website where the user can input a set of images to test the project and output the accuracy results. Unit testing will focus on:

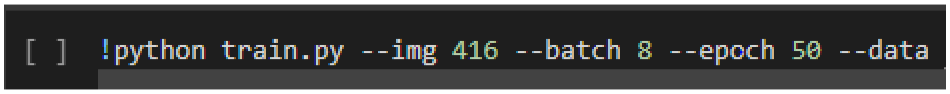
Testing	Condition	Description
Image train set upload	T/F	If image set was uploaded successfully
Display upload images	T/F/Partial	If image set was able to be displayed or partially displayed
Image test upload	T/F	If test image was uploaded successfully
Recognition Accuracy	T/F	If accuracy was able to be displayed

Table 7.100 Testing Scope

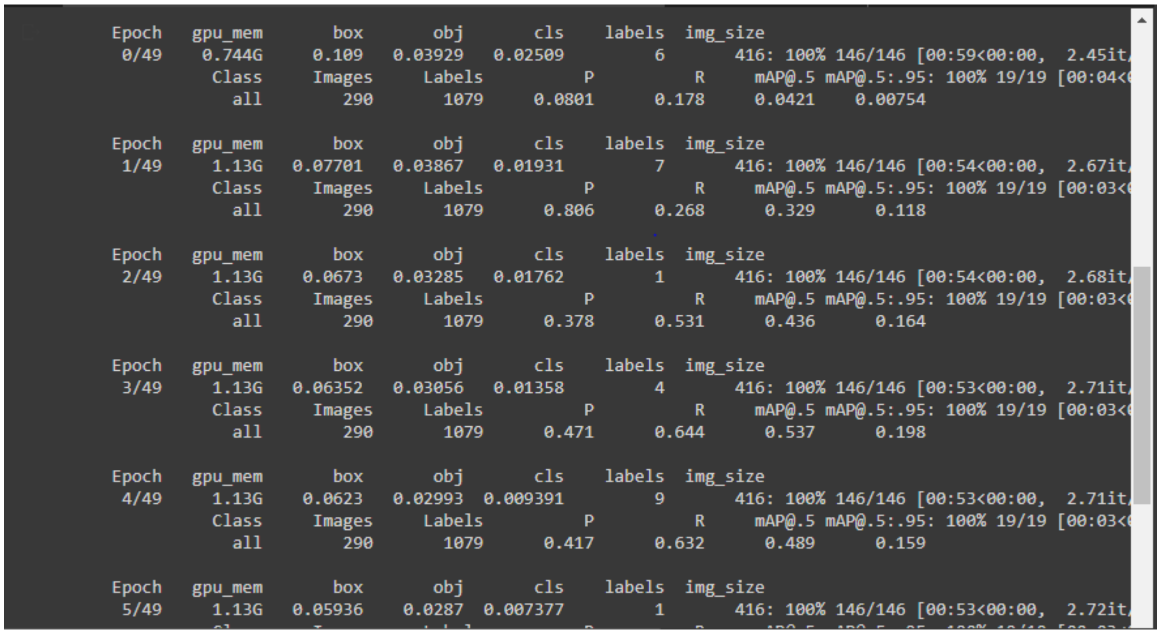
7.2 Testing and Experiment Approach

The testing and experiment approach that our team used to ensure that we achieved the best results possible included multiple methods. After testing multiple different algorithms and functions, we were able to narrow down our testing methods to those which are the most accurate in identifying whether or not an individual is wearing a mask or not. We are focusing on refining these test methods to aim for 100% test coverage for the final product launch on demo day.

Sample Python epoch training:



```
[ ]  !python train.py --img 416 --batch 8 --epoch 50 --data
```



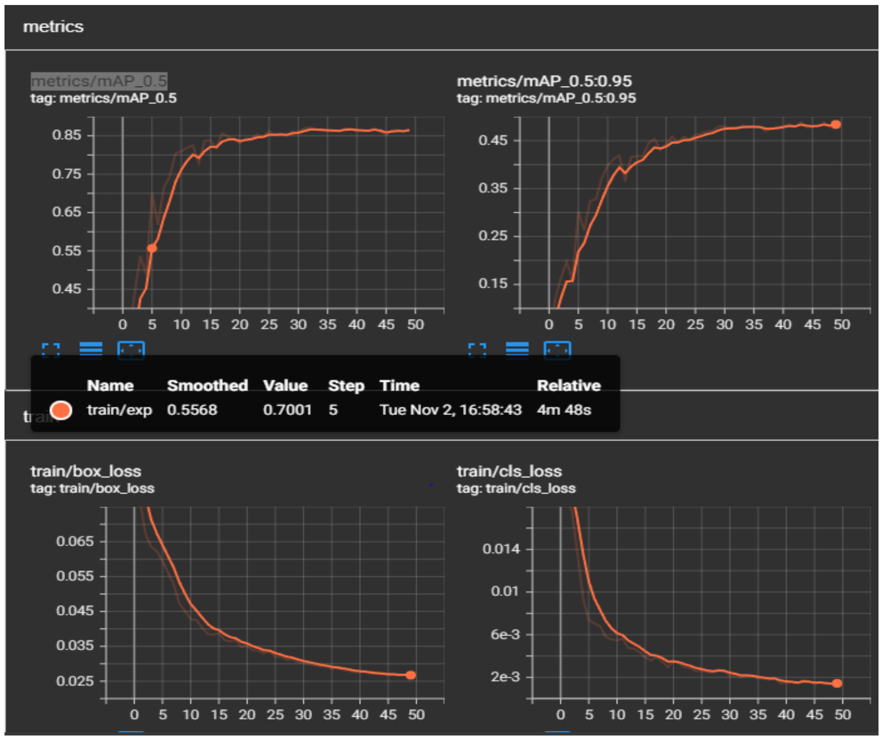
```
     Epoch   gpu_mem       box       obj       cls    labels  img_size
      0/49    0.744G     0.109   0.03929   0.02509         6       416: 100% 146/146 [00:59<00:00,  2.45it
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 19/19 [00:04<
                 all        290       1079     0.0801      0.178     0.0421    0.00754

     Epoch   gpu_mem       box       obj       cls    labels  img_size
      1/49     1.13G   0.07701   0.03867   0.01931         7       416: 100% 146/146 [00:54<00:00,  2.67it
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 19/19 [00:03<
                 all        290       1079      0.806      0.268      0.329      0.118

     Epoch   gpu_mem       box       obj       cls    labels  img_size
      2/49     1.13G    0.0673   0.03285   0.01762         1       416: 100% 146/146 [00:54<00:00,  2.68it
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 19/19 [00:03<
                 all        290       1079      0.378      0.531      0.436      0.164

     Epoch   gpu_mem       box       obj       cls    labels  img_size
      3/49     1.13G   0.06352   0.03056   0.01358         4       416: 100% 146/146 [00:53<00:00,  2.71it
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 19/19 [00:03<
                 all        290       1079      0.471      0.644      0.537      0.198

     Epoch   gpu_mem       box       obj       cls    labels  img_size
      4/49     1.13G    0.0623   0.02993  0.009391         9       416: 100% 146/146 [00:53<00:00,  2.71it
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 19/19 [00:03<
                 all        290       1079      0.417      0.632      0.489      0.159

     Epoch   gpu_mem       box       obj       cls    labels  img_size
      5/49     1.13G   0.05936    0.0287  0.007377         1       416: 100% 146/146 [00:53<00:00,  2.72it
```

Figure.7.201 Python Epoch Training

This testing method works by utilizing epochs in deep learning. An epoch is the training of a neural network with all of the training data for one cycle through the dataset.

Essentially this number of epochs we chose to run was 50 stands as a hyperparameter which defines the number of times that the Yolov5 learning algorithm will run through the dataset images. It is defined in the article “Difference Between a Batch and an Epoch in a Neural Network” in the “Machine Learning Mastery” journal that “One epoch means that each sample in the training dataset has had an opportunity to update the internal model parameters. An epoch consists of one or more batches. For example, as above, an epoch that has one batch is called the batch gradient descent learning algorithm.” The time it takes to train each model for our purpose takes approximately 1 hour for 50 epochs. Moving forward, we will perform more iterations to minimize the possibility of obtaining an error in our results.

Same TensorFlow machine learning framework:

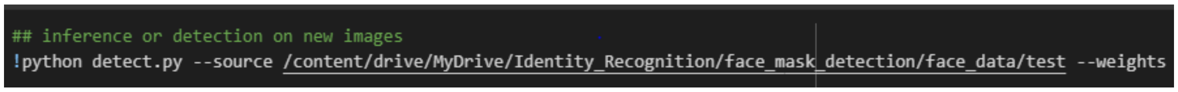


Name	Smoothed	Value	Step	Time	Relative
train/exp	0.5568	0.7001	5	Tue Nov 2, 16:58:43	4m 48s

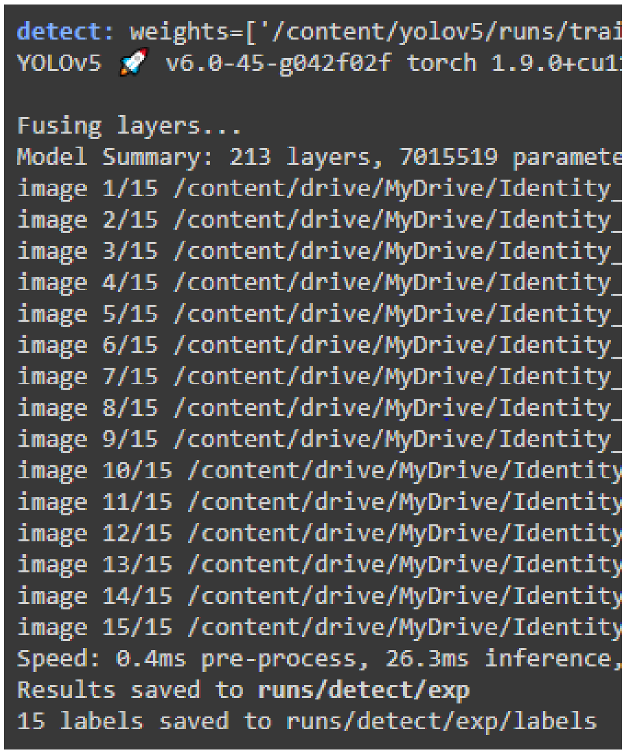
Figure.7.202 TensorFlow metrics

The TensorFlow metrics serve as a good baseline to show proof and/or monitor whether or not the training the Yolov5 model was conducting was actually happening or not. This test method provided us with a real-time test model to show how the software was reacting to the training model.

Sample Inference/Detection on Images outside of the training model:



```
## inference or detection on new images
!python detect.py --source /content/drive/MyDrive/Identity_Recognition/face_mask_detection/face_data/test --weights
```



```
detect: weights=['/content/yolov5/runs/trai
YOLOv5 🚀 v6.0-45-g042f02f torch 1.9.0+cu1

Fusing layers...
Model Summary: 213 layers, 7015519 paramete
image 1/15 /content/drive/MyDrive/Identity_
image 2/15 /content/drive/MyDrive/Identity_
image 3/15 /content/drive/MyDrive/Identity_
image 4/15 /content/drive/MyDrive/Identity_
image 5/15 /content/drive/MyDrive/Identity_
image 6/15 /content/drive/MyDrive/Identity_
image 7/15 /content/drive/MyDrive/Identity_
image 8/15 /content/drive/MyDrive/Identity_
image 9/15 /content/drive/MyDrive/Identity_
image 10/15 /content/drive/MyDrive/Identity
image 11/15 /content/drive/MyDrive/Identity
image 12/15 /content/drive/MyDrive/Identity
image 13/15 /content/drive/MyDrive/Identity
image 14/15 /content/drive/MyDrive/Identity
image 15/15 /content/drive/MyDrive/Identity
Speed: 0.4ms pre-process, 26.3ms inference,
Results saved to runs/detect/exp
15 labels saved to runs/detect/exp/labels
```

Figure.7.203 Detection on Images

The next step that our team took in regards to testing was to utilize detection on our newly trained model. Once we had trained the model, we gathered a new set of images (separate from the model we trained) and ran the test on those images to verify the software works properly.

Image Results:

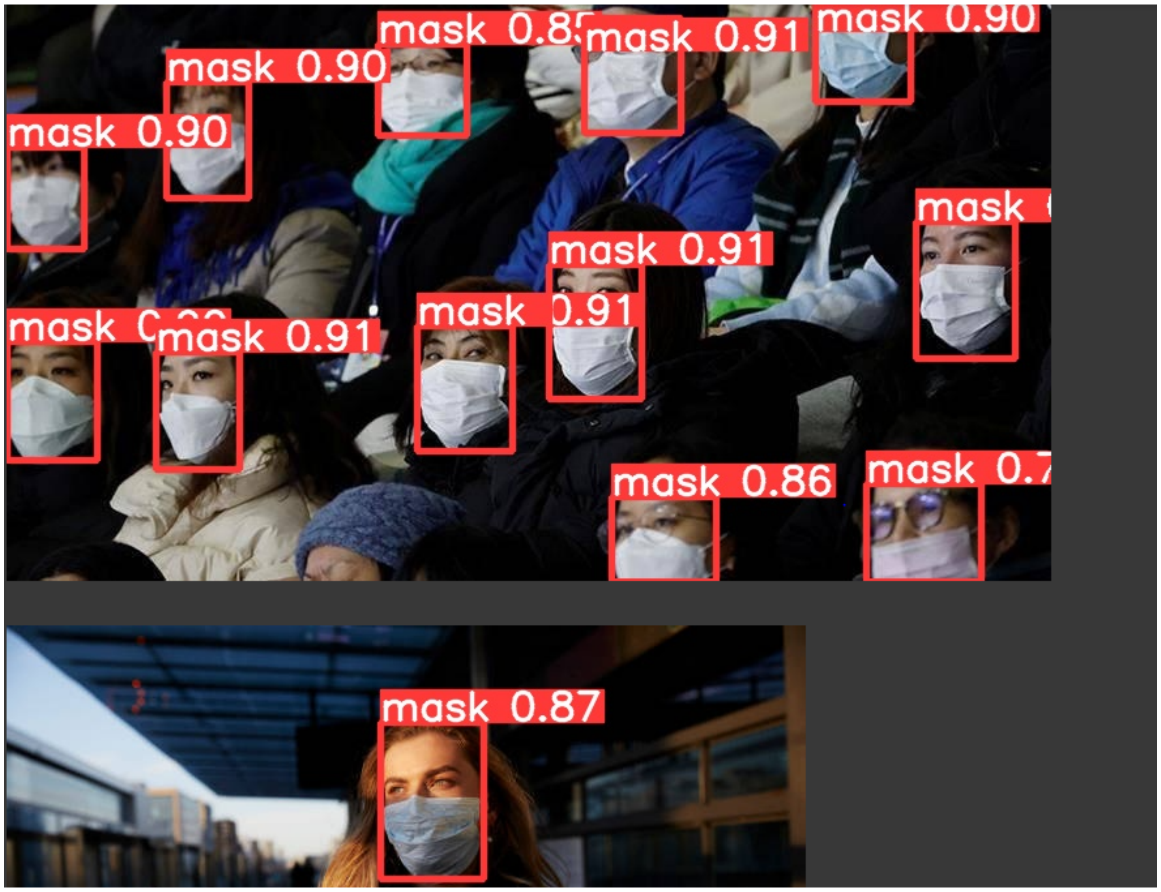


Figure.7.204 Image Results (1)

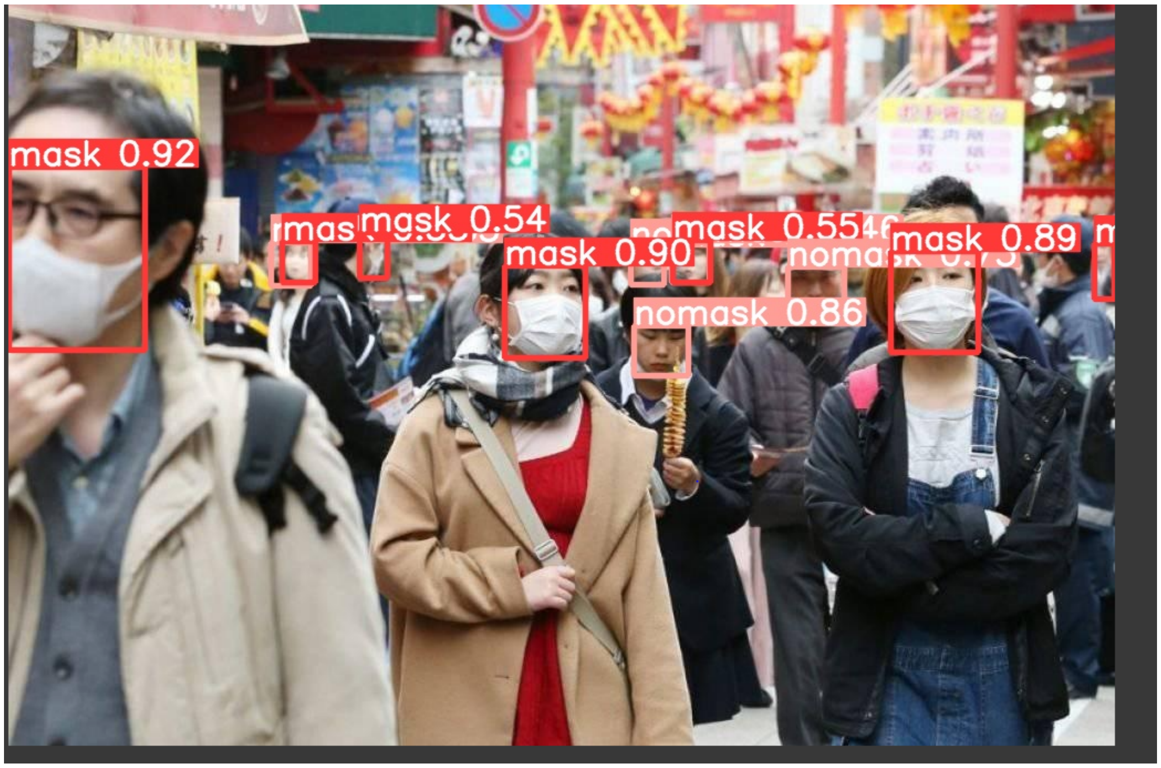


Figure.7.205 Image Results (2)

The final test that our group performed was a visual test on the images. As you can see in both Figure.7.204 and Figure.7.205, a box is placed around an individual's head and labels them as either having a mask on or not having a mask on (mask or no mask). This test method allowed us to verify that the test methods previously defined were working properly.

7.3 Testing and Experiment Results and Analysis

Test ID Number	Use Case	Bugs Identified	Description / Analysis	Test Result Summary
01	User	No	Is able to successfully identify whether the person(s) in the image are wearing a mask or not.	Pass. Box is placed around person(s) head labeled with mask or no mask correctly
02	User	No	Is able to successfully identify whether the person(s) in the image are wearing a mask or not.	Pass. Box is placed around person(s) head labeled with mask or no mask correctly
03	User	No	Is able to successfully identify whether the person(s) in the image are wearing a mask or not.	Pass. Box is placed around person(s) head labeled with mask or no mask correctly
04	User	No	Is able to successfully identify whether the person(s) in the image are wearing a mask or not.	Pass. Box is placed around person(s) head labeled with mask or no mask

				correctly
05	User	No	Is able to successfully identify whether the person(s) in the image are wearing a mask or not.	Pass. Box is placed around person(s) head labeled with mask or no mask correctly
06	User	No	Is able to successfully identify whether the person(s) in the image are wearing a mask or not.	Pass. Box is placed around person(s) head labeled with mask or no mask correctly
07	User	Yes	Is not able to successfully identify whether the person(s) in the image are wearing a mask or not.	Fail. Some of the people in the image have boxes around their head with the wrong label or no box at all.
08	User	No	Is able to successfully identify whether the person(s) in the image are wearing a mask or not.	Pass. Box is placed around person(s) head labeled with mask or no mask correctly
09	User	No	Is able to successfully identify whether the	Pass. Box is placed around person(s)

			person(s) in the image are wearing a mask or not.	head labeled with mask or no mask correctly
10	User	No	Is able to successfully identify whether the person(s) in the image are wearing a mask or not.	Pass. Box is placed around person(s) head labeled with mask or no mask correctly
11	User	No	Is able to successfully identify whether the person(s) in the image are wearing a mask or not.	Pass. Box is placed around person(s) head labeled with mask or no mask correctly
12	User	No	Is able to successfully identify whether the person(s) in the image are wearing a mask or not.	Pass. Box is placed around person(s) head labeled with mask or no mask correctly
13	User	No	Is able to successfully identify whether the person(s) in the image are wearing a mask or not.	Pass. Box is placed around person(s) head labeled with mask or no mask correctly
14	User	No	Is able to successfully	Pass. Box is placed

			identify whether the person(s) in the image are wearing a mask or not.	around person(s) head labeled with mask or no mask correctly
15	User	Yes	Is not able to successfully identify whether the person(s) in the image are wearing a mask or not.	Fail. Some of the people in the image have boxes around their head with the wrong label or no box at all.

Table.7.300 Use Case Analysis

Overall, the various use cases depicted in Figure.7.3.000 play a role in determining the accuracy of the Yolov5 trained model.

Chapter 8 Conclusion and Future Work

8.1 Conclusion

In this thesis, we addressed the problem of facial recognition systems for people wearing masks using a complex program to capture and recognize an image of an individual when they are wearing a mask/face covering. One of the main contributions of our work is to provide AI and ML developers with information on how to manipulate existing facial recognition software to increase the capabilities of the program to identify humans who are wearing face masks. This facial recognition system incorporated biometric concepts, machine learning, and artificial intelligence concepts to identify, compare, and process the given facial image. In addition, python in the Yolov3 and Yolov5 libraries is utilized to implement the algorithm, and OpenCV technology is used to identify an individual successfully if they had a mask on or not .

Moreover, tests on different images and circumstances have been provided. In particular, we tested the performance of both people wearing masks and not wearing masks, and the tests proved that the system is working correctly to identify individuals if they are wearing masks. However the system had mixed results when attempting to identify a person underneath the mask

References

Anderson, M. The State of Facial Recognition Software in 2020. (2020, August 24). Retrieved March 16, 2021, from https://www.iflexion.com/blog/facial-recognition-software

Facial recognition: Top 7 trends (tech, Vendors, markets, use cases & latest news). (2020, February 20). Retrieved March 17, 2021, from https://www.thalesgroup.com/en/markets/digital-identity-and-security/government/biometrics/facial-recognition

Fontenot, S. Study Outlines What Creates Racial Bias In Facial Recognition Technology. (2020, December 4). Retrieved March 15, 2021, from https://news.utdallas.edu/science-technology/racial-bias-facial-recognition-2020/

Gershgorn, D. New Facial Recognition Tech Only Needs Your Eyes and Eyebrows. (2020, May 7). Retrieved March 16, 2021, from https://onezero.medium.com/new-facial-recognition-tech-only-needs-your-eyes-and-eyebrows-9e7dc155cd7f

Jain, Anil, Bolle, Ruud, and Pankanti, Sharath. Biometrics. Vol. 479. Boston: Springer, 2006. The International Ser. in Engineering and Computer Science. Web.

Leprince-Ringuet, D. Facial recognition: Now algorithms can see through face masks. (2021, January 6). Retrieved March 17, 2021, from https://www.zdnet.com/article/facial-recognition-now-algorithms-can-see-through-face-masks/

NEC face recognition engine provides highly accurate results even when face masks are worn. (2020, September 24). Retrieved March 17, 2021, from https://www.nec.com/en/press/202009/global_20200924_01.html

Prasanna, Mary D., and Ganapathy C. Reddy. “Development of Real Time Face Recognition System Using OpenCV.” *International Research Journal of Engineering and Technology (IRJET)*, vol. 04, no. 12, 12 Dec. 2017.

Sharma, Sahil, and Kumar, Vijay. "Performance Evaluation of Machine Learning Based Face Recognition Techniques." Wireless Personal Communications (2021): Wireless Personal Communications, 2021-02-16. Web.

Vedula, L. A.(2020) FaceMask_Validator [Source code]. https://github.com/alekhyaved/FaceMask_Validator

Woodward, John D, et al. *Biometrics: A Look at Facial Recognition*. Rand, 2003, apps.dtic.mil/sti/pdfs/ADA414520.pdf.

Wagner, Philipp. *Face Recognition with Python*. 18 July 2012, larmor2.nuigalway.ie/~emil/ma500/facerec_python.pdf.

Appendices (Optional)

Appendix A – Appendix Title

[Typical example: you can include a specific standard here.]

Appendix B – Appendix Title

[Typical example: you can include a specific interface detail here.]