Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра мережевих та інтернет технологій

ЗАТВЕРДЖУЮ
завідувач кафедри
мережевих та інтернет технологій
_____ Юрій КРАВЧЕНКО
«____»_____2023 року

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

галузі знань 17 «Електроніка та телекомунікації»
за спеціальністю 172 «Телекомунікації та радіотехніка»
освітньо-професійна програма «Мережеві та інтернет технологій»

на тему:

ВИКОРИСТАННЯ МЕТОДІВ DEEP LEARNING ДЛЯ СТВОРЕННЯ СИСТЕМИ МОНІТОРИНГУ МАСКОВОГО РЕЖИМУ

Виконала: студентка групи МІТ -41

Катерина ШМАТ _____
(ім'я та ПРІЗВИЩЕ) (підпис)

Керівник: доцент кафедри мережевих та інтернет технологій

к.т.н., доцент Олександр ТРУШ _____
(науковий ступень, вчене звання, ім'я та ПРІЗВИЩЕ) (підпис)

Київ 2023

Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра мережевих та інтернет технологій

ЗАТВЕРДЖУЮ
завідувач кафедри
мережевих та інтернет технологій
_____Юрій КРАВЧЕНКО
«____»_____2023 року

ЗАВДАННЯ
НА ДИПЛОМНУ РОБОТУ

Здобувачу вищої освіти Шмат Катерина Сергіївна
(прізвище, ім'я, по батькові)

1. Тема роботи:
Використання методів Deep learning для створення системи моніторингу маскового режиму

затверджена на засіданні кафедри МІТ « 7 » грудня 2022 р. протокол № 5

2. Термін здачі закінченої роботи «30» травня 2023 р

3. Вихідні дані до проекту (роботи)

- Методи системи моніторингу маскового режиму
- Тренування нейронних мереж

4. Зміст пояснювальної записки (перелік питань, що їх потрібно розробити, обсяг – 35-40 стор.)

1. Принципова функціональна схема системи моніторингу
2. Обробка візуальної інформації з OpenCV
3. Тренування нейронних мереж

5. Перелік графічного матеріалу 8-10 слайдів

Презентація доповіді із 10 слайдів

Дата видачі завдання _____

Керівник роботи _____ к.т.н., доцент Олександр ТРУШ
(підпис) (посада, ім'я, ПРІЗВИЩЕ)

Завдання прийняв до виконання Катерина ШМАТ
(підпис) (ім'я, ПРІЗВИЩЕ)

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ РОБОТИ

Номер	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Підготовчий	20.01.2023	Вик.
2	Розділ 1	17.02.2023	Вик.
3	Розділ 2	24.03.2023	Вик.
4	Розділ 3	21.04.2023	Вик.
5	Доповідь та слайди	20.05.2023	Вик.
6	Пояснювальна записка	30.05.2023	Вик.

Здобувач вищої освіти _____ Катерина ШМАТ
(підпис) (ім'я, ПРІЗВИЩЕ)

Керівник _____ Олександр ТРУШ
(підпис) (ім'я, ПРІЗВИЩЕ)

РЕФЕРАТ

Мета роботи: створення системи моніторингу маскового режиму задля того, щоб запобігти розповсюдженню вірусу Covid-19 та інших аерозольних інфекцій наскільки це можливо.

Об'єкт дослідження: світова пандемія COVID-19 та поширені аерозольні інфекції

Предмет дослідження: система моніторингу, що включає методи deep learning та роботу з нейронними мережами.

***Методи дослідження*:** deep learning, робота з бібліотеками OpenCV, Keras, TensorFlow, сегментація зображень, класифікація зображень

Короткий зміст роботи: у зв'язку з виникненням масової пандемії у місцях масового скупчення людей виникла потреба постійного контролю маскового режиму задля зниження захворюваності серед населення. Через те, що в громадських місцях зазвичай сотні людей, а між місцями контролю маскового режиму люди можуть вдягати маску не правильно або ж зовсім знімати, що може призвести до підвищення рівня захворюваності, буде доцільно встановити камери – системи моніторингу, що будуть стежити за відвідувачами та визначати чи не підвищує людина можливість розповсюдження хвороби, чи правильно носить засіб особистого захисту.

По ходу роботи було розроблено програму, яка розпізнає обличчя відвідувачів торгівельного закладу або будь-якого іншого громадського місця, опрацьовує зображення та проводить класифікацію за заданими параметрами. Після чого фото порушників зберігається та відправляється в окрему папку, а сповіщення про порушення надходить адміністратору.

Ключові слова: система моніторингу, масковий режим, deep learning, нейронна мережа, розпізнавання обличь

ЗМІСТ

ВСТУП

На сьогоднішній день більшість країн світу, охоплені пандемією COVID-19. Вже доволі тривалий час вчені та лікарів намагаються протистояти вірусу, винаходять вакцину та рятують людей. Нажаль зараз ще немає способу повністю захиститися від вірусу, проте є деякі рекомендації щодо того, як можна знизити ризик захворюваності.

Одним з засобів індивідуального захисту є медичні маски. Якщо кожна людина буде носити маску у людних місцях, то ризик захворюваності може дорівнювати близько 5%, тоді як вірогідність зараження людини, якщо вона нехтує носінням маски, майже 100% . Отже доцільним буде проводити постійний контроль маскового режиму у людних місцях, адже саме правильне та сумлінне носіння маски може стати шляхом до подолання пандемії [1].

Зараз у кожному торгівельному або розважальному центрі, а також у публічному транспорті або у закладах харчування можна знайти працівника, який при вході у приміщення веде контроль правильності одягнення маски у кожної людини. Таким чином це, здавалося б, приводить до зниження ризику зараження вірусом серед відвідувачів. Проте це не завжди так. Адже можна помітити, що коли людина проходить контроль, то вже спускає маску, носить її неправильно або ж і геть знімає та кладе у кишеню. Так як вірус COVID-19 з кожним днем змінюється та зазнає мутацій, то навіть вакциновані люди можуть захворіти, а тим більш стати носієм хвороби. Тому кожному потрібно сумлінно носити маску увесь час знаходячись у людних місцях, а адміністрація закладу повинна забезпечити контроль маскового режиму не тільки при вході у заклад, а й по всьому приміщенні, адже через недотримання правил носіння маски деякими людьми, інші можуть захворіти та рознести цю хворобу далі.

Для постійного контролю маскового режиму по всьому закладу буде простіше використовувати камери з можливістю маскового контролю. Адже це може зберегти ресурси самого закладу та час працівників, які будуть стежити за

відвідувачами. Також контролерові буде доволі складно слідкувати за сотнями, а можливо навіть тисячами людей в одному приміщенні водночас.

На основі deep learning було розроблено систему моніторингу маскового режиму, яка може замінити контролерів у багатьох закладах. Навчена програма може стежити водночас за декількома людьми. Вона починає цикл моніторингу з розпізнавання обличчя за допомогою OpenCV. Наступним етапом є сегментація зображень. Це виконується задля того щоб було простіше виявляти правильність одягнення маски. Сегментовані та відформатовані зображення передаються для формування та тренування моделі, яка створюється за допомогою нейронної мережі. Вже створена модель, яка попередньо пройшла тренування, в змозі класифікувати фотографії за розділами по мірі правильності одягнення маски. Класифіковані дані надходять до сховища, у якому вже безпосередньо адміністрація закладу може побачити фото порушників та попередити високу захворюваність інших відвідувачів, після отримання звукового сповіщення про недотримання правил носіння маски від самої системи моніторингу маскового режиму.

Камеру з програмою для контролю маскового режиму можна встановити у будь-якому місці та положенні так, щоб було добре видно всіх людей навколо, тому буде доцільно використовувати саме такий спосіб моніторингу. Для початку система контролю може замінити декількох працівників, що стежать за дотриманням маскового режиму, а згодом й зовсім зможе замінити усіх контролерів, задля забезпечення повного моніторингу по всьому закладу усіх відвідувачів та навіть зекономить ресурси адміністрації.

РОЗДІЛ 1. ПРИНЦИПОВА ФУНКЦІОНАЛЬНА СХЕМА СИСТЕМИ МОНІТОРИНГУ

1.1 Постановка завдання

Система моніторингу – це цілеспрямоване спостереження за одним або декількома об’єктами в просторі і часі, а також збирання, обробку та збереження та отриманої інформації з довкілля, прогнозування змін, аналіз, а також передача інформації, розробка рекомендацій щодо подальшого прийняття рішень та уникнення непотрібних змін. Завдяки системам моніторингу можна встановити постійний нагляд за будь-чим у реальному часі у будь-якій сфері.

Загалом системи моніторингу поділяються на дві категорії: активний моніторинг та проактивний. Перший тип забезпечує спостереження за об’єктом у реальному часі та має здатність виявляти невідповідність параметрів програми до заданих, а також знаходить уразливі місця системи. Активний моніторинг є чудовою системою для тестування справності всіх функцій системи, де він використовується прямо під час її роботи. Проактивний моніторинг у свою чергу може давати рекомендації щодо покращення програми та навіть здатний побудувати плани розвитку системи. Він також прогнозує та може запобігти несправності системи шляхом своєчасно зібраних даних та аналізу поведінки програми [2].

Таким чином система моніторингу завдяки своїм властивостям є ідеальним рішенням для впровадження задля контролю маскового режиму у людних місцях.

У комплексі з системою моніторингу обов’язково повинні працювати IT-технології – процеси створення та збереження інформації, а також передачі та сприйняття інформації. За допомогою IT-технологій можна визначити методи для реалізації заданих процесів.

Інформаційні технології потрібні містити наступні компоненти:

- програмне забезпечення
- організаційне забезпечення
- технічні засоби IT

Кожний з компонентів є дуже важливим та потребує до себе уваги. Програмне забезпечення надає можливість використовувати в роботі різні програмні коди для керування системами. Виконання програм забезпечує правильну дію системи, дозволяє прописати таку кодовану інструкцію, що буде відповідати запитам розробника, виконувати чітко задані дії. Технічні засобі являють собою сукупність засобів та пристроїв, які в свою чергу будуть здатні відтворити те, що подається з програмного забезпечення. Також вони дають можливість розробнику переглянути як саме працює програма та чи виконуються всі прописані функції та дії. Організаційне забезпечення показує взаємодію розробника з безпосередньо програмами та створеною системою.

На сьогоднішній день існує безліч IT-технологій і вони з’являються майже кожен день та описати їх усі майже неможливо, тому ми зупинимо свою увагу на тих технологіях, які будуть використані у даній роботі [3]. Повна схема представлена на рисунку 1.1.

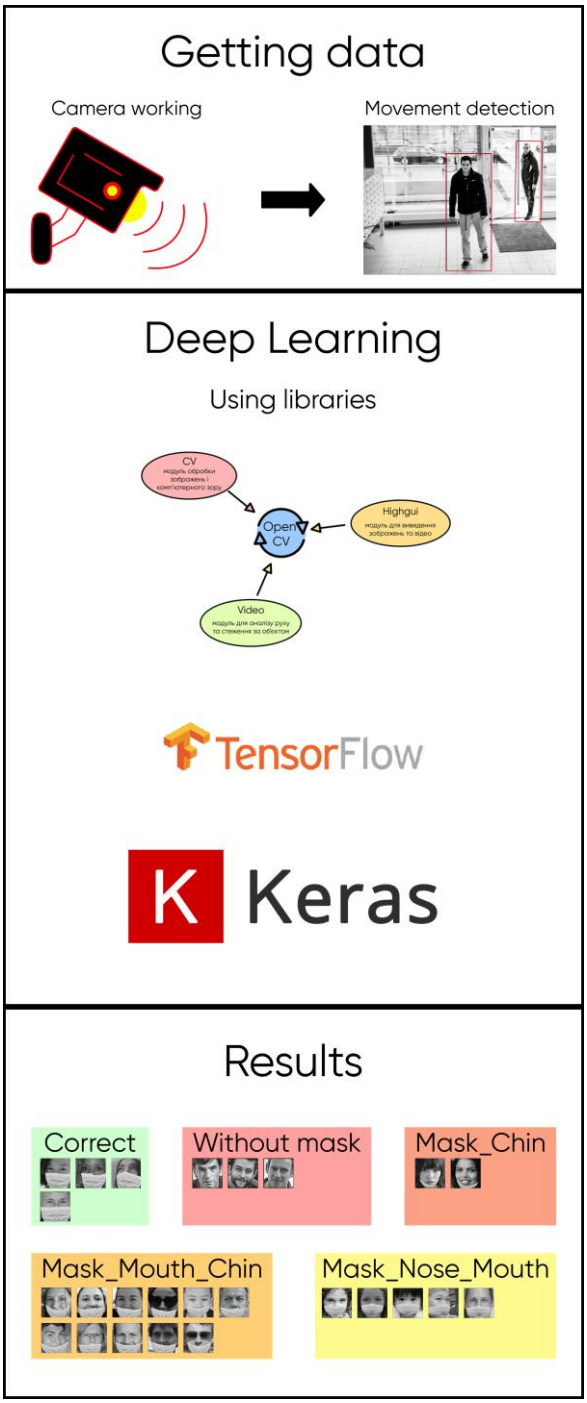

Рисунок 1.1 - Схема функціонування системи

Перш за все найважливішим елементом є камера відео нагляду, в якій буде знаходитися ціла система для контролю маскового режиму. Камеру можна встановити у будь якому місці та положенні на території з гарною видимістю.

Наступний, не менш важливий елемент – OpenCV. Це ціла бібліотека алгоритмів, до яких входять алгоритми комп'ютерного зору, алгоритми загального призначення, алгоритми з відкритим кодом, алгоритми для обробки

зображень. За допомогою даної бібліотеки у системі моніторингу маскового режиму можна буде зробити по-перше, можливість фіксації руху об’єктів на заданій території. По-друге, розпізнавання обличь усіх людей, які потрапили у поле спостереження камери. Далі буде підключена можливість обрізати обличчя та піддавати його сегментуванню задля того, щоб вирішити чи правильно вдягнена маска.

У даній системі також буде використано TensorFlow – бібліотеку для розробки систем, які використовують технології машинного навчання. Ця бібліотека реалізовує багато алгоритмів різної складності задля вирішення поширених задач машинного навчання, серед яких якраз таки можна зазначити розпізнання образів та прийняття рішень.

Разом з Keras – бібліотекою, яка дозволяє створювати нейронні мережі, TensorFlow дозволяє користуватися спрощеною конструкцією нейронної мережі.

Нейронна мережа – це модель, включаючи її програмне втілення, яка побудована за таким принципом, як організація та функціонування біологічних нейронних систем. Вона дозволяє вирішити багато питань таких як розпізнавання образів, їх класифікація, кластеризація, прогнозування, аналіз даних та інше. Нейронні мережі використовують для моделювання різних процесів а також систем.

Виходячи з вищезазначеного стає зрозуміло, що усі використані IT-технології є необхідними та будуть дуже потрібні для складання цілої, повноцінної, робочої системи. Моделювання з використанням нейронної мережі буде також важливим та полегшуючим фактором, адже саме завдяки їй система зможе розпізнавати зображення, аналізувати відео, класифікувати їх та виносити прогнози та рішення на майбутнє.

Підсумовуючи вищезазначене у цьому підрозділі можна поставити план завдання для розробки системи моніторингу:

- по-перше, впровадити бібліотеку OpenCV для фіксації руху, розпізнання людей у полі зору, виділення облич кожної людини, подальше їх форматування та сегментування

- по-друге, впровадити бібліотеки, які зможуть забезпечити полегшене будування нейронної мережі

- по-третє, безпосередньо будування нейронної мережі для вже класифікації виділених облич

- по-четверте, збереження фотографій порушників маскового режиму задля покарання та забезпечення іншим людям максимального захисту від COVID-19 [4].

1.2 Компоненти системи

У даній роботі більш за все за допомогою нейронної мережі було виконано класифікацію отриманих обличь. Так як носити маску можна по-різному, то зображення були розподілені на декілька класів способів одягнення маски, які було представлено у наборі даних, взятих для створення нейронної мережі та моделювання, а саме:

- Маска одягнена правильно
- Маска не прикриває ніс
- Маска не прикриває підборіддя
- Маска одягнена лише на підборіддя
- Маска взагалі відсутня

Після того, як OpenCV виконає свою роботу з розпізнання та обробки зображення, починається класифікація. Програма пробігається по кожному зображенню, порівнює його з тим, яке вона вже колись бачила та виводить результат у вигляді відношення обличчя до однієї з зазначених виділених категорій. Таким чином система розпізнає чи правильно вдягнена маска та може сповістити про це адміністрацію.

Принципова функціональна схема системи моніторингу має структуру, яку можна побачити на рисунку 1.2.

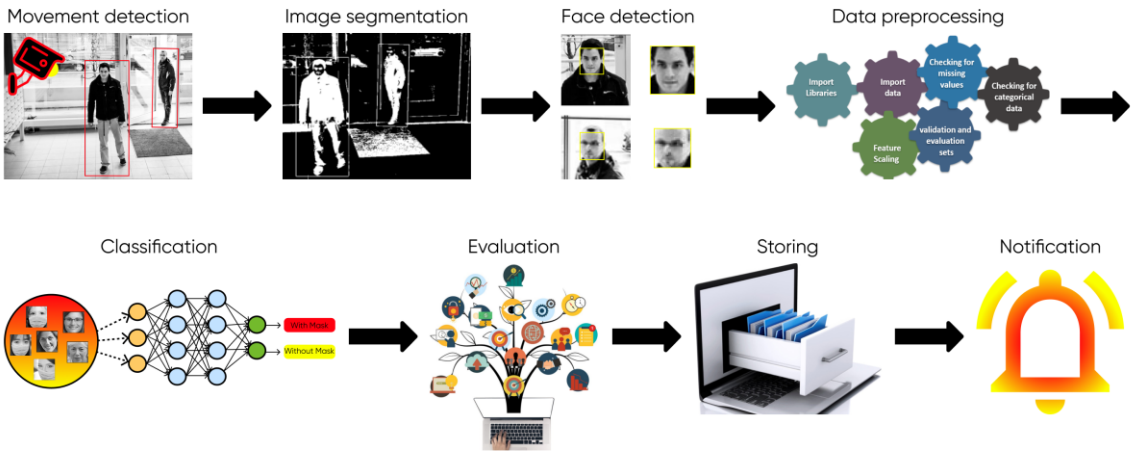

Рисунок 1.2 - Принципова функціональна схема

Використовуючи усі зазначені компоненти комплексно, можна досягти максимально вдосконаленої роботи системи моніторингу маскового режиму у громадських місцях, чим можна по-перше, забезпечити безпеку відвідувачів того чи іншого торгівельного центру, спортивного комплексу, тощо, по-друге, розвантажити людей, що слідкують власними очима за відвідувачами, по-третє, завдяки збереженню зображень порушників, можна полегшити систему покарань та штрафів за некоректне носіння маски у громадських місцях [5].

1.3 Етапи роботи програми

На першому етапі «Виявлення руху» відбувається спостереження за навколишнім середовищем за допомогою камери, у якій наявні усі потрібні для системи моніторингу алгоритми. Проте найголовнішим у цей момент є звернення до OpenCV, адже саме ця бібліотека виконує функцію розпізнавання обличь. За допомогою саме OpenCV можливо побачити та отримати навіть опис того, що на даний момент часу бачить камера відео нагляду, включаючи розміри та інші характеристики зображення. Як бачить OpenCV можна побачити на рисунку 1.3.

Рисунок 1.3 – Бачення за допомогою бібліотеки OpenCV

На другому етапі «Сегментація зображень» відбувається форматування зображень, а саме їх ділення на множину пікселів для спрощення представленого зображення. Це потрібно для того, щоб наступні бібліотеки та алгоритми могли виконати свою роль у аналізі, класифікації та збереженні зображень легше та швидше. Сегментація зображень робить їх простішими для сприйняття подальшою системою та впливає на складність аналізу. Сегментація буде працювати з зображенням по заданих розробником параметрах, а саме може сегментувати зображення за різними властивостями, наприклад, глибиною

кольору, текстурою, роздільністю та інше [6]. Як відбувається сегментація зображень можна побачити на рисунку 1.4.

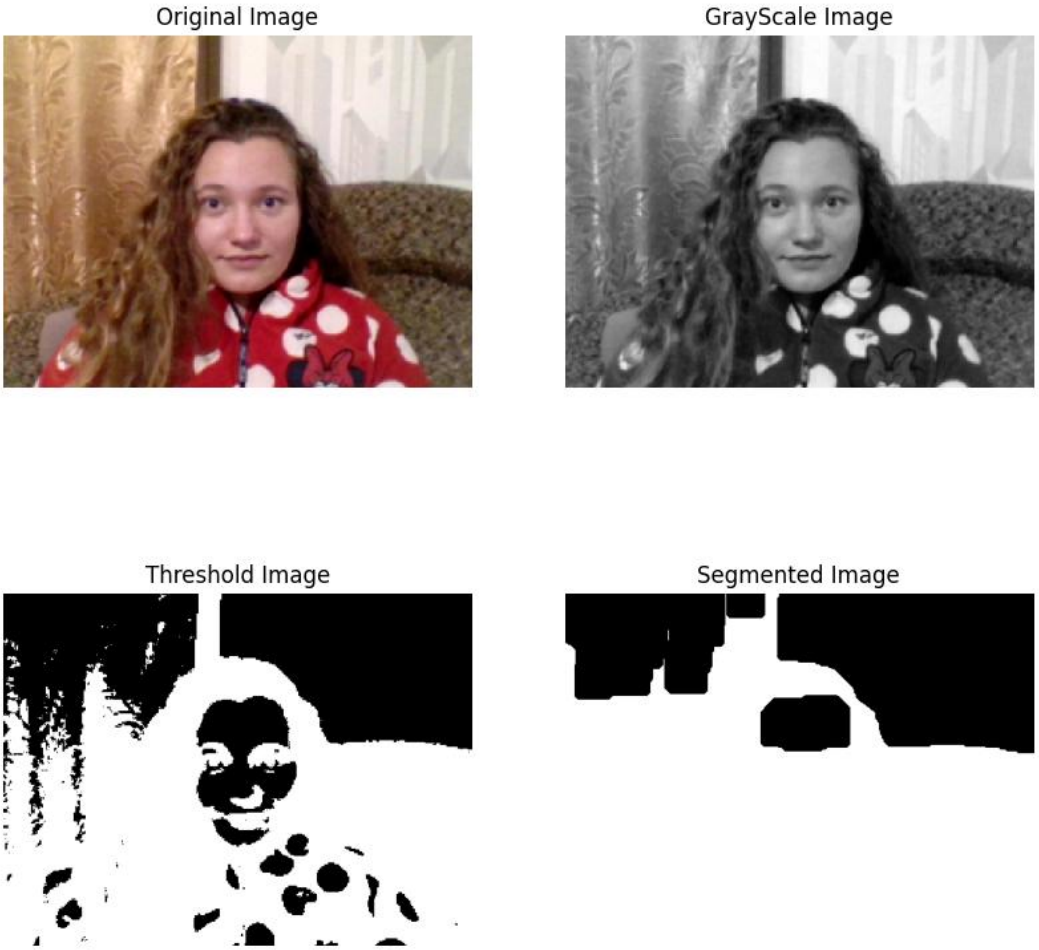

Рисунок 1.4 – Сегментація зображень

Третій етап «Виявлення обличь» полягає у тому, щоб на вже відформатованих фотографіях відшукати усі наявні обличчя людей та виділити їх. На цьому ж етапі також відбувається обрізка відшуканих обличь для передачі на наступний етап. Коли система обробляє отримане зображення, то вирізає кожне обличчя, що було знайдено та тримає їх в пам’яті для подальшої передачі на наступні етапи обробки інформації. Схему роботи з найденим на зображенні обличчям можна побачити на рисунку 1.5.

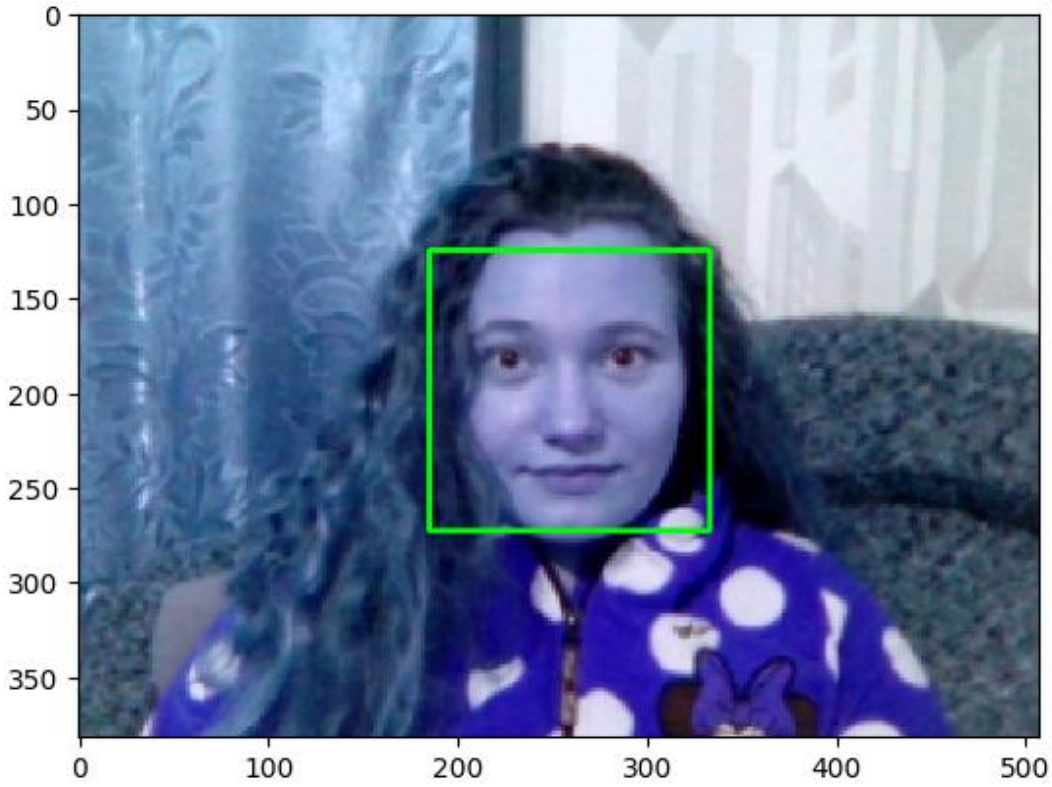

Рисунок 1.5 – Схема виявлення та виділення обличь на зображенні

На четвертий етап «Підготовка та обробка даних» приходять фотографії обличь, які є сегментованими та відформатованими – такі зображення наведені на рисунку 1.6. Починає проходити перевірка усіх цих зображень та пошук невірно обрізаних фрагментів або пошкоджених зображень, а також фотографій занадто низької якості, де не можна розрізнити будь що. Такі фотографії видаляються та не йдуть далі. Коли фотографії пройшли сортування, то проходить повторна та фінальна перевірка усіх даних задля того, щоб не допустити надходження помилок на наступний етап. Таким чином підвищується точність класифікації зображень та виявлення порушень, а також збереження часу адміністрації за рахунок того, що працівники не відволікаються на помилкові сповіщення системи про порушення маскового режиму відвідувачами їхнього закладу [7].

Рисунок 1.6 - Сегментовані та відформатовані зображення

П’ятий етап «Класифікація» на якому відбувається вже безпосередньо класифікація усіх отриманих зображень. За допомогою нейронної мережі всі фотографії займають місце в одному з п’яти відділів за призначенням. Зображення можуть бути розміщенні в таких папках:

- Маска одягнена правильно
- Маска не прикриває ніс
- Маска не прикриває підборіддя
- Маска одягнена лише на підборіддя
- Маска взагалі відсутня

На шостому етапі «Еволюції даних» система вже повністю обізнана про зібрані дані та стійка до них. Також вже на цьому етапі є можливість для системи взяти повний контроль над даними, що вона має, задля того, щоб спокійно провести їх по іншим етапам [8].

На сьомому етапі «Збереження зображень» відбувається збереження знімків у сховище так, щоб у будь-який момент адміністрація змогла витягнути потрібну інформацію, в тому числі і про порушників та вплинути на поліпшення дотримання відвідувачами маскового режиму. На рисунку 1.7 можна побачити як виглядає сховище для зображень з різними типами порушень або без них.

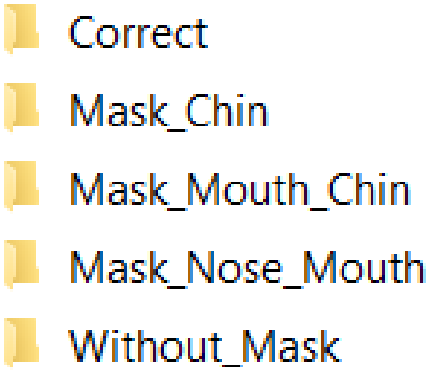

Рисунок 1.7 – Сховище зображень, що поділяється на відділів

Восьмий етап «Сповіщення» завершує цикл системи моніторингу даних. Після того як дані розділені по папкам і в папці з порушеннями є хоча б одне зображення, система надсилає повідомлення про це до адміністрації для вжиття заходів.

Описана принципова схема дозволяє проводити весь комплекс моніторингу починаючи від розпізнавання обличь за допомогою камери з налаштованим в ній оточенням і закінчуючи створенням нейронної мережі, яка у свою чергу дозволяє виявляти правильність вдягненої маски та передавати інформацію до сховища, де адміністрація може у будь-який момент віднайти фото порушників для вживання заходів [9].

РОЗДІЛ 2. ОБРОБКА ВІЗУАЛЬНОЇ ІНФОРМАЦІЇ З OPENCV

2.1 Бібліотека OpenCV

Всі дії, що виконує система моніторингу, мають по-своєму складні алгоритми та безпосередньо всі є дуже важливими. Але найпершою з найважливіших дій є обробка візуальної інформації для можливості виконувати подальші дії над нею. Для обробки візуальної інформації використовується бібліотека комп'ютерного зору, а також бібліотека машинного навчання, яка має відкритий код – а саме OpenCV.

OpenCV призначена саме для підвищення ефективності обробки та обчислень відеозображення тут і зараз у реальному часі. Дана бібліотека дозволяє швидко і ефективно проходитись по алгоритмам машинного зору від простого до складного. Бібліотека містить багато різноманітних функцій, більше 500. Наявні функції дозволяють реалізовувати різноманітні програми у різноманітних областях та галузях, в тому числі для контролю маскового режиму, у інтерфейсі користувача, робототехніці.

Бібліотека OpenCV має достатньо корисних функцій та потрібних алгоритмів, які може використати будь-який розробник. За допомогою даного коду можна покращувати зображення або розмити фон, що не потрібен для класифікації зображень даною системою, можна видалити зайве на зображенні або робити зображення чіткішим за допомогою прибирання зайвих пікселів.

OpenCV також містить бібліотеку загальних функцій штучного інтелекту Machine Learning Library. Вона служить, в основному, для розпізнавання фрагментів зображення і їх кластеризації. Дану бібліотеку можна застосувати для різних цілей, наприклад, для стандартного інтерфейсу комп'ютерного зору, для створення нових різних моделей з використання PC, для прискорення обробки за допомогою OpenCV.

OpenCV має декілька модулів. Основними з них будуть:

- CV – модуль, який займається обробкою зображень і комп'ютерного зору. Він здатен виконувати базові операції над зображеннями:

перетворення кольорових просторів, фільтрацію, різноманітні геометричні перетворення. Проводить аналіз зображень: морфологічний, створення гістограм, пошук контурів. Також може спостерігати за об’єктами, проводити аналіз руху, виявлення об’єктів та калібрування камер. Саме цей модуль має змогу розпізнавати, що зображено на фото, включаючи обличчя людей. За допомогою модуля CV можна підлаштовувати камеру, її зір, а також налаштовувати характеристики зображення в процесі зйомки.

- Highgui – модуль, призначений для виведення зображень та відео. Також він створює призначений для користувача інтерфейс задля того щоб можна було захопити відео з камер або файлів, виділити статичне зображення для подальшого його запису або читання. Даний модуль забезпечує організацію протого для користування інтерфейсу.

- Video – модуль для аналізу руху та стеження за об’єктом. Також можливе створення шаблонів руху та видалення фону. На відео є можливість отримати статичний кадр задля подальшої обробки його вже іншими ситемами.

- Cxcore – ядро OpenCV. Даний модуль дозволяє аиконувати різноманітні операції над масивами інформації, а також корисний для відновлення даних з формату .xml. Cxcore може використовуватися для простої графіки у 2D форматі.

Зазначені складові та модулі бібліотеки OpenCV можна побачити на рисунку 2.1.

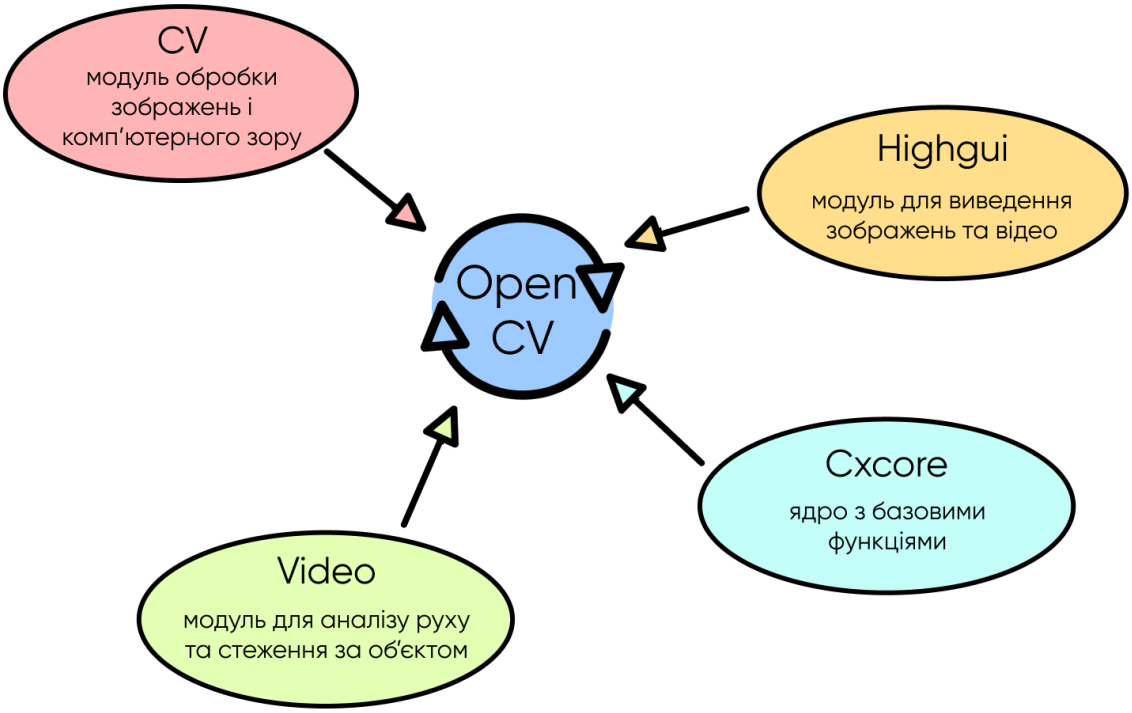

Рисунок 2.1 - Складові та модулі OpenCV

Завдяки модулям бібліотеки OpenCV відкривається багато можливостей для обробки та аналізу зображень та відео. Вона містить лише корисні функції та алгоритми для роботи з зображеннями [10].

2.2 Схема роботи OpenCV

У даній роботі, з початку за допомогою камери, OpenCV бачить кольорове зображення навколишньої середи та очікує потрапляння в поле зору людини. Коли людина потрапляє в кадр, дана програма спрацьовує таким чином, що виділяє людину повністю та підготовлює зображення до сегментації.

Дана бібліотека виводить зображення у сірих кольорах, щоб було легше сегментувати. Далі відбувається безпосередньо сегментування, коли сіре зображення для початку перетворюється на двійкове інвертоване зображення, потім відбувається групування наборів пікселів та інших частин зображення задля спрощення подальшої класифікації зображень та моделювання нейронної мережі.

Коли вже можна визначити саме обличчя людини, то програма обрізає його в заданому розмірі та передає на подальшу обробку. За допомогою OpenCV також ще можна повертати зображення, покращити різкість та змінювати кольори зображення. Це не весь список можливосте бібліотеки, проте саме такі зміни та редагування властивостей зображень є найпоширенішими та найбільш потрібні для системи моніторингу.

Увесь процес, який виконує OpenCV, можна побачити на рисунку 2.2

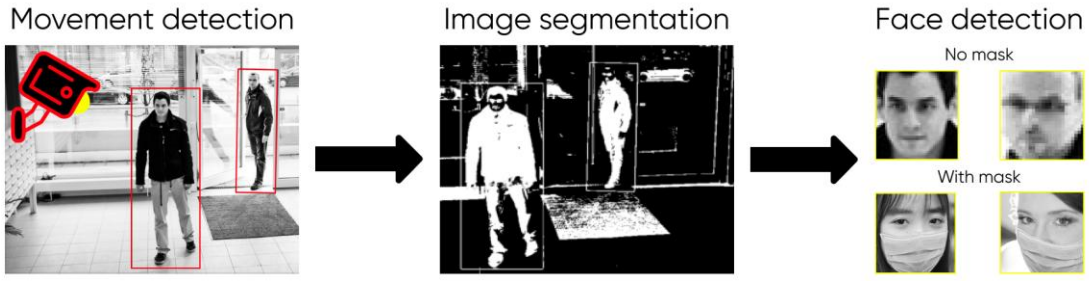

Рисунок 2.2 - Процес роботи бібліотеки OpenCV

Подальша обробка зображень полягає в тому, щоб розрізнити, порівнявши з базовими даними, як саме вдягнена маска та віднести фотографію до однієї з зазначених категорій класів. Коли вже зображення розподілені, то направляються до сховища. Якщо в папках «Маска не прикриває ніс», «Маска не прикриває підборіддя», «Маска одягнена лише на підборіддя» та «Маска взагалі відсутня»

наявні файли – фотографії порушників, то спрацьовує система сповіщення адміністрації про це для подальшого реагування та винесення зауваження або покарання порушнику маскового режиму у даному закладі. Міри, передбачені для застосування до порушників можуть визначатися окремо кожним закладом. Контроль порушень значно знижує ризик захворюваності людей у громадських місцях, а також наявність камер з системою моніторингу маскового режиму звільнює людей від постійного нагляду за всіма відвідувачами будь-якого громадського закладу [11].

РОЗДІЛ 3. ТРЕНУВАННЯ НЕЙРОННИХ МЕРЕЖ

3.1 Задача класифікації

Класифікація – це процедура, задача якої полягає у тому, щоб розділити вибірку об’єктів на підмножини зі заздалегіть визначеними ознаками. Якщо розглядати це з математичної точки зору, то нехай множина пар [об’єкт, клас] - це простір, з невідомою ймовірнісною мірою A. Візьмемо скінченну вибірку (3.1), яка згенерована, опираючись на ймовірнісну міру A:

$$X^k = \{(x_1, y_1), \dots, (x_k, y_k)\} \tag{3.1}$$

Якщо будувати алгоритм, де X наближається до Y, то він зможе класифікувати усі об’єкти x ∈ X.

Класифікація також буває декількох типів:

- Двокласова класифікація – найчастіше використовується для вирішення складних задач, водночас є найпростішим типом класифікації.
- Багатокласова класифікація – застосовується, коли кількість класів вже більша за декулька тисяч. Допомагає вирішувати надскладні задачі та завдання класифікації.
- Непересічні класи – при класифікаціє об’єкт може належати лише до одного класу.
- Пересічні класи – при класифікації об’єкт може належати до двох або більше класів та пересікатися з іншими об’єктами.
- Нечіткі класи – при класифікації невідомо наскільки об’єкт належить до того чи іншого класу, що потребує додаткового визначення.

Вирішити задачу класифікації можна декількома способами, які поділяються на 2 методи: статистичні та методи машинного навчання. До статистичних методів відноситься логістична регресія, дискримінантний аналіз та наївна класифікація. До методів машинного навчання відносять класифікацію за допомогою дерев рішень, алгоритмів покриття, штучних нейронних мереж, опорних векторів, k-найближчих сусідів. У даній роботі використовується метод машинного навчання – класифікація за допомогою штучних нейронних мереж.

У нейронних мережах наявна функція активації, яка визначає вихід вузлів з заданого набору даних, де функція нелінійної активації дозволяє мережі відтворювати складну нелінійну поведінку. Так за допомогою однієї з найчастіше використовуваних функцій, а саме ReLU, нейрон активується тільки тоді, коли вхідний сигнал перевищує поріг. Дана функція найкраща для глибинного навчання, здатна пришвидшити навчання нейронної мережі, а також реалізується достатньо просто через перетворення матриці. Функція ReLU визначена таким чином(3.2), де x – вхідне значення нейрона. Вона також є аналогом напівперіодичного випрямляча у схемотехніці.

$$f(x) = x^+ = \max(0, x) \tag{3.2}$$

Softmax також можна застосувати до вихідного шару для задач багатокласової класифікації. Дана функція є логістичною та застосовується коли є набір значень. Вона найпоширеніша у застосуванні для мшинного навчання. Softmax може використовуватися навійть на найнижчих та найглибших рівнях при тренуванні нейронних мереж для виконання задачі класифікації. Функція Softmax має таку форму (3.3):

$$y_j = \frac{e^{\approx j}}{\sum_{t=1}^{k} e^{\approx t}} \tag{3.3}$$

Функція гарантує, що сума усіх без вийнятку вихідних нейронів дорівнює 1, а також показує, що значення інтервалу [0, 1], відповідного до кожного виходу, є ймовірністю виходу. Результатом виходу є кінцевий прогноз [12].

Для створення нейронної мережі та моделювання від самого початку нам були необхідні дані, які можна використати для побудування моделі. Набір готових даних було взято у Аднане Кабані та його команди. Команда людей взяла декілька тисяч фотографій та на кожну штучно вдягнули маску. Вони також виділили 5 категорій вдягнення маски:

- Маска одягнена правильно
- Маска не прикриває ніс
- Маска не прикриває підборіддя
- Маска одягнена лише на підборіддя

- Маска взагалі відсутня

У кожній категорії знаходиться декілька тисяч фотографій людей з масками вдягнутими по-різному від наймолодшого віку до похилого. На рисунку 3.1 можна побачити який вигляд мають зображення обличь людей зі штучно вдягнутими масками.

Рисунок 3.1 – Штучно вдягнені маски на зображення обличь людей

Але цей набір даних потрібно було вдосконалити для подальшого тренування моделі. Для цього було написано скрипт, у якому прописана функція, що буде обирати задану кількість фотографій з усієї структури папок та розмістить їх в папки з відповідними назвами до тренувального та тестувального набору. Даний скрипт приведено у додатку А.

При запуску скрипту розробник має право обрати кількість зображень, які він бажає взяти до тренувального набору. На рисунку 3.2 можна побачити, що при введені будь-якого числа зображень, програма рандомно обирає їх з папок та відправляє до потрібного набору.

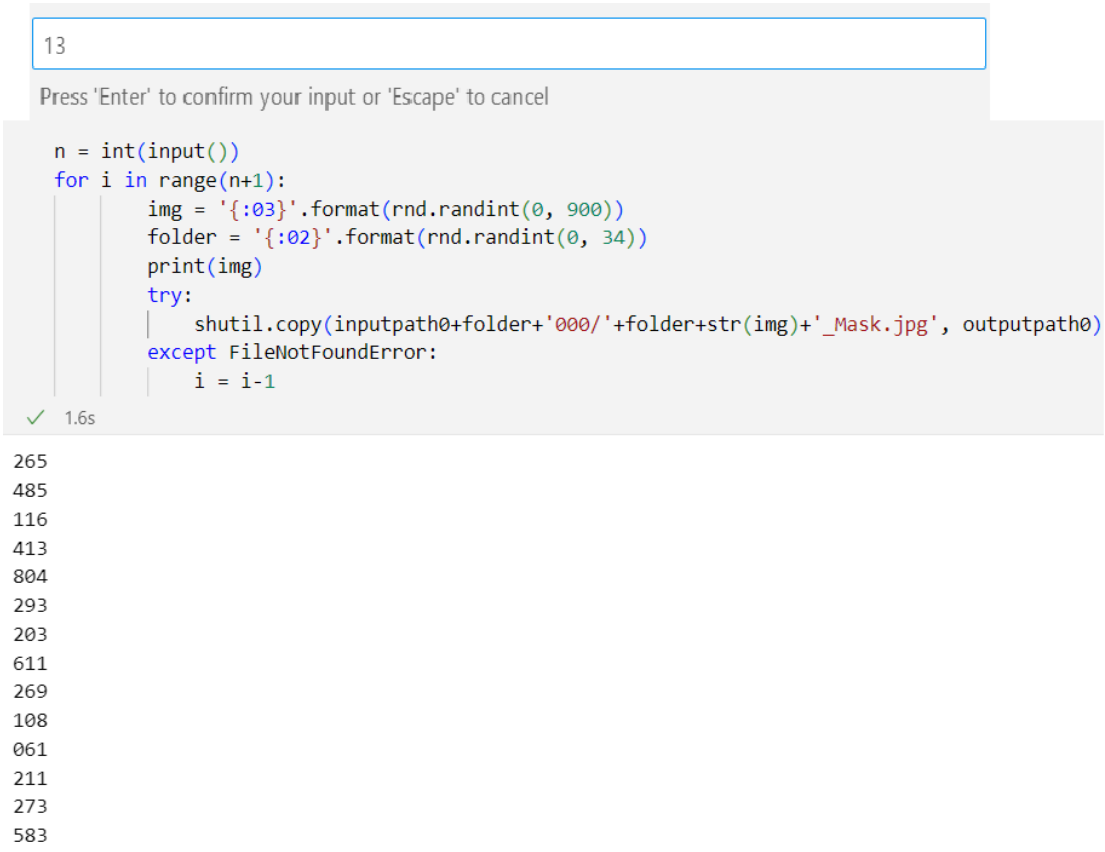

Рисунок 3.2 – Робота скрипту розподілу зображень до тренувального та тестувального набору.

Коли всі дані були підготовлені, то настав етап безпосередньо тренування моделі. Для пошуку параметрів моделей використовувався алгоритм оптимізації – ADAM. За допогою нього у Deep Learning можна швидго досягнути добрих результатів [13].

3.2 Моделі нейронних мереж

Штучні нейронні мережі мають декілька моделей: перша - нейронна мережа прямого поширення, друга - згорткова нейронна мережа, третя - попередньо натренована нейронна мережа. Кожна з них моє свої особливості.

Нейронна мережа прямого поширення – це той вид мережі, де сигнали від вхідного шару нейронів йдуть лише в одному напрямку. Вони можуть проходити навіть приховані шари. Доки сигнал йде від вхідного до вихідного шару, фомується результат роботи сигналу та в кінці людині демонструється звіт. У даній мережі немає шляху в зворотньому напрямку. Нейронна мережа прямого поширення може сама навчатися, розвиватися та згодом навчитися розв’язувати задачі та завдання різного рівня: від простого до дуже складного. Даний тип мережі не часто використовується для роботи із зображеннями.

Яскравим прикладом нейронної мережі прямого поширення може бути перцептрон Розенблатта. Дана модель наглядно демонструє як працює мозок людини та як він сприймає інформацію. Коли перцептрон тільки но з’явився, його навчали розпізнавати літери. Дана нейронна мережа була однією з перших, що взагалі існували, проте вміла швидко навчатися. На випробувальних текстах, нейронна мережа навчилася читати та узагальнувати літери, які бачить, будь то друкований чи написаний від руки текст.

Згорткова нейронна мережа – вона найчастіше застосовується для аналізу зображень, тому що здатна глибоко навчатися та аналізувати. Згорткові нейронні мережі також використовують перцептрони, але зазвичай багатошарові. Завдяки цьому, даний вид мереж майже не вимагає попердньої обробки зображень або тексту. Згорткові мережі мають за основу біологічний процес, при якому враховано як відбувається з'єднання нейронів при утворенні зорової кори у тварин. Принцип такий, що самі нейрони кори зори зазвичай реагують лише на те, що трапляється в обмеженому полі зору. Таке поле нейрона називається рецептивним. Рецептивні поля розташованія таким чином, щоб можливо було покрити всю кору зору. Згорткові мережі, якщо розглядати їх порівнянно з

іншими, найменше потребують попередньої классифікації зображень, обробки, сегментації, тощо. Якщо розглядати згорткові нейронні мережі в порівнянні з мережами прямого пощирення, то саме згорткові потребують достатньої кількості параметрів, які мають передаватися. Ця особливість звісно збільшує час та важкість їх тренування. Це пов’язано з тим, що чим більше невідомих параметрів, тим більше треба данних або навіть наборів даних, використовується більше обчислень, ресурсів, потужностей.

Згорткові нейронні мережі чудово підходять для аналізу відео або зображень, можуть також аналізувати розмову людини та навіть здатні грати в шашки. Мережа може аналізувати зображення, класифікувати отримане та побачене, розпізнавати предмети та людей або їх обличчя на фото та відео. Саме зявдяки згортці – процессу, при якому елементи одного зображення додаються до елементів іншого зображення, дана нейронна мережа здатна обробляти різноманітні фото, бо може зменшити їх, сжати інформацію та розпізнати деталі. Мережа здатна розрізнити контури на зображенні, об’єкти та навіть текстури. Саме згорткова нейронная мережа буде використана у даній роботі, тому що найкраще здатна справлятися з операцією класифікації різноманітних зображень, в тому числі людей та обличь на фото та відео.

Попередньо натренована нейронна мережа може мати два та більше вже безпосередньо згорткових шарів. Це потрібно для того, щоб розділити ознаки, які потрібно побачити та виділита на зображенні. Наприклад, перший шар відповідає за тестуру, другий – за лінії, а третій – за кольори зображення. Попередньо натреновані нейронні мережі також тренуються заздалегіть. Проте таким мережам потрібні дуже великі набори даних, наприклад, підходять ImageNet, MSCOCO, тощо. Перевагою даних мереж є наявніть відкритого коду. Це дає можливість будь-якому розробнику застосувати дану мережу в своїх проектах та програмах, з подальшим поліпшенням та додаванням їй можливостей. Таке використання економить час розробника, бо вже не треба вигадувати код, тренувати модель, залишається лише налаштувати під свої потреби та вдосконалювати.

Попередньо натренована нейронна мережа дає розробнику можливість зробити тонке налаштування (fine tuning) моделі. Воно підходить для того, аби обрати лише декілька шарів для тренування, за автоматичними налаштуваннями, мережа використовує останні шари. Якщо потрібне використання мережі у реальному часі тут і зараз або на особливих пристроях, то дана модель нейронної мережі повинна мати якомога менший розмір та при йьому високу швидкість дії та аналізу даних [14].

В даній роботі було проведено дослідження різних нейронних мереж, для того, щоб отримати ідеальну, натреновану систему моніторингу, взято декілька моделей, щоб віднайти ту, яка найбільш ідеально буде підходити за розміром, точністю та швидкістю тренування. А саме Tiny CNN, ResNet, ResNet (ft), EffNet, EffNet (ft).

Tiny CNN – згорткова нейронна мережа, націлена на ефективне розпізнавання образів. Ідея даної мережі полягає в чергуванні згорткових шарів та шарів підвиборки, що схоже на особливості зорової кори. Схема нейронної мережі представлена на рисунку 3.3.

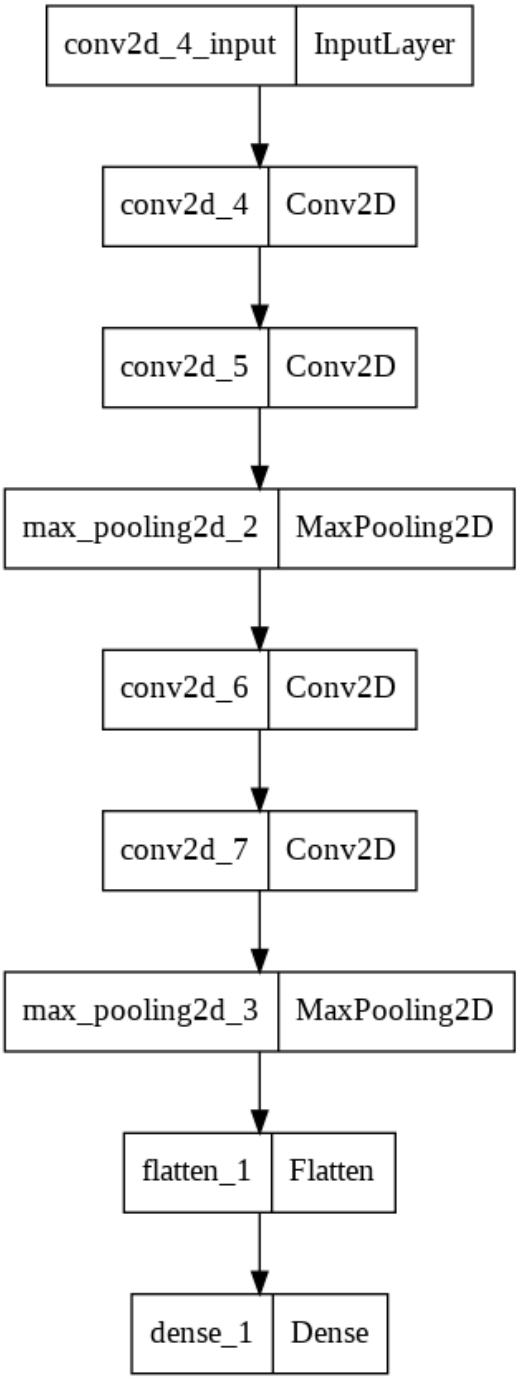

Рисунок 3.3 - Схема нейронної мережі Tiny CNN

ResNet – залишкова згорткова мережа, до скаду якої входять ідентифікаційні ярлики з’єднань, які пропускають навчання декількох шарів, створюючи залишковий блок. Дана модель здатна підтримувати навіть тисячі згорткових шарів.

ResNet найчастіше використовується програмами, що мають зв’язок з комп’ютерним зором. Процесс навчання даної мережі відбувається на основі

зворотнього поширення. Вчені пов’язують цю модель з роботою мозку комах. Ідентифікаційні ярлики з’єднань у ResNet схожі з біологічною системою лічинки плодової мушки. Схема використання даної мережі представлена на рисунку 3.4.

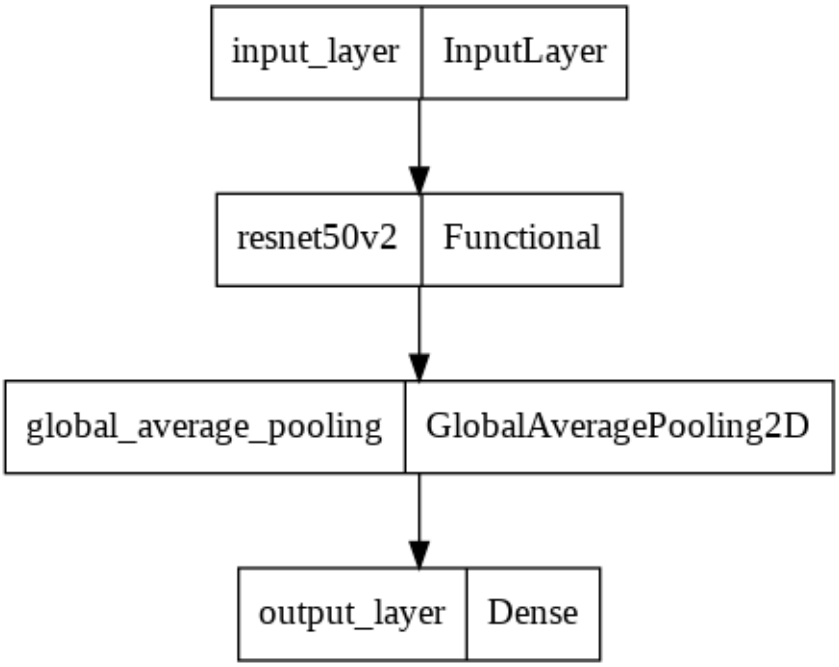

Рисунок 3.4 - Схема згорткової нейронної мережі на основі ResNet.

EffNet - ефективна структура для згорткових нейронних мереж, оптимізована для тонких моделей та створена для вирішення проблем у існуючих моделях. Має новий блок згортки завдяки чому значно зменшує обчислювальне навантаження, перевершуючи поточний стан техніки. Схему даної нейронної мережі можна побачити зображеною на рисунку 3.5 [15].

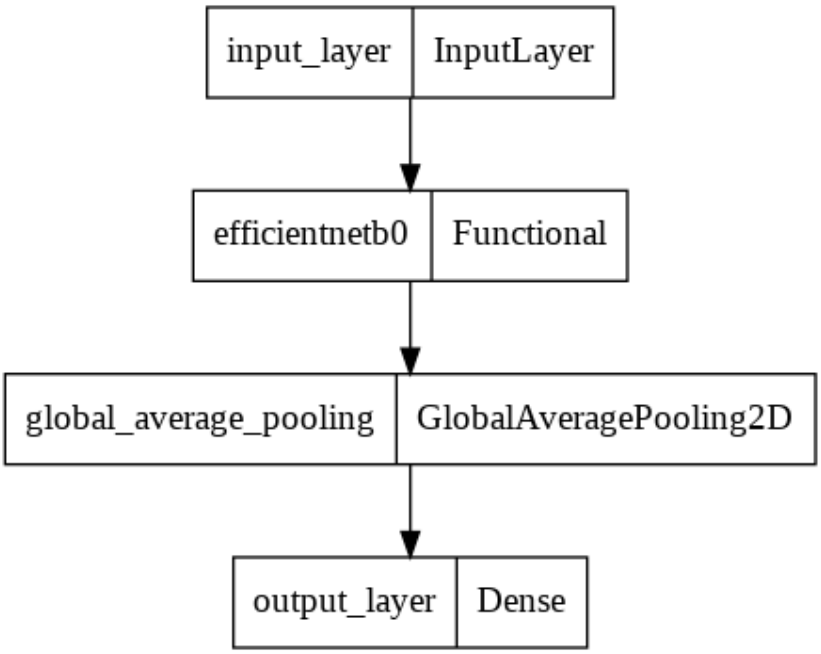

Рисунок 3.5 - Схема згорткової нейронної мережі на основі EfficientNet.

Кожна модель проходила тренування задля виявлення її характеристик та поведінки. Для навчання перш за все використовують функцію витрат C, яка може показати наскільки розв’язок завдання тією чи іншою мережею є оптимальним. Для розрахунку використовують формулу (3.4), яка зазначає, що розв’язок, отриманий в ході роботи тієї чи іншої мережі не повинен витрачати більше ресурсів ніж оптимальний.

$$C(f*) \leq C(f) \forall f \in F \tag{3.4}$$

Як саме проводиться тренування моделі зображено на рисунку 3.6.

```
# Set the seed
tf.random.set_seed(42)

# Preprocess data (get all of the pixel values between 1 and 0, also called scaling/normalization)
# train_datagen = ImageDataGenerator(rescale=1./255)
# valid_datagen = ImageDataGenerator(rescale=1./255)

train_datagen = ImageDataGenerator()
valid_datagen = ImageDataGenerator()

# Setup the train and test directories
train_dir = "train/"
test_dir = "test/"

# Import data from directories and turn it into batches
train_data = train_datagen.flow_from_directory(train_dir,
                                               batch_size=32, # number of images to process at a time
                                               target_size=(224, 224), # convert all images to be 224 x 224
                                               class_mode="categorical", # type of problem we're working on
                                               seed=42)

valid_data = valid_datagen.flow_from_directory(test_dir,
                                               batch_size=32,
                                               target_size=(224, 224),
                                               class_mode="categorical",
                                               seed=42)

# Setup base model and freeze its layers (this will extract features)
base_model = tf.keras.applications.ResNet50V2(include_top=False)

# Setup model architecture with trainable top layers
inputs = layers.Input(shape=(224, 224, 3), name="input_layer") # shape of input image
x=inputs
x = base_model(x, training=False) # put the base model in inference mode so we can use it to extract features without updating the weights
x = layers.GlobalAveragePooling2D(name="global_average_pooling")(x) # pool the outputs of the base model
outputs = layers.Dense(len(class_names), activation="softmax", name="output_layer")(x) # same number of outputs as classes
model1 = tf.keras.Model(inputs, outputs)

# Compile
model1.compile(loss="categorical_crossentropy",
               optimizer=tf.keras.optimizers.Adam(), # use Adam with default settings
               metrics=["accuracy"])

# Fit
history1 = model2.fit(train_data,
                                          epochs=5, # fit for 5 epochs to keep experiments quick
                                          validation_data=valid_data,
                                          validation_steps=int(len(valid_data)))

initial_epoch=5

base_model.trainable = True

# Freeze all layers except for the
for layer in base_model.layers[:-10]:
  layer.trainable = False

# Recompile the model (always recompile after any adjustments to a model)
model1.compile(loss="categorical_crossentropy",
               optimizer=tf.keras.optimizers.Adam(learning_rate=0.00001), # lr is 10x lower than before for fine-tuning
               metrics=["accuracy"])
history1ft=model1.fit(train_data,
              epochs=initial_epoch +5,
              validation_data=valid_data,
              initial_epoch=initial_epoch)
```

Рисунок 3.6 - Тренування моделей для виявлення їхніх характеристик

Спочатку модель оцінюється за графіком, на якому, в ідеалі, лінія тренування та лінія тестування моделі слідкуюдь одна за одною при чому лінія тренування повинна бути вишче за іншу. Вихочи з отриманого спочатку графіка можна побачили, чи модель переоснащена даними чи ні. В залежності від отриманого результату потрібно змінювати фільтри та у згортковій мережі додавати шари до того моменту, доки лінії тренування та тестування моделі на графіку не займуть правильне потрібне положення. Порівняльна таблиця моделей представлена на рисунку 3.7.

	Кількість параметрів моделі	Кількість змінних параметрів моделі	Час тренування 1-ї епохи	Час тренування наступних епох	Точність моделі	Розмір збереженої моделі, Mb
1. CNN	143465	143465	19	18	0,96	1,69
2. ResNet	23575045	3426309	105	78	0,98	90,23
3. ResNet(ft)	23575045	23529605	133	95	0,99	116,4
4. EffNet	4055976	6405	25	19	0,98	15,9
5. EffNet(ft)	4055976	899637	25	20	0,99	22,73

Рисунок 3.7 - Порівняльна таблиця моделей

Після порівняння моделей вже можна обрати одну для подальшої роботи. Згорткова нейронна мережа 5 (EfficientNet + fine tuning) найбільш за все підходить для створення моделі системи моніторингу маскового режиму. EffNet - найефективніша структура саме для згорткових нейронних мереж. Як можна побачити вище, завдяки додатковому блоку згортки, у мережі знижується навантаження. EffNet вміє стискати данні та зображення, завдяки чому може проводити аналіз великих зображень високої якості.

Обравши такий спосіб створення моделі, було проведено всі етапи тренування задля того, щоб модель працювала якомога краще. Результати тренування моделі представлено на рисунку 3.8.

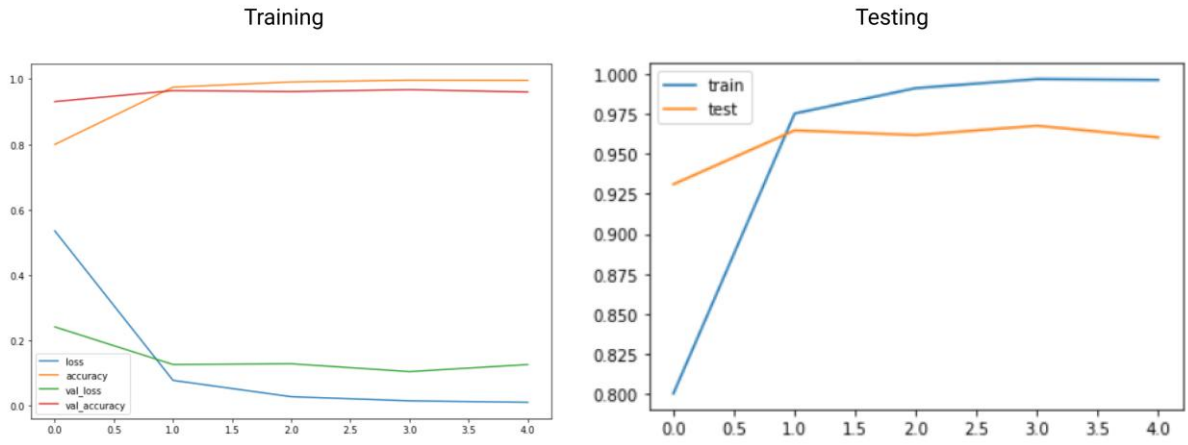

Рисунок 3.8 - Графік тренування моделі для моніторингу

Коли модель була добре натренована, за її допомогою вже можна робити прогнозування на різних інших даних. Прогнозування є важливим процессом у роботі, адже саме завдяки йому можна провести аналіз та оцінку отриманної

інформації, на основі якої в подальшому буде вдосконалюватися робота нейронної мережі та системи в цілому.

Для перевірки роботи системи моніторінгу було взято фотографії деяких людей у правильно і неправильно вдягнених реальних масках та запущено вже натреновану модель на цих даних. Результати неодразу були втішні адже можна було побачити деякі втрати.

Перше, що було зроблено це перевірено зображення на те, що кожне з них приймає потрібну програмі форму та розмір, потім модель було знову натреновано для кращих результатів. Також для отримання позитивного результату тренування у модель було додано деякі функції для перетворення прогнозування.

Після виконання удосконалення, модель запрацювала майже на 100% правильно. Розпізнавання реально вдягнених масок на обличях людей продемонстровано на рисунку 3.9.

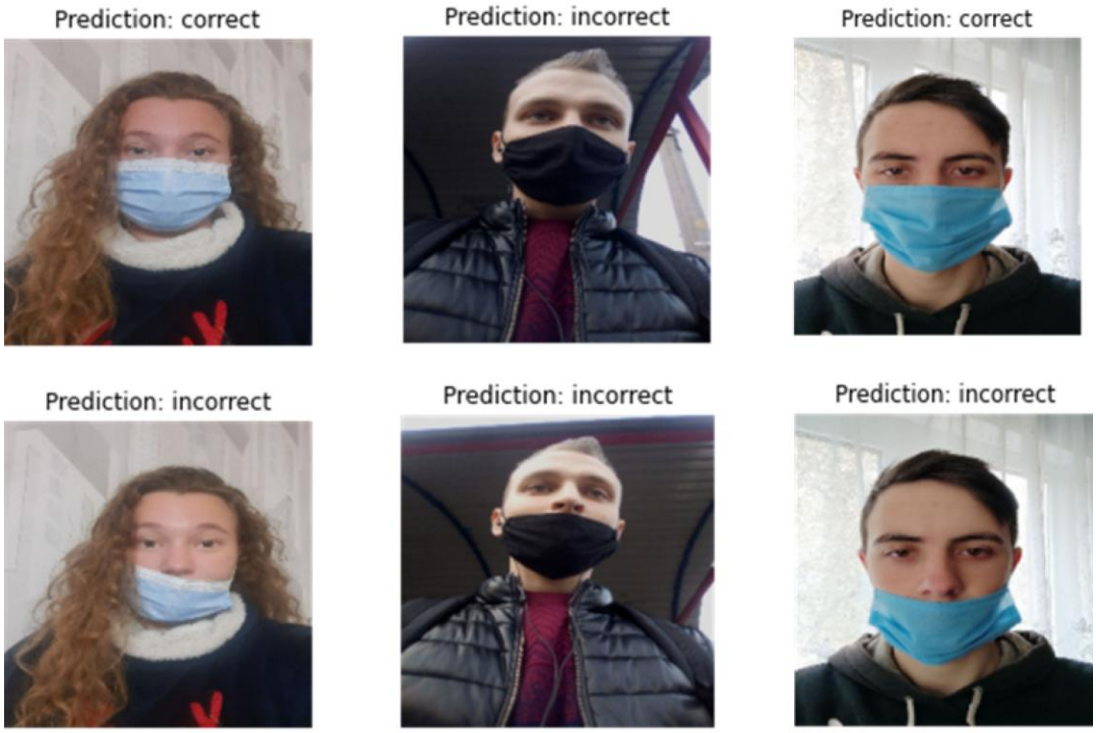

Рисунок 3.9 - Результати розпізнавання реально вдягнених масок

Можна підвести підсумок, що обрана модель згорткової нейронної мережі для розробки системи маскового режиму підходить найкраще, адже вона має

багато переваг. Вони полягають у тому, що модель мала за розміром, має доволі високу точність, а також швидкий процес тренування. Саме згорткова нейронна мережа здатна глибоко аналізувати дані, розпізнавати предмети та людей або їх обличчя на зображеннях. Варто також зазначити, що обрана мережа не потребує великої кількості параметрів, а також має функцію розпаралелювання дій та алгоритмів, що використовуються при тренуванні [16].

3.3 Впровадження системи моніторингу маскового режиму

В результаті попередніх досліджень отримано модель системи моніторингу маскового режиму, яка складається з декількох етапів обробки інформації, починаючи від отримання інформації зі спостереження за навколишнім середовищем і закінчуючи класифікацією зображень обличь людей по заданим класам та сповіщенням адміністрації про порушення маскового режиму у закладі.

Для реалізації отриманих результатів необхідно взяти одноплатний комп'ютер, який мінімальним набором необхідного для роботи може виконати доволі багато різноманітних задач. Такі комп'ютери мають мікропроцесор, оперативну пам'ять, системи вводу-виводу, а також не вимагають встановлення будь-яких додаткових периферійних плат. У одноплатних комп'ютерів немає зайвих функцій та пристроїв, як наприклад, HDMI або аудіо роз'єм. Комп'ютер достатньо легко підключити та налаштувати для роботи у громадському місті в центрі адміністрації.

Як варіант, для впровадження системи моніторингу маскового режиму можна використати одноплатний комп'ютер Raspberry PI. Цей одноплатний комп'ютер побудовано на системі-на-чипі, до якої входить 32-розрядний процесор, який має чотири ядра та тактову частоту 700 МГц. Він також містить графічний процесор, зазвичай це VideoCore IV, а ще має 256 або 512 Гб оперативної пам'яті. Жорсткого диску комп'ютер не має. Замість нього використовується SD карта. Такий склад комп'ютера дозволяє відтворювати відео високого формату, з роздільною здатнвстю 1080p. Графічні можливості Raspberry PI можна прирівняти до продуктивності Xbox 2001 року. Як виглядає одноплатний комп'ютер Raspberry PI можна побачити на рисунку 3.10.

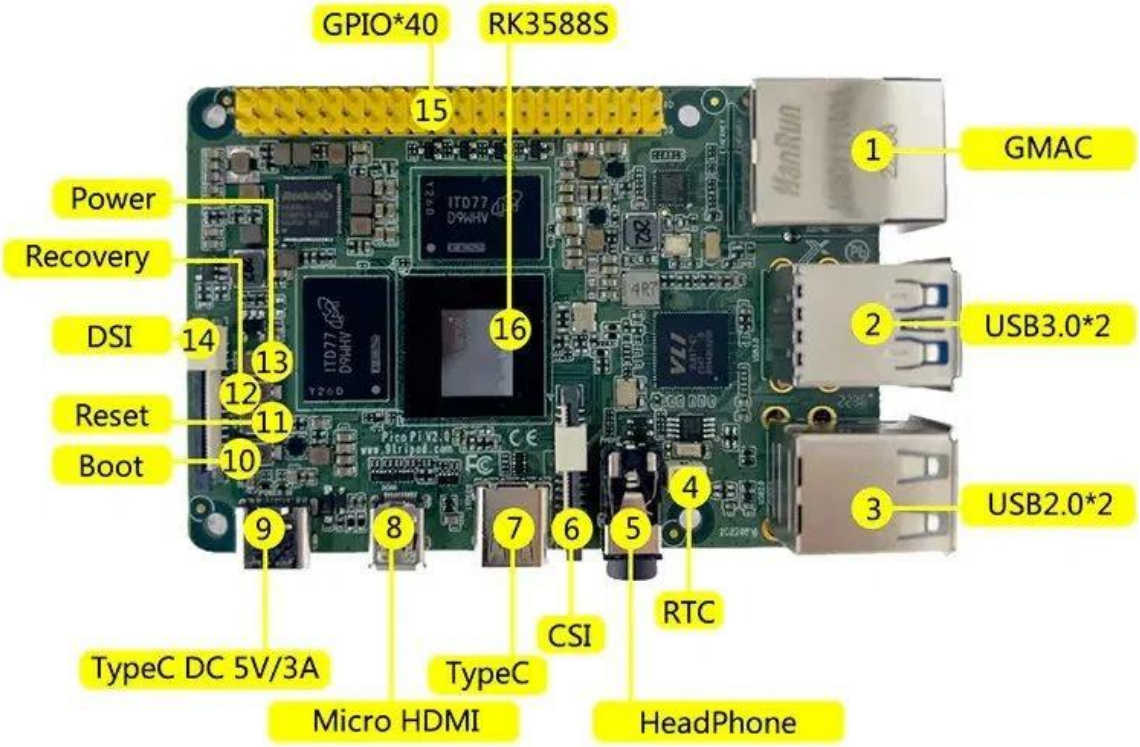

Рисунок 3.10 – Вигляд одноплатного комп’ютера Raspberry PI

Деяка частина систем, що працюють з чипом Raspberry PI, можуть розігнатися до 800 МГц, а деякі з них навіть до 1500 МГц, не враховуючи функції безпеки та обмеження, пов’язані з надвисокою напругою. На Raspberry PI також можливо встановити потрібне програмне забезпечення та навіть операційну систему. Так для створеної у даному проекті системи моніторингу можна встановити Linux систему, а також всі необхідні бібліотеки Python: TensorFlow – для роботи з нейронними мережами, NumPy – для обробки масивів, математичних функцій та виконання операцій над ними, os – для роботи з операційною системою.

Роботу даної системи продемонстровано на рисунку 3.11. Вся інформація з одноплатних комп’ютерів, які встановлені у закладі з потрібним функціоналом для обробки інформації, будуть надсилати отримані дані за допомогою мережі Wi-Fi до центрального головного комп’ютера, на zкому отримані та оброблені зображення будуть зберігатися, а також надсилати сповіщення про порушення маскового режиму.

Рисунок 3.11 - Схема системи моніторингу

Виходячи з характеристик Raspberry Pi можна зробити висновок, що такий одноплатний комп’ютер зможе підійти для установки моделі моніторингу маскового режиму. Адже Raspberry Pi дає змогу встановити усе необхідне програмне забезпечення та доволі швидко виконувати потрібні системі функції [17].

ВИСНОВКИ

Метою роботи було створити систему моніторингу маскового режиму задля того, щоб запобігти розповсюдженню вірусу Covid-19 наскільки це можливо. Для досягнення поставленої мети було розроблено різні підсистеми, використано різні бібліотеки та методи, а також розроблено варіанти впровадження даної моделі системи моніторингу маскового режиму в реальному закладі.

На початку було задіяно камеру відео нагляду, в яку було підключено створену систему моніторингу. Таку камеру можна встановити у будь-якому місці та положенні на території з гарною видимістю.

У камері було використано бібліотеку OpenCV. За допомогою алгоритмів комп'ютерного зору у даній бібліотеці програма робить фіксацію руху на заданій території, потім передає інформацію для розпізнавати обличь усіх людей, які потрапили на зображення. Далі ці фотографії підпадають під форматування за розміром та сегментуванню задля того, щоб на наступних етапах було зручніше та легше розпізнавати чи правильно одягнена маска.

Використання у даній системі бібліотеки TensorFlow, яка призначена для розробки систем з технологіями машинного навчання, поліпшило алгоритм розпізнавання образів та прийняття рішень. Разом з цією бібліотекою було використано бібліотеку Keras, що при суміжному використанні дозволило користуватися спрощеною конструкцією нейронної мережі на подальших етапах моделювання.

Нейронна мережа створена на основі отриманих та оброблених даних, стала спроможною вирішувати питання розпізнання образів, здатна їх класифікувати, провести кластеризацію, зробити прогнозування, а також провести аналіз даних. Створена нейронна мережа пройшла натренована та з її допомогою стало можливо проводити прогнозування та розпізнавання обличь.

Вже класифіковані зображення відправляються системою на головний комп'ютер, де зберігаються у сховищі. Сховище в свою чергу поділено на папки з видами порушень та без порушень.

У подальшому для вдосконалення роботи системи, можна запровадити систему сповіщень, що буде спрацьовувати при потраплянні в папки з порушеннями будь-яких фотографій.

Система моніторингу маскового режиму є дуже потрібною громадським закладам та закладам, з великою скупченістю людей, адже коли знаходиться порушник маскового режиму, який наражає на небезпеку інших відвідувачів, його покарання або зауваження до нього може врятувати життя інших людей, що знаходяться поруч із ним.

ПЕРЕЛІК ПОСИЛАНЬ

1. Саулюс Чаплінскас, Носіння маски знижує ризик зараження коронавірусом до 70%, 2020. URL: https://mind.ua/news/20210579-nosinnya-maski-znizhue-rizik-zarazhennya-koronavirusom-do-70-ekspert (електронний ресурс)
2. Моніторинг IT-інфраструктури. URL: https://it-solutions.ua/servisy/sistemy-monitoringa-i-upravleniya/ (електронний ресурс)
3. Kenneth Dawson-Howe, A Practical Introduction to Computer Vision with OpenCV. Wiley: 1st edition, 2014. 240 с. (книга, один автор)
4. Aurélien Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media: 2nd edition, 2019. 856 с. (книга, один автор)
5. Face detection in Python, 2019. URL: https://api-2d3d-cad.com/python-face-detection/ (електронний ресурс)
6. Python image segmentation. URL: https://www.askpython.com/python/examples/image-segmentation (електронний ресурс)
7. Detecting and recognizing faces in photos with OpenCV and python, 2021. URL: https://www.analyticsvidhya.com/blog/2021/06/learn-how-to-implement-face-recognition-using-opencv-with-python/ (електронний ресурс)
8. Kenneth Dawson-Howe, A Practical Introduction to Computer Vision with OpenCV. Wiley: 1st edition, 2014. 240 с. (книга, один автор)
9. Alberto Fernandez Villan, Mastering OpenCV 4 with Python. Packt Publishing, 2019. 532 с. (книга, один автор)
10. Jeremy Howard, Sylvain Gugger, Deep Learning for Coders with Fastai and PyTorch: AI Applications Without a PhD. O'Reilly Media: 1st edition, 2020. 621 с. (книга, 2 автори)
11. MaskedFace-Net . URL: https://github.com/cabani/MaskedFace-Net (електронний ресурс)

12. TensorFlow for Deep Learning. URL: https://dev.mrdbourke.com/tensorflow-deep-learning/ (електронний ресурс)
13. Samuel Burns, Python Deep learning: Develop your first Neural Network in Python Using TensorFlow, Keras, and PyTorch. Independently published, 2019. 170 с. (книга, один автор)
14. TensorFlow Hub. URL: https://tfhub.dev/ (електронний ресурс)
15. EfficientNetV2: Smaller Models and Faster Training. URL: https://arxiv.org/abs/2104.00298v2 (електронний ресурс)
16. TinyCNN: A Tiny Modular CNN Accelerator for Embedded FPGA. URL: https://arxiv.org/abs/1911.06777 (електронний ресурс)
17. Raspberry Pi, 2021. URL: https://uk.wikipedia.org/wiki/Raspberry_Pi (електронний ресурс)

ДОДАТКИ

Додаток А

```
n = int(input())
for i in range(n+1):
        img = '{:03}'.format(rnd.randint(0, 900))
        folder = '{:02}'.format(rnd.randint(0, 34))
        print(img)
        try:
          shutil.copy(inputpath0+folder+'000/'+folder+str(img)+'_Mask.jpg', outputpath0)
        except FileNotFoundError:
           i = i-1
n = int(input())
for i in range(n+1):
        img = '{:03}'.format(rnd.randint(0, 900))
        folder = '{:02}'.format(rnd.randint(0, 34))
        print(img)
        try:

shutil.copy(inputpath1+folder+'000/'+folder+str(img)+'_Mask_Mouth_Chin.jpg',
outputpath2)
        except FileNotFoundError:
           i = i-1
        try:
           shutil.copy(inputpath1+folder+'000/'+folder+str(img)+'_Mask_Chin.jpg',
outputpath1)
        except FileNotFoundError:
           i = i-1
        try:
```

```
shutil.copy(inputpath1+folder+'000/'+folder+str(img)+'_Mask_Nose_Mouth.jpg',
outputpath3)
        except FileNotFoundError:
            i = i-1
```