КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра інтелектуальних технологій

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня «магістр»

НА ТЕМУ:

Розробка інтелектуального модуля системи ідентифікації осіб без засобів індивідуального захисту

Галузь знань: 12 «Інформаційні технології»
Спеціальність: 122 «Комп'ютерні науки»
Освітньо-наукова програма «Технології штучного інтелекту»

Виконав:
студент 2 курсу магістратури,
групи ТШІ-21 Голубчиков Павло Сергійович

Науковий керівник: Кудін Володимир Іванович

Доктор технічних наук,
професор

Засвідчую, що в цій кваліфікаційній роботі немає запозичень з праць інших авторів без відповідних посилань

Студент _____
підпис

Кваліфікаційна робота допущена до захисту
рішенням кафедри *інтелектуальних технологій*

Протокол № ____ від «____» травня 2021 р.

Зав. кафедри доц. Іларіонов О.Є.
підпис

Київ 2021

Вступ

У грудні 2019 року в китайському місті Ухань спалахнула нова інфекційна первинна атипова (вірусна) пневмонія. Пізніше було встановлено, що нове захворювання, яке отримало назву COVID-19, викликано раніше невідомим зоонозним коронавірусом під назвою SARS-CoV-2. Щоб забезпечити розповсюдження цього нового коронавірусу, Всесвітня організація охорони здоров'я (ВООЗ), медичні експерти, а також уряди всього світу тепер рекомендують людям носити маски для обличчя, якщо у них є респіраторні симптоми, якщо вони дбають про людей з симптомами. або іншим чином часто взаємодіють з великими групами людей. У відповідь на це, розробки та дослідження по виявленню лицьових масок нещодавно привернули увагу багатьох вчених та розробників, особливо тих, хто пов’язан з комп'ютерним зором. Було написано багато статей та розроблено декілька систем автоматичного виявлення людей без медичних масок, але їх точність досить невелика, що може призводити до багатої кількості помилкових спрацювань, що є найголовнішою проблемою. Також системи, які розроблені на сьогоднішній день, не мають можливості саме ідентифікувати особу без медичної маски.

Тенденція носіння масок в громадських місцях росте через епідемію коронавіруса COVID-19 в усьому світі. У багатьох країнах за законом люди змушені носити маски на публіці. Ці правила і закони були розроблені як дія, спрямована на експоненціальне зростання числа випадків і смертей. Однак процес спостереження за великими групами людей в громадських місцях стає все важче. Отже, було прийнято рішення створити таку систему, яка зможе в цьому допомагати. Модель може бути інтегрована з камерами спостереження, щоб у реальному часі виявляти людей, які не носять медичних масок.

Метою даної роботи є створення системи, яка буде з максимальною точністю визначати чи є маска на обличчі та у разі відсутності автоматично відправляти фото порушника до відділу охорони в будь-який месседжер або на email.

Об’єкт дослідження – процес ідентифікації людей без медичних масок.

Предмет дослідження – модуль системи ідентифікації людей без медичних масок.

Наукова новизна роботи:

- Реалізовано систему, яка в автоматичному режимі відправляє фото людей без засобів індивідуального захисту до спеціального каналу охоронців у месседжері Microsoft Teams, що дозволяє швидко прийняти адміністративні міри до цієї особи.
- Досягнуто досить високу точність, що дозволяє мінімізувати помилки в результаті роботи системи.

Практичним результатом роботи є розробка системи, яка в реальному часі за допомогою методів глибинного навчання та згорткових нейронних мереж, буде визначати наявність на обличчі людині маски, та у разі її відсутності надсилати зображення порушника до спеціального чату.

Аналітичний огляд процесу ідентифікації осіб з використанням методів глибинного навчання

1.1 Аналіз сучасного стану процесу розпізнавання засобів індивідуального захисту з використанням нейронних мереж

Одним з найбільщ інтенстивно досліджуваним напрямом у сфері ІТ на сьогоднішний день є штучний інтелект (ШІ). Застосування ШІ дозволяє розв'язувати такі задачі, які без нього взагалі вирішити було неможливо, або ї\ розв'язання було вкрай трудомістким та неефективним, що зводило нанівець такий варіант роз'язку. Гарним прикладом застосування ШІ є, скажімо, системи ідентифікації осіб в аеропортах чи на вокзалах, які за досить короткий проміжок часу зуміли показати себе з найкращих сторін, допомогаючи ловити злочинців, ідентифікувати свідомо підозрілих осіб та багато іншого.

Однією з найбільш звичних і розвинених технологій з використанням ШІ є комп'ютерний зір. Основним його призначенням є те, що він моделює властивості людського зору, таким чином має змогу розпізнавати та ідетифікувати людей, розпізнавати образи та дрібні деталі на кожному зображенні або навіть відеопотоці, таким чином отримуючи максимальну кількість можливої інформації. Все це відбувається за допомогою моделей глибинного навчання. Моделі дозволяють вирішувати велику кількість задач та, насамперед, найчастіше вони використовуються в системах розпізнавання облич. Не секрет, що такі системи зараз використовуються в усіх сферах життя, починаючи з аеропортів та вокзалів, як було зазначено вище, та закінчуючи розблокуванням телефону. Це вже взагалі стає непомітним та люди не приділяють цьому ніякої уваги, але кожного дня продовжують цим користуватись. Саме цьому ця сфера продовжує з кожним днем розвиватися, вирішуючи все різні й різні задачі, наприклад, такі як розпізнавання людини в захисній масці. Саме на цій проблемі й хотілося зусередитися більш детально.

Розпізнавання маски на обличчі людини представляє собою як, в першу чергу, проблему, взагалі, розпізнавання людини та її обличчя та задачу класифікації, оскільки спочатку треба знайти на зображеннях або відео людину, а вже потім приймати рішення, носить вона маску чи ні. Перша частина цієї задачі вже широко вивчена та можно знайти багато літератури з комп'ютерного зору, яка буде присвячена цій проблемі завдяки широкій застосовності технології виявлення обличчя. Але щодо другої частини, то тут матеріалу набагато менше та сама проблема ще майже не вивчена. Незважаючи на те, що за останній рік було проведено декілька робот, пов'язаних з цією проблемою, вони, як правильно, намагаються лише виявити чи присутня на зображенні маска. Особливої уваги не приділяється тому, чи правильно розташована маска на обличчі і, взагалі, чи носиться вона за рекомандаціями МОЗ та так далі. Це обмежує прикладну цінність існуючих методів та вимагає дослідження та створення таких моделей комп'ютерного зору, які будуть здатні не тільки виявити наявність маски на обличчі, але й визначити, чи правильно маски носяться.

Ще півтора роки тому, майже не було таких моделей, які би визначали, чи знаходиться людина в захисній масці, чи ні. Зараз ця проблема є доволі великої та значною, особисто для таких сфер життя як громадський транспорт або, наприклад, офісні приміщення та багато іншого. На сьогоднішний день такі моделі вже існують та вже запроваджені в деяких країнах, але точніть цих моделей залишає бажати кращого, насамперед тому, що немає великої кількості данних та немає добре структурованого датасету, за домогою якого можна було би навчити модель.

Тому основною метою даної роботи є створення такої системи, точність якої була би максимально можливою, не менш ніж 90%, та, як зазначалося вище, щоб модель не тільки визначала чи є маска, а й ще визначала її розташування. Адже у цій задачі точність є основним показником якості.

1.2 Аналіз існуючих моделей та систем

Виявлення маски на обличчі є складним завданням. У цю епоху їй приділяється дедалі більше уваги через поширення вірусу короновірусу. Звідси багато країн дотримуються правила на кшталт «Немає в'їзду без маски». Виявлення маски для обличчя є дуже важливим питанням з метою безпеки та запобігання дії Covid-19. Що стосується галузі медицини, маска зменшує потенційний ризик зараження інфікованою людиною незалежно від того, чи є у них симптоми чи ні. Розпізнавання маски для обличчя доцільно було б використовувати в аеропортах, лікарнях, офісах та навчальних закладах тощо.

Проблема розпізнавання маски на обличчі вже вирішувалась та існує деяка кількість розроблених методів для її вирішення, але у кожному з них є свої певні недоліки. Спочатку цю проблему вирішували за допомогою традиційних методів розпізнавання об’єктів, таких як алгоритм Віоли-Джонса, алгоритм адаптивного бустінгу та гістограми направлених градієнтів. Нижче наведені останні з запропонованих методів.

У 2012 році виявлення обличчя за допомогою згорткових мереж та фільтрів Габора [1], запропоноване Боданом Кволеком, використовувалося для виявлення областей обличчя шляхом складання фільтрів Габора та згорткової нейронної мережі. Фільтр Габора зосереджений на витягуванні внутрішніх рис обличчя. Основними перевагами фільтра Габора є можливість проводити аналіз сигналів в різних масштабах та роздільній здатності. Згорнутий шар нейронної мережі складається з однієї або декількох площин, в цьому методі використано 6 згорткових нейронних мереж. В результаті метод показав, що він забезпечує краще розпізнавання та високу швидкість розпізнавання обличчя, ніж поодинці CNN.

У 2017 році за допомогою каскадної структурі, запропонованої Вейбу Цзяньцзіньнь Сяо та Чуанхон Чжоу, була використана проста система виявлення маски. Архітектура складається з каскадних 3 згорткових детекторів масок: Mask – 12, Mask – 24-1 та Mask - 24-2. Тут використовується згорнутий шар ResNet 5

model – 7 з наступним шаром об’єднання. Маска 1 - це перша стадія, а Маска 3 - остання стадія детектора обличчя з маскою. Використовується набір даних із замаскованим обличчям, який містить 160 зображень для тренування та 40 зображень для тестування. Навчальний процес включає в себе моделі попередньої підготовки та моделі настройки. Для оцінки якості використовувався набір даних PASCAL VOC. Тестування на замаскованому обличчі досягло точності 86,6%.

У 2017 році з’явилося виявлення та сегментація обличчя на основі вдосконаленої Mask R-CNN [6], запропонованої Кайхан Лінь та Сяоюн Лю. Зокрема, мережа з архітектурою Mask R-CNN дозволяє виділяти на фотографіях контури («маски») екземплярів різних об'єктів, навіть якщо таких екземплярів кілька, вони мають різний розмір і частково перекриваються. Використовувалися популярні набори даних тестування обличчя, набори даних FDDB (Face Detection Data Set and Benchmark) та AFW. Для створення маски використовалася повністю згорнута мережа шарів, за якою йде функція максимума (max pooling шар). В результаті це дало досить високу точність у 87%.

У 2019 р. впровадження Методу головних компонент для виялення маски на обличчі. Метод запропонували проф. Саббір Еджаз та Рабіул Іслам [8]. Використовуваний набір даних - це датасет для розпізнавання облич Olivetti та датасет з Oracle Research Laboratory (ORL). Етапи, які використовуються в цій роботі, включають отримання зображень обличчя та вилучення особливостей обличчя за допомогою МГК та обчислення власних векторів. В результаті це дає високий рівень розпізнавання на обличчях без маски.

У 2020 році оцінка роботи інтелектуальної системи виявлення маски для обличчя з різними класифікаторами глибокого навчання [10], запропонована К. Джагадесварі, М. Удай Теджа. Тут ефективність виявлення маски на обличчі може бути проаналізована за допомогою різних класифікаторів глибокого навчання, таких як mobileNet V2, ResNet 50, VGG 16, ADAM, SGD. Це класифікатори, що використовалися для цього. Для кожного класифікатора застосовувалося декілька оптимізаторів та для кожного оцінювалася

ефективність. Тут використовувалися такі оптимізатори як ADAM, ADAGRAD, SGD (стохастичний градієнтний спуск). Як результат, ефективність оптимізатора ADAM є найкращою, а також відзначено, що класифікатор MobileNet V2 має високі результати з досить непоганою точністю.

1.3 Дослідження застосування методів штучного інтелекту в системах розпізнавання облич

Системи розпізнавання облич - це технологія, здатна ідентифікувати або перевірити особу на цифровому зображенні або відеокадрі. Існує багато методів, які використовуються в системах розпізнавання облич, але в цілому вони ґрунтуються на порівнянні рис заданого обличчя з обличчями, які зберігаються в базі даних. Технологія також описується як біометричний додаток на основі штучного інтелекту, який може однозначно ідентифікувати людину шляхом аналізу моделей на основі текстур обличчя та форми людини. []

Протягом кількох останніх десятиліть було запропоновано та впроваджено багато методів розпізнавання облич, але не всі вони якісно себе показали. Деякі отримували замалу точність розпізнавання, деякі розпізнавали те, що обличчям не було. Хотілося б подовше зусередитися на основних методах, які показали себе якісніше за інших.

1.3.1 Метод головних компонент

Першим таким методом є метод головних компонент (МГК). Це один із найуспішніших методів, які використовувались при розпізнаванні та стисненні зображень. Основною метою цього методу є зменшення великої розмірності шляхом втрати незначної кількості даних. Основна ідея використання МГК для розпізнавання обличчя полягає у вираженні великого одновимірного вектора пікселів, побудованих із двовимірного зображення обличчя, у компактні основні компоненти простору зображень. Це можна назвати проекцією власного простору. Власний простір обчислюється шляхом ідентифікації власних векторів матриці коваріації, отриманих із набору зображень обличчя (векторів).

Двовимірне зображення обличчя може бути представлене як одновимірний вектор шляхом об'єднання кожного рядка (або стовпця) у довгий вектор. Припустимо, є M векторів розміром N (рядки зображення × стовпці зображення), що представляють набір вибіркових зображень. p_j представляють значення пікселів.

$$x_i = [p_1 \dots p_j]^T, i = 1, \dots, M \quad (1.1)$$

МГК обчислює основу простору, який представлений його тренувальними векторами. Ці базисні вектори, власне власні вектори, розраховані за допомогою МГК, спрямовані на найбільшу дисперсію тренувальних векторів. Кожну власну грань можна переглянути як особливість. Коли конкретне обличчя проектується на простір обличчя, його вектор у просторі обличчя описує важливість кожної з цих особливостей обличчя. Грань виражається в особовому просторі його власними коефіцієнтами (або вагами). Ми можемо впоратися з великим вхідним вектором, лише взявши його малий вектор ваги в лицьовому просторі. Це означає, що ми можемо реконструювати вихідне обличчя з певною помилкою, оскільки розмірність простору зображення набагато більша, ніж розмір обличчя.

1.3.2 Лінійний дискримінантний аналіз

Другим методом є лінійний дискримінантний аналіз (ЛДА). ЛДА шукає проекційні осі, на яких точки даних різних класів знаходяться далеко одна від одної, в той момент, коли точки даних одного класу повинні бути близькими одна до одної. На відміну від МГК, який кодує інформацію в ортогональному лінійному просторі, ЛДА кодує дискримінаційну інформацію в лінійно відокремленому просторі, використовуючи базиси, які не обов'язково є ортогональними. Зазвичай вважається, що алгоритми, засновані на ЛДА, перевершують алгоритми, засновані на МГК. Однак деякі роботи показали, що коли набір навчальних даних невеликий, МГК може перевершити ЛДА, а також МГК менш чутливий до різних наборів даних навчання.

ЛДА тісно пов'язаний з дисперсним аналізом і регресійний аналізом, які також намагаються виразити будь-яку залежну змінну через лінійну комбінацію інших ознак або вимірювань. У цих двох методах залежна змінна - чисельна величина, а в ЛДА вона є величиною номінальної (міткою класу).

В основі дискримінантного аналізу лежить припущення про те, що опис об'єктів кожного k-го класу є реалізацією багатовимірної випадкової величини, розподіленої за нормальним законом $N_m(\mu_k; \sum_k)$ з середнім μ_k та матрицею коваріації:

$$C_k = \frac{1}{n_k - 1} \sum_{i=1}^{n_k} (x_{ik} - \mu_k)^T (x_{ik} - \mu_k) \quad (1.2)$$

Індекс m вказує на розмірність простору ознак.

Розглянемо кілька спрощену геометричну інтерпретацію алгоритму ЛДА для випадку двох класів. Нехай дискримінантні змінні x - осі m-мірного евклідового простору. Кожен об'єкт (спостереження) є точкою цього простору з координатами, що представляють собою фіксоване значення кожної змінної. Якщо обидва класи відрізняються один від одного по спостережуваним змінним, то їх можна представити як скупчення точок в різних областях розглянутого простору, які можуть частково перекриватися. Для визначення положення кожного класу можна обчислити його "центр ваги", який є уявної точкою, координатами якої є середні значення змінних в даному класі.

Завдання дискримінантного аналізу полягає в проведенні додаткової осі z яка проходить через множину точок таким чином, що проекції на неї забезпечують найкраще розбиття на два класи. Її положення задається лінійної дискримінантної функцією (linear discriminant, LD) з ваговими коефіцієнтами β_j

що визначає внесок кожної вихідної змінної x_j

$$z(x) = \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_m x_m \quad (1.3)$$

У системі розпізнавання облич кожне обличчя представлено великим числом значень пікселів. Лінійний дискримінантний аналіз застосовується тут головним чином для скорочення числа ознак до більш керованого числу перед спробою класифікації. Кожна з нових розмірностей є лінійної комбінацією значень пікселів, утворюючи шаблон. Лінійні комбінації, отримані використанням лінійного дискримінанту Фішера, називаються обличчами Фішера, в той час як комбінації, отримані за допомогою методу головних компонент, називаються власними обличчами

1.3.3 Метод незалежних компонент

Ще одним методом є метод незалежних компонент (МНК). Застосовуючи МГК до набору зображень обличчя, ми знаходимо набір базисних векторів, використовуючи статистику нижчого порядку відношень між пікселями. Зокрема, ми максимізуємо дисперсію між пікселями, щоб відокремити лінійні залежності між пікселями. МНК є узагальненням МГК, оскільки він намагається ідентифікувати статистичні зв'язки високого порядку між пікселями, щоб сформувати кращий набір базисних векторів. Подібно до МГК та ЛДА, як тільки будуть знайдені нові базисні вектори, дані навчання та тестування проектуються в підпростір і для класифікації використовується такий метод, як нейронні мережі про які піде мова далі.

Метод незалежних компонент заснований на тому, що записані n сигналів є лінійною комбінацією m невідомих базових сігналов. Математична модель ICA в разі, якщо $A_{n\times m}$ - матриця з елементами $n \times m$, а x - вектор рядка, виглядає наступним чином. Нижче представлені формули перетвореного базового сигналу.

$$x_j^*(k) = \sum_{i=1}^{n} a_{ji}\, s_i^*(k) + \varepsilon_j(k), \quad (1.4)$$

де k=1,…,m

$$x^*(k) = \mathrm{A}s^*(k) + \varepsilon(k), \quad (1.5)$$

$$\text{де } s^*(k) = [s_1^*(k), s_2^*(k), \dots, s_n^*(k)]^T - \text{незалежний базовий сигнал}, \quad (1.6)$$

$$a \quad \varepsilon(k) = \left[\varepsilon_{1(k)}, \varepsilon_{2(k)}, \dots, \varepsilon_{n(k)}\right]^T - \text{аддитивний шум} \quad (1.7)$$

та $A_{n\times m}$ - шукана матриця

1.3.4 Нейронні мережі

Основним та найпопулярнішим на сьогоднішній день методом є нейронні мережі.

Нейронні мережі, також відомі як штучні нейронні мережі (ШНМ), є підмножиною машинного навчання і лежать в основі алгоритмів глибокого навчання. Їх назва та структура натхненні людським мозку, імітуючи спосіб сигналізації біологічних нейронів один одному.[12]

Штучні нейронні мережі (ШНМ) складаються з шарів вузлів, що містять вхідний шар, один або кілька прихованих шарів та вихідний шар. Кожен вузол, або штучний нейрон, з'єднується з іншим і має відповідну вагу та поріг. Якщо вихід будь-якого окремого вузла перевищує вказане порогове значення, цей вузол активується, надсилаючи дані на наступний рівень мережі. В іншому випадку дані не передаються на наступний рівень мережі.[12]

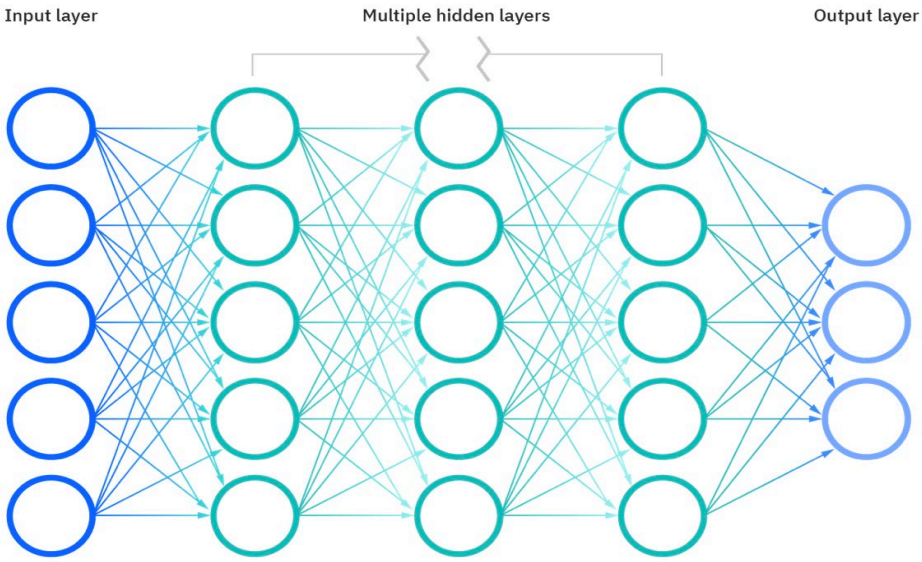


Рисунок 1.1. Архітектура нейронної мережі

Нейронні мережі покладаються на дані навчання, щоб навчатися та підвищувати їх точність з часом. Однак, як тільки ці алгоритми навчання правильно налаштовані на отримання максимальної точності, вони стають потужними інструментами в галузі інформатики та штучного інтелекту, що дозволяє класифікувати та кластеризувати дані з великою швидкістю. Завдання

з розпізнавання зображень можуть зайняти хвилини проти годин у порівнянні з ідентифікацією, проведеною людьми експертами.

Представимо кожен окремий вузол як власну модель лінійної регресії, що складається з вхідних даних, вагових коефіцієнтів, зміщення (або порогового значення (bias)) та вихідних даних. Формула виглядала б приблизно так:[13]

$$\sum_{i}^{m} w_i x_i + bias = w_1 x_1 + w_2 x_2 + w_3 x_3 + bias \quad (1.8)$$

$f(x) = 1$, якщо $\sum w_i x_i + b \geq 0$;

$f(x) = 0$, якщо $\sum w_i x_i + b < 0$.

Після визначення вхідного рівня призначаються ваги. Ці ваги допомагають визначити важливість будь-якої даної змінної, причому більш вагоміші вносять більший внесок у результат порівняно з іншими вхідними даними. Потім усі вхідні дані множать на відповідні ваги, а потім підсумовують. Потім вихід передається через функцію активації, яка визначає вихід. Якщо цей вихід перевищує заданий поріг, він «спрацьовує» (або активує) вузол, передаючи дані наступному шару в мережі. Це призводить до того, що вихід одного вузла стає входом наступного вузла. Цей процес передачі даних з одного рівня на наступний рівень визначає цю нейронну мережу як мережу прямого зв’язку.

Якщо використовувати більш практичні випадки нейронних мереж, такі як розпізнавання облич або класифікація, вже буде застосовуватися навчання з учителем або розмічені набори даних для навчання алгоритму. Навчаючи модель, хочеться оцінити її точність, використовуючи функцію витрат (або збитків). Це також зазвичай називають середньоквадратичною помилкою (MSE). У наведеному нижче рівнянні:

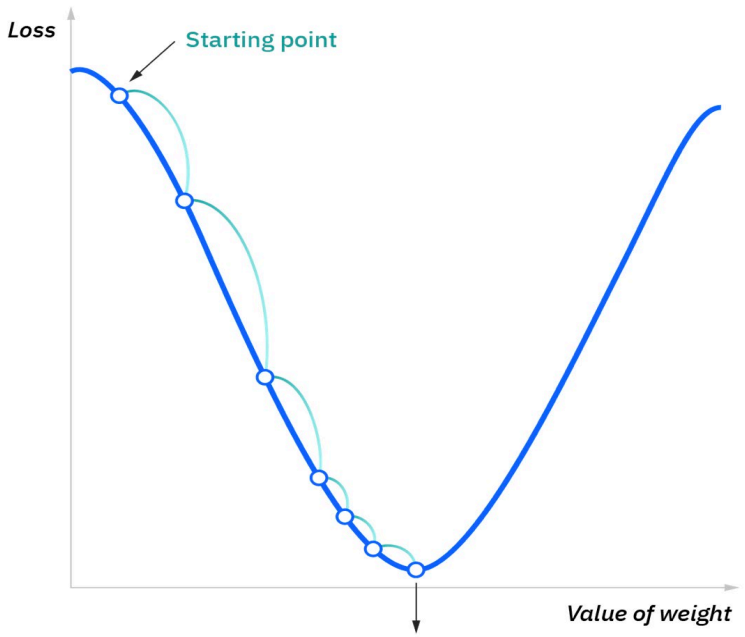
i - являє собою індекс вибірки,

y' - прогнозований результат,

y - фактичне значення,

m - кількість зразків.

$$CostFunction = MSE = 1/(2m) \sum_{i}^{m} (y' - y)^2 \quad (1.9)$$



Точка збіжності, тобто де функція витрат досягає мінімума

Рисунок 1.2. Функція витрат

Зрештою, мета - мінімізувати функцію витрат, щоб забезпечити правильність застосування для будь-якого даного спостереження. Оскільки модель коригує свої ваги та упередження, вона використовує функцію витрат, щоб досягти точки збіжності, або локального мінімуму. Процес, в якому алгоритм регулює свої ваги, здійснюється за допомогою градієнтного спуску, що дозволяє моделі визначити напрямок, щоб рухатись, щоб зменшити помилки (або мінімізувати функцію витрат). З кожним навчальним прикладом параметри моделі регулюються, щоб поступово збігатися до мінімуму.

Більшість глибинних нейронних мереж є прямими, тобто вони рухаються лише в одному напрямку, від входу до виходу. Однак також можна навчити модель шляхом зворотного розповсюдження; тобто рухатися в протилежному напрямку від виходу до входу. Зворотне розповсюдження дозволяє розрахувати відносну помилку, пов'язану з кожним нейроном, що дозволяє належним чином налаштувати та підігнати параметри моделі.

Нейронні мережі можна класифікувати на різні типи, які використовуються для різних цілей. Незважаючи на те, що це не повний перелік типів, наведені нижче варіанту представляють найпоширеніші типи нейронних мереж, з якими найчастіше зустрічаються:

• Персептрон - найстаріша нейронна мережа, створена Френком Розенблаттом у 1958 році. Вона має єдиний нейрон і є найпростішою формою нейронної мережі.[14] (Рис 1.3)

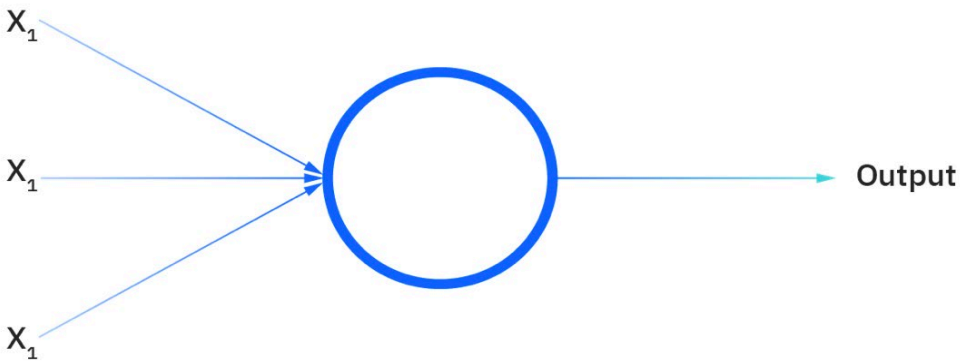


Рисунок 1.3 Одношаровий персептрон

• Нейронні мережі прямої передачі або багатошарові персептрони - вони складаються з вхідного шару, прихованого шару або шарів та вихідного шару. Хоча ці нейронні мережі також часто називають багатошаровим персептроном, важливо зазначити, що вони насправді складаються з сигмоподібних нейронів, а не персептронів, оскільки більшість реальних проблем є нелінійними. Дані зазвичай подаються в ці моделі для їх навчання, і вони є основою длякомп'ютерного зору, обробки природної мови та інших нейронних мереж. (Рис.1.4)

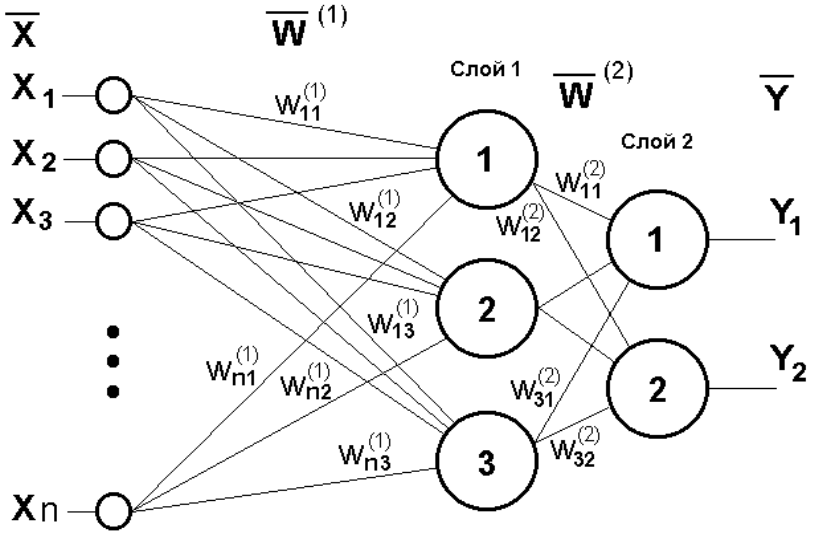


Рисунок 1.4 Багатошаровий персептрон

• Згорткові нейронні мережі подібні до мереж прямого поширення, але їх зазвичай використовують для розпізнавання зображень, розпізнавання образів та комп'ютерного зору. Ці мережі використовують принципи лінійної алгебри, зокрема множення матриць, для виявлення закономірностей у зображенні.[15]

• Рекурентні нейронні мережі ідентифікуються за їх зворотним зв'язком. Ці алгоритми навчання в першу чергу використовуються при використанні даних часових рядів для прогнозування майбутніх результатів, таких як прогнозування на фондовому ринку або прогнозування продажів.[15]

1.4 Дослідження застосування бібліотек та модулів, які використовуються в системах розпізнавання облич.

1.4.1 Бібліотека Tensorflow

Іншим не менш популярним та основним, так би мовити, конкурентом є модуль під назвою Tensorflow.

Створенна командою Google Brain, TensorFlow - це бібліотека з відкритим кодом для чисельних обчислень та масштабного машинного навчання. TensorFlow поєднує в собі безліч моделей машинного навчання та глибокого навчання (він же нейронних мереж) та алгоритмів, і робить їх дуже корисними. Вона використовує Python для забезпечення зручного інтерфейсного API для побудови програм із фреймворком, виконуючи ці програми у високопродуктивному C ++. TensorFlow може навчати та запускати глибокі нейронні мережі для рукописної класифікації цифр, розпізнавання зображень, вбудовування слів, рекурентні нейронні мереж, моделі послідовності в послідовності для машинного перекладу, обробки природної мови та багато іншого.[16]

TensorFlow дозволяє розробникам створювати графіки потоків даних - структури, що описують, як дані рухаються через графік або серію вузлів

обробки. Кожен вузол на графіку представляє математичну операцію, а кожне з'єднання або ребро між вузлами - це багатовимірний масив даних або тензор.

TensorFlow забезпечує все це за допомогою мови Python. Python простий у вивченні та роботі з ним, і він пропонує зручні способи висловити, як абстракції високого рівня можуть поєднуватися. Вузли та тензори в TensorFlow - це об’єкти Python, а програми TensorFlow - самі програми Python.

Однак фактичні математичні операції в Python не виконуються. Бібліотеки перетворень, доступні через TensorFlow, записані як високопродуктивні двійкові файли C ++. Python просто спрямовує трафік між фрагментами та забезпечує абстракції програмування високого рівня, щоб з'єднати їх між собою.

Додатки TensorFlow можна запускати на більшості зручних цілей: на локальній машині, кластері в хмарі, пристроях iOS та Android, центральних процесорах або графічних процесорах. Отримані в результаті моделі, створені TensorFlow, можуть бути розгорнуті на більшості будь-яких пристроїв, де вони будуть використовуватися для обслуговування прогнозів.[16]

TensorFlow 2.0, випущений у жовтні 2019 року, багато в чому переробив структуру, базуючись на відгуках користувачів, щоб полегшити роботу (наприклад, за допомогою порівняно простого API Keras для навчання моделей) та підвищив продуктивність. Розподілене навчання легше запускати завдяки новому API, а підтримка TensorFlow Lite дає можливість розгортати моделі на більшій кількості різноманітних платформ.

Плюси Tensorflow:

- Маштабованість. Можливість як зменшувати, так і збільшувати розмір моделі.
- Багатофункціональність.
- Має свій фреймворк для деплою у продакшині.
- Зрозуміла документація

Мінуси Tensorflow:

- Немає підтримки GPU, окрім NVIDIA.
- Досить складно дебажити, бо tensorflow має свою унікальну структуру.
- Швидкість роботи. З результатів тесту Benchmark tensorflow показав себе гірше аніж конкуренти

Існує ще багато різних модулів та бібліотек, але виділемо ще пару, такі як: Caffe2 та MxNet.

1.4.2 Модуль Caffe2

Caffe2 - довгоочікуваний наступник оригінальної Caffe, творець якої Янцин Цзя зараз працює у Facebook. Caffe2 - це другий фреймворк для глибокого навчання, який підтримується Facebook після Torch / PyTorch. Здається, основною відмінністю є твердження, що Caffe2 є більш масштабованим і легким. Це передбачає глибоке навчання для виробничих середовищ. Як і Caffe та PyTorch, Caffe2 пропонує API Python, що працює на движку C ++.[17]

1.4.3 Модуль MXNet

MXNet використовується для визначення, навчання та розгортання глибоких нейронних мереж. Він легкий, гнучкий та надмасштабний, тобто дозволяє швидко навчати моделі та підтримує гнучку модель програмування та різні мови.Успішна основа глибокого навчання - це та, яка відрізняється програмованістю, портативністю та масштабованістю.

MXNet підтримує декілька мов, таких як C ++, Python, R, Julia, Perl тощо. Це позбавляє від необхідності вивчення нової мови для використання фреймворку та спрощення мережевих визначень. Моделі MXNet є портативними таким чином, щоб вони вмістили дуже малий обсяг пам'яті. Отже, можна навчити свою модель у хмарі та розгорнути її на мобільному або підключеному пристрої. MXNet також може масштабуватися до декількох графічних процесорів та декількох машин.[18]

Завдяки різним перевагам перед іншими системами глибокого навчання, MXNet був обраний компанією Amazon для програм глибокого навчання своїх веб-служб.

1.4.4 Модуль Torch

Одним з найпоширеніших модулів, які використовуються у deep learning є модуль під назвою PyTorch. Версія Torch на Python, відома як Pytorch, була відкрита Facebook у січні 2017 року. PyTorch пропонує графіки динамічних обчислень, які дозволяють обробляти входи та виходи змінної довжини, що корисно, наприклад, при роботі з рекурентними нейронними мережами. З моменту появи PyTorch швидко став улюбленим серед дослідників машинного та глибинного навчання, оскільки дозволяє легко будувати певні складні архітектури. Інші фреймворки, що підтримують графіки динамічних обчислень, - це DyNet CMU та Chain PFN.

Torch - це обчислювальна структура з API, написаним на Lua, який підтримує алгоритми машинного навчання. Деякі його версія використовується великими технологічними компаніями, такими як Facebook та Twitter. Lua - це сценарій написання багатопарадигм, який був розроблений у Бразилії на початку 1990-х.

Було виділено основні позитивні та негативні риси цього модуля. **Почнемо з плюсів:**

- Багато модульних деталей, які легко поєднувати.
- Легко писати власні типи шарів і працювати на графічному процесорі.
- Багато попередньо навчених моделей.

Мінуси:

- Зазвичай доведеться писати власний тренувальний код.
- Відсутність комерційної підтримки.
- Не дуже вдала документація.

Проаналізувавши основні модулі, було обрано таку бібліотеку, як PyTorch. Хоча вона й має свої недоліки, наприклад, такі як швидкість роботи вони всі відходять на другий план в порівнянні зі зручністю, розумінням та багатофункціональністю. Виходячи з цього, було обрано таку мову програмування як Python.

1.5 Дослідження застосування баз даних для зберігання зображень для ідентифікації осіб

Існує чимала кількість баз даних, кожна з яких має свої особливості та, як плюси, так і недоліки. Проведемо короткий аналіз основних та найпопулярніших баз даних та оберемо ту, яка буде використовуватися в подальшому.

1.5.1 MySQL

MySQL - одна з найпопулярніших баз даних для веб-додатків. Фактично, є стандартом для веб-серверів, які працюють під управлінням операційної системи Linux. MySQL - це безкоштовний пакет програм, однак нові версії виходять постійно, розширюючи функціонал і покращуючи безпеку. Існують спеціальні платні версії, призначені для комерційного використання. У безкоштовної версії найбільший наголос робиться на швидкість і надійність, а не на повноту функціоналу, який може стати і перевагою і недоліком - в залежності від області застосування. [13]

Плюси:

- Розповсюджується безкоштовно.
- Гарна документація.
- Великий набір вбудованих функцій.
- MySQL - це кросплатформенний сервер баз даних. Він може працювати на різних платформах, таких як Linux, Solaris, Windows тощо. Це хороший вибір для тих проектів, які націлені на декілька платформ, зокрема веб-додатків.

Мінуси:

- MySQL не підтримує великий розмір бази даних.
- У порівнянні з іншими базами даних MySQL не має гарного інструменту розробки та налагодження.
- Платна підтримка.

1.5.2 Microsoft SQL Server

SQL Server є однією з найпопулярніших систем керування базами даних (СКБД) в світі. Вона підходить для всіх типів проектів: від невеликих вебзастосунків до великих високонавантажених проектів. СКБД була розроблена компанією Microsoft. Перша версія Microsoft SQL Server вийшла ще у 1987 році. А поточна версія вийшла ще у 2019 році та використовується по сьогоднішній день. Тривалий час sql сервер був винятково в користуванні для Windows, проте починаючи з версії 16 ця система доступна і на Linux.

Плюси:

- Швидкість. Microsoft SQL Server працює дуже швидко.
- Надійність і безпека. Має власне шифрування даних.
- Простота. Дуже просто в користуванні та має велику кількість застосунків для більш зручного адміністрування.

Мінуси:

- **Вартість.** Одним з основних недоліків використання Microsoft SQL Server замість альтернативної системи управління реляційними базами даних є те, що вона досить дорога. Незважаючи на те, що використання програмного забезпечення для розробок чи освітніх цілей є безкоштовним, за будь-яке комерційне використання вимагається ліцензійний збір. Наприклад, для SQL Server 2019 SQL Server Standard Edition коштує 7171 долар за процесор. Для малого бізнесу та приватних осіб, які працюють на комерційних веб-сайтах, це недоступно. Конкуруюче програмне забезпечення, таке як MySQL, часто є безкоштовним для використання. Однак у випадках, коли це не так,

найдорожчий пакет MySQL Enterprise коштує $ 4999 на сервер щороку. Це значно дешевше, ніж навіть стандартний пакет Microsoft SQL.

• **Несумісність з різними ОС.** Microsoft SQL Server призначений лише для роботи на серверах під керуванням Windows. З різних причин, включаючи витрати на ліцензування та проблеми безпеки, розробники можуть вибрати розміщення своїх веб-сайтів на машинах, що базуються на Unix. У цьому випадку вони не зможуть використовувати SQL Server. Конкуренти часто можуть працювати на інших платформах. На відміну від Microsoft SQL Server, MySQL підтримується на всіх основних платформах, включаючи Windows, Linux, Mac OSX та інші варіанти Unix.

1.5.3 PostgreSQL

PostgreSQL - це система управління базами даних з відкритим кодом корпоративного класу. Він підтримує як SQL, так і JSON для реляційних та нереляційних запитів щодо розширюваності та відповідності SQL. PostgreSQL підтримує розширені типи даних та функції оптимізації продуктивності, які доступні лише у дорогих комерційних базах даних, таких як Oracle та SQL Server. Він також відомий як Postgres.

Плюси:

• PostgreSQL може запускати динамічні веб-сайти та веб-програми як опцію стека LAMP

• Ісходний код PostgreSQL знаходиться у вільному доступі за ліцензією з відкритим кодом. Це дає свободу використовувати, модифікувати та впроваджувати це відповідно до бізнес-потреб.

• Низькі витрати на обслуговування та адміністрування як для особистого, так і для корпоративного використання PostgreSQL

Мінуси:

• Зміни, зроблені для покращення швидкості, вимагають більше роботи, ніж MySQL, оскільки PostgreSQL зосереджується на сумісності

• Багато програм з відкритим кодом підтримують MySQL, але можуть не підтримувати PostgreSQL

• Що стосується показників продуктивності, то postgres повільніше, ніж MySQL.

Проаналізувавши основні бази даних, було прийнято рішення використовувати базу даних PostgreSQL, тому що вона повністю задовольняє потребам, є досить швидкою, а саме головне, що вона є майже безкоштовною. Для розміщення бази даних було обрано Microsoft Azure, бо вони мають гарну сумісність та все доволі легко та швидко налаштовується

1.6 Обгрунтування обрання мови програмування Python

Переваги використання мови:

• Гнучкість – це основна перевага мови, так як завдяки своїй гнучкості мова отримала популярність серед багатьох розробників. Динамічна типізація. У python не треба заздалегідь оголошувати тип змінної, що дуже зручно при розробці.

• Простота синтаксису. Синтаксис - це саме те, через що розробники використовують саме Python, з синтаксису було прибрано все зайве, код чистий і зрозумілий без зайвих дужок і виразів.

• Інтерпретованість. Інтерпретатор Python існує для всіх популярних платформ і за замовчуванням входить в більшість дистрибутивів Linux, а значить є на більшості серверів «з коробки».[20]

• Широке застосування. Використовується для розробки веб-додатків, ігор, зручний для автоматизації, математичних обчислень, машинного навчання, в області інтернету речей. Існує реалізація під назвою Micro Python, оптимізована для запуску на мікроконтролерах (можна писати інструкції, логіку взаємодії пристроїв, організовувати зв'язок, реалізовувати розумний будинок).[20]

• Інтеграція з C / C ++, якщо можливостей python недостатньо.

• PEP - єдиний стандарт для написання коду, що робить код підтримуваним і читабельним навіть при переході від одного програміста до іншого. Це підтримує популярність Python.[20]

• Open Source - код інтерпретатора Python є відкритим, що дозволяє будь-кому, хто зацікавлений у розвитку мови взяти участь в його розробці і поліпшити його. Якщо дивитися деталі релізу однією з версій мови, то можна помітити, що величезні частини нового функціоналу реалізовані сторонніми розробниками.[20]

• Ком'юніті - навколо Python утворилося досить дружнє і приємне ком'юніті, яке готове прийти на допомогу будь-якому починаючому або вже вмілому розробнику і розібратися в його проблемі.[20]

Всі ці переваги мови зробили його популярним і затребуваним на даний момент, дозволивши Python розвиватися величезними темпами. Існує вже третя версія мови, яка є основною сьогодні. Друга версія мови перестала підтримуватися в грудні 2019 го року.[20]

Але незважаючи на всі явні переваги, Python також має ряд недоліків, які, на мою думку, з лишком перекриваються його перевагами:

• **Продуктивність.** Більшість розробників, та й сам творець мови, сходяться на думці, що Python не так шустрий, наскільки хотілося б. Це обумовлено тим, що Python інтерпретована мова. Але навіть у порівнянні з іншими інтерпретованими мовами помітно, що Python програє в продуктивності. Але це легко можна нівелювати за допомогою C реалізацій тій чи іншої проблемної ділянки коду. В умовах сьогоднішніх потужностей - це не дуже помітно.[20]

• **Динамічна типізація** - завдяки динамічної типізації Python споживає більше ресурсів, ніж міг би, але це часто компенсується внутрішнім кешування.[20]

Виходячи з перерахованого вище є зрозумілим, що всі недоліки мови з лишком нівелюються його перевагами, які набагато вагоміші в сьогоднішніх

реаліях. І не варто забувати, що мова розвивається. Це дає надію на те, що всі недоліки будуть або скорочені, або усунуті зовсім.

1.6 Постановка задачі

Тема: Розробка программного модуля інтелектуальної системи ідентифікації осіб без засобів індивідуального захисту на основі методів глибинного навчання.

Метою даної роботи є створення системи, яка буде з максимальною точністю визначати чи є маска на обличчі та у разі відсутності вирізати лице порушника та автоматино відправляти його до відділу охорони в будь-який зручий месседжер або на email. Для цього необхіодно вирішити задачі:

- Дослідити основні методи глибинного навчання та обрати оптимальний для створення моделі.
- Розробити датасет для тренування, тестування та валідації нейронної мережі.
- Розробити програмну реалізацію продукту, використовуючи мову програмування Python.
- Провести тестування та зробити аналіз реалізованого програмного продукту.

Основними функціональними вимогами до інтелектуальної системи ідентифікації осіб є:

- Можливість ідентифікувати особу по фото або відео.
- Можливість підключатися до системи RTSP за домогою live стріму, щоб система працювала та надавала результати в реальному часі.

• У разі визначення людини без маски автоматично відправляти його фото до відділу охорони.

• Можливість взаємодіяти з любим зручним меседжером. На даному етапі було обрано Microsoft Teams.

• Ідентифікувати порушника за допомогою Azure Face Api.

• Система повинна мати точність розпізнавання більше ніж 90%.

• Система повинна без помилок ідентифікувати порушника, уникаючи False Positive та False Negative:

«False Positive» - це коли система розпізнавання осіб зіставляє обличчя людини із зображенням в базі даних, але насправді цей збіг невірний. Це коли охорона ідентифікує особу як «Іван Петров», але система помилково повідомляє охоронцю, що це «Петро Іванов».

«False Negative» - це коли система розпізнавання осіб не може зіставити обличчя людини із зображенням, яке фактично міститься в базі даних. Іншими словами, система помилково поверне нульовий результат у відповідь на запит.

• Система повинна зберігати в базі даних ФІО кожного співробітника та його акуальну фотографію.

• Система не повинна відсилати фото одного й того ж порушника порушника декілька раз за 1 хвилину.

• Можливість конфігурування частоту обробки фреймів.

2. Опис методів, які використовуються в дослідженні

2.1 Бінарна класифікація

Бінарна класифікація - це завдання класифікації елементів набору на дві групи на основі правила класифікації[19]. У багатьох практичних завданнях бінарної класифікації ці дві групи не є симетричними, і замість загальної точності становить інтерес відносна частка різних типів помилок. Наприклад, в нашому випадку виявлення порушника, коли його немає (хибнопозитивний результат), розглядається інакше, ніж невиявлення порушника, коли він присутній (помилково негативні результати).

Статистична класифікація - це проблема, яка вивчається в машинному навчанні. Це тип навчання з учителем, метод машинного навчання, в якому категорії зумовлені, і використовується для категоризації нових імовірнісних спостережень за зазначеним категоріям. Коли є тільки дві категорії проблема відома як статистична бінарна класифікації.

Деякі з методів, які зазвичай використовуються для бінарної класифікації:

- Дерева рішень;
- Випадкові ліси;
- Баєсові мережі;
- Метод опорних векторів;
- Нейронні мережі;
- Логістична регресія.

Задача визначення людини без засобів індивідуального захисту - є задачею бінарної класифікації, тобто є два відповідних класа: засоби індивідуального захисту присутні - (1); засобів індивідуального захисту немає - (0). В даній роботі для вирішення проблеми бінарної класифікації буде використовуватися згорткова нейронна мережа, про яку піде пізніше.

2.1.1. Оцінка якості роботи класіфікатора

Якщо дана класифікація багатьох даних, існує чотири базові комбінації дійсної категорії та визначеної категорії[19]:

• правильно визначені позитивні класифікації - True Positive (TP);

• правильно визначені негативні класифікації - True Negative (TN);

• помилково визначені позитивні класифікації - False Positive (FP);

• помилково визначені негативні класифікації - False Negative (FN).

Нижче наведено таблицю для більш детального розуміння. По вертикалі вказані актуальні значення, по горизонталі - значення, які надала система.

	Людина в масці (Відповідь системи)	**Людина без маски (Відповідь системи)**
Людина в масці	True Positive	False Negative
Людина без маски	False Positive	True Negative

Таблиця 2.1. Базові комбінації варіантів класифікації

Скажімо, ми перевіряємо деяких людей на наявність масок. Деякі з цих людей одягають захисні маски, і система правильно говорить, що вони позитивні. Їх називають справжніми позитивними даними (TP). Деякі одягають маски, але система неправильно стверджує, що вони без захисних масок. Їх називають помилково визначений негатив (FN). У деяких немає маски і система показує, що у них немає - правильно визначений негатив (TN). Нарешті, можуть люди без масок, яких система віднесла до людей в масках - помилково визначений позитив (FP). Така матриця (2.1), що наведена зверху називається матрицею помилок або confusion matrix

Існує багата кількість методів перевірки точності роботи класифікатора. Нижче наведені основні з них:

• Accuracy

Інтуїтивно зрозумілою, очевидною, але майже невикористовуваною метрикою є accuracy - частка правильних відповідей алгоритму:

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (2.1)$$

Цю метрику намагаються не використовувати в задачах з нерівними класами, і це легко показати на прикладі.

Припустимо, ми хочемо оцінити роботу класифікатора. У нас є 10 людей без масок, 5 з яких наш класифікатор визначив вірно (True Negative = 5, False Positive = 5), і 100 в масках 90 з яких класифікатор також визначив вірно (True Positive = 90, False Negative = 10). Тоді accuracy:

$$accuracy = \frac{90+5}{90+5+5+10} = 86.4$$

Однак якщо ми просто будемо передбачати всіх людей в масках то отримаємо більш високу accuracy:

$$accuracy = \frac{100+0}{100+0+10+0} = 90.9$$

При цьому, модель абсолютно не володіє ніякою передбачуваною силою, так як спочатку стояла задача визначати людей без масок. Подолати це допоможе перехід із загальною для всіх класів метрики до окремимх показників якості класів.

• Precision та recall

Для оцінки якості роботи алгоритму на кожному з класів окремо використовуються метрики precision (точність) і recall (повнота).

$$precision = \frac{TP}{TP+FP} \quad (2.2)$$

$$recall = \frac{TP}{TP+FN} \quad (2.3)$$

Precision можна інтерпретувати як частку об'єктів, названих класифікатором позитивними і при цьому дійсно є позитивними, а recall показує, яку частку об'єктів позитивного класу з усіх об'єктів позитивного класу знайшов алгоритм.[22]

Саме введення precision не дозволяє нам записувати всі об'єкти в один клас, так як в цьому випадку ми отримуємо зростання рівня False Positive. Recall демонструє здатність алгоритму виявляти даний клас взагалі, а precision - здатність відрізняти цей клас від інших класів. Precision і recall не залежить, на відміну від accuracy, від співвідношення класів і тому застосовні в умовах незбалансованих вибірок.

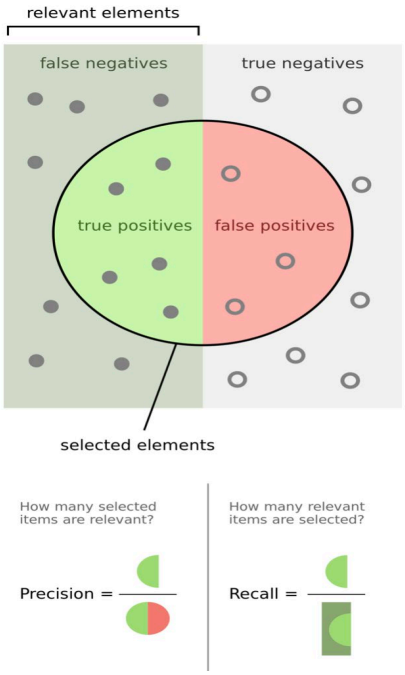


Рисунок 2.1 Precision та recall

• F-міра

Зрозуміло що чим вище precision і recall, тим краще. Але в реальному житті максимальна точність і повнота недосяжні одночасно і доводиться шукати якийсь баланс. Тому, хотілося б мати якусь метрику яка об'єднувала б у собі інформацію про precision та recal алгоритму. Першим, що спадає на думку, є використання простого арифметичного середнього, але, як показує практика, це зовсім неефективно. Тому використовують гармонічне середнє або F-міру.

$$F = 2\frac{Precision*Recall}{Precision+Recall} \tag{2.4}$$

Дана формула надає однакову вагу precision і recall, тому F-міра буде падати однаково при зменшенні і precision і recall.

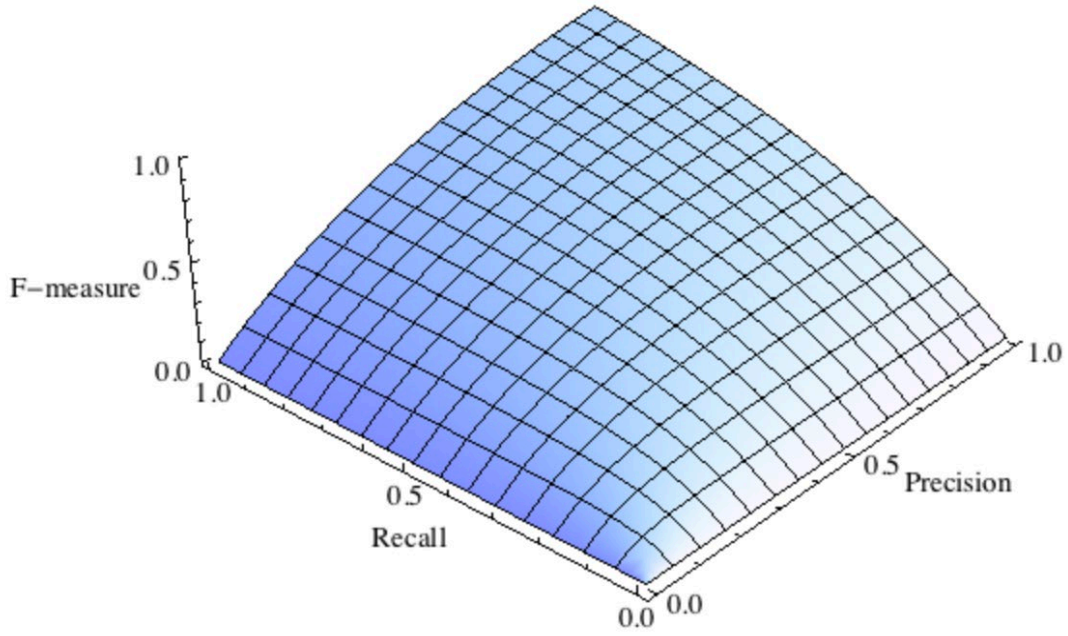


Рис 2.2 Сбалансована F-міра

2.2 Згорткова нейронна мережа (ЗНМ)

Згорткова нейронна мережа - це алгоритм, який може приймати зображення як вхідний сигнал, призначати важливість (у формі тренувальних ваг та упереджень) аспектам чи особливостям зображення та виводити рішення чи іншу форму логіки на основі того, що воно побачило на зображенні. Попередня обробка та підготовка даних дуже мінімальна, коли мова йде про архітектури ЗНМ, оскільки особливостями даних є фактичні значення пікселів, і при достатній кількості навчальних даних ЗНМ автоматично визначить, які з цих пікселів є найбільш важливими для прийняття рішень.

Будь-яке зображення, що зберігається в електронному вигляді, є лише матрицею значень пікселів, що варіюються від 0 (чорний) до 255 (білий), при цьому значення між цим діапазоном вказують на “сірість” пікселя - чим ближче значення до 0, тим темніше сірий (і навпаки). Можна було би просто представити матрицю як єдиний вектор пікселів і подати її безпосередньо в багатошаровий

перспетрон, де кожен піксель відповідає віходному нейрону, але все трохи складніше.

У випадку стандартної нейронної мережі прямого поширення, кожен вхідний нейрон безпосередньо відображатиметься до об'єкта в наборі даних, і припущення тут полягає в тому, що кожен нейрон є повністю незалежним один від одного. Однак це не стосується даних зображень. Пікселі на зображенні мають як просторову, так і часову залежності. Наприклад, якщо уявити зображення місяця на нічному небі, очікується, що всі пікселі поблизу місяця мають однакові значення пікселів (всі вони повинні бути приблизно 200–255), і чим далі від місяця піксель, тим темніше стає піксель (і чим ближче, тим його значення зменшується до 0). Стандартна нейронна мережа прямого поширення не зможе зберегти цей тип просторової та тимчасової інформації, і її продуктивність обмежена інформацією, яку вона може отримати від кожного окремого пікселя на зображенні, не враховуючи інших пікселів поблизу.

Згорткова нейронна мережа, навпаки, може фіксувати ці залежності за допомогою застосування відповідних фільтрів. Архітектура краще підходить до набору даних зображення завдяки зменшенню кількості задіяних параметрів та багаторазовому використанню ваг. Іншими словами, мережу можна навчити краще розуміти витонченість зображення.

На рисунку маємо зображення RGB, яке розділене на три кольорові площини - червону, зелену та синю. Розміри цього базового зображення становлять 4 x 4 x 3 - чотири пікселі у висоту та ширину та три кольорові канали.

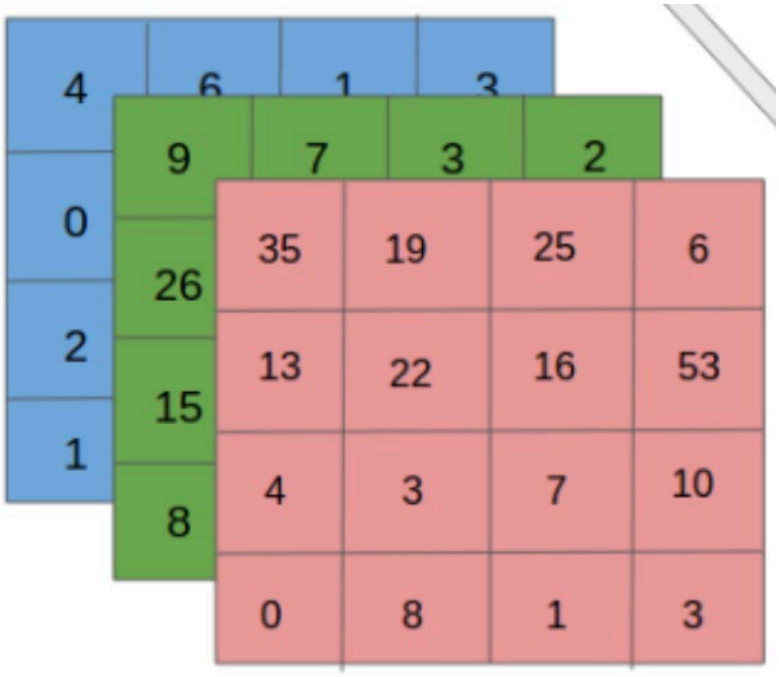


Рисунок 2.3 Матриця зображення

Наведене зображення 4 x 4 має 16 пікселів на кожен кольоровий канал. Для RGB-зображення це в цілому дорівнює 16 x 3 (кольорові канали) = 48 пікселів. Можна уявити, наскільки обчислювально це стало б, як тільки зображення досягнуть набагато вищих розмірів. Наприклад, зображення розміром 8K із розмірами (7680 x 4320 x 3) буде дорівнювати загальній кількості 99 532 800 пікселів. Однією з головних ролей CNN є зменшення цих зображень до форми, яку легше обробити, не втрачаючи при цьому особливостей, що значною мірою сприяють її прогнозуванню. Це першорядне, коли розробляється архітектура, яка не тільки добре виводить особливості зображення, але й може масштабуватися до масивних наборів даних зображень (десятки тисяч зображень і вище).

Згортковий слой нейронної мережі являє себою застосування операції згортки до виходів з попереднього шару, де ваги ядра згортки є параметрами,

що навчаються[22]. Ще одна вага використовується в якості константного зсуву (англ. Bias). При цьому є кілька важливих деталей:

• В одному згротковому шарі може бути декілька згорток. В такому разі для кожної згортки на виході вийде окреме зображення. Наприклад, якщо вход мав розмірність $w \times h$, а в його шарі було n згорток з ядром розмірності $k_x \times k_y$, то на виході отримуємо розмірність:

$$n \times (w - k_x + 1) \times (h - k_y + 1)$$

• Ядра згортки можуть бути тривимірними. Згортка тривимірного входу з тривимірним ядром відбувається аналогічно, просто скалярний добуток рахується ще й по всіх шарах зображення. Наприклад, для усереднення інформації про кольори початкового зображення, на першому шарі можна використовувати згортку розмірності $3 \times w \times h$. На виході такого шару буде вже одне зображення (замість трьох);

• Можна помітити, що застосування операції згортки зменшує зображення. Також пікселі, які знаходяться на межі зображення беруть участь в меншій кількості згорток, ніж внутрішні. У зв'язку з цим в згортальних шарах використовується доповнення зображення. Виходи з попереднього шару доповнюються пікселями так, щоб після згортки зберігся розмір зображення. Такі згортки називають однаковими, а згортки без доповнення зображення називаються правильними (англ. Valid convolution).

• Ще одним параметром згорткового шару є зсув. Хоч зазвичай згортка застосовується поспіль для кожного пікселя, іноді використовується зсув, відмінний від одиниці - скалярний добуток рахується не з усіма можливими положеннями ядра, а тільки до положень, кратними деякого зсуву S. Тоді, якщо вхід мав розмірність $w \times h$, а ядро згортки мало розмірність $k_x \times k_y$ та використовувася зсув S, то вихід буде мати таку розмірність:

$$\left(\frac{w - k_x}{S} + 1\right) \times \left(\frac{w - k_y}{S} + 1\right)$$

Пулінговий шар призваний знижувати розмірність зображення. Початкове зображення ділиться на блоки розміром $w \times h$ і для кожного блоку обчислюється деяка функція. Найчастіше використовується функція максимуму (max pooling) або (зваженого) середнього[25]. Параметрів, що начаються, у цього шару немає. Основні цілі пулінгова шару:

- зменшення зображення, щоб наступні пакунки оперували над більшою областю початкового зображення;
- збільшення інваріантності виходу мережі по відношенню до малого переносу входу;
- прискорення обчислень.

Проходження відповідних шарів, а саме згорткового та пулінгового має дати змогу моделі успішно зрозуміти найважливіші особливості зображення. На цьому етапі можемо згладити дані та передати їх через нейронну мережу прямого поширення, що дозволить виконати класифікацію за вихідними категоріями. Загальна структура згорткової мережі виглядає наступним чином:

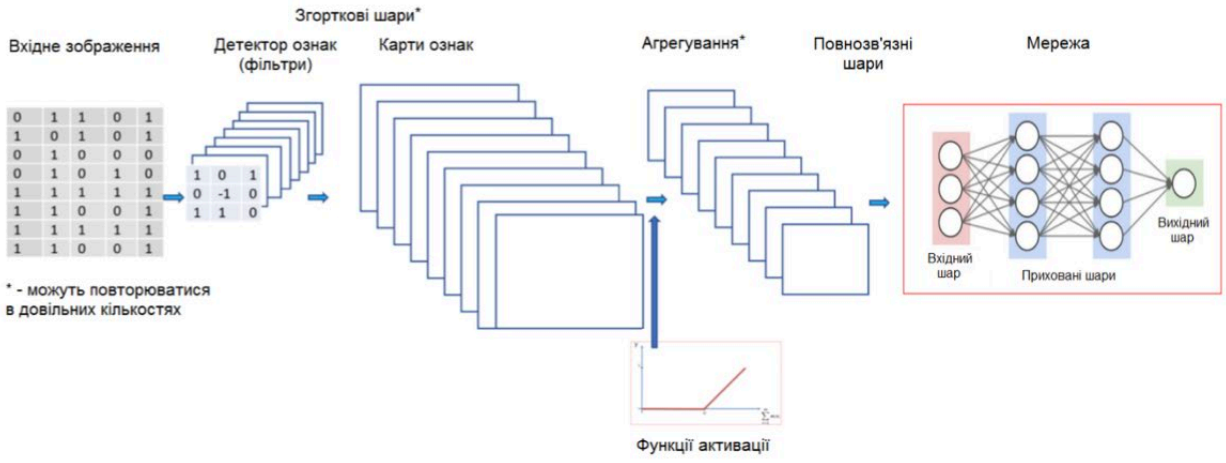


Рисунок 2.4 Схема проектування згорткової нейронної мережі

2.2.1 Варіанти згорткових нейронних мереж

Як зазначалося раніше в роботі, для класифікації осіб в масках та без буде використовуватися згорткова нейронна мережа, але існує декілька різновидів згорткової нейронної мережі, на яких треба зупинитися більш детально.

2.2.1.1 R-CNN

Архітектура R-CNN (Region-Based Convolutional Neural Network) була розроблена в 2014 році Ross Girshik і іншими [30]. В основі цього методу лежить наступний алгоритм:

- Знаходження потенційних об'єктів на зображенні і розбиття їх на регіони за допомогою методу selective search.
- Витяг ознак кожного отриманого регіону за допомогою згорткових нейронних мереж.
- Класифікація оброблених ознак за допомогою методу опорних векторів (SVM, Support Vector Machine) і уточнення меж регіонів за допомогою лінійної регресії.

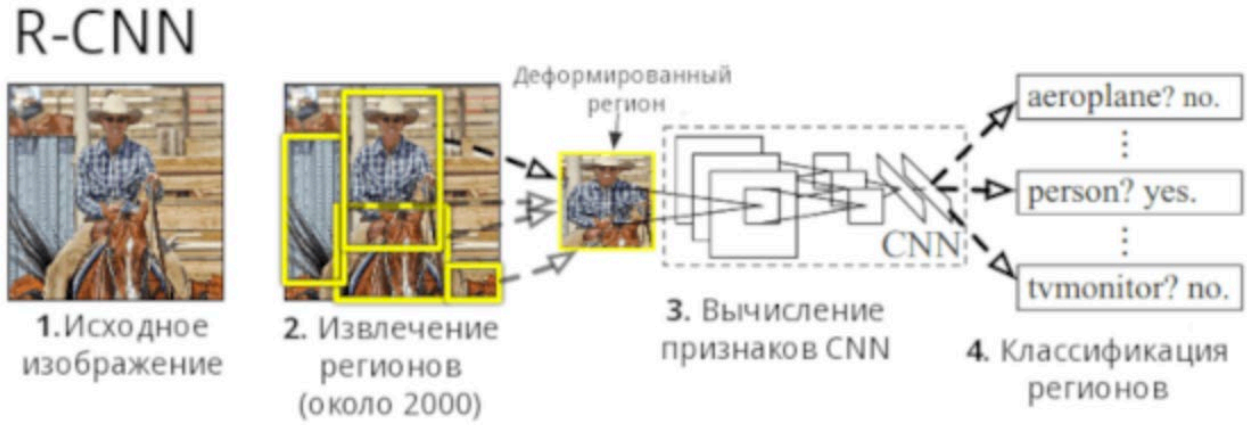


Рисунок 2.5 Архітектура R-CNN

У підсумку, отримуємо окремі регіони з об'єктами і їх класами. У центрі стоять згорткові нейронні мережі, які показують високу точність на прикладі зображень. Але у такої архітектури є недоліки:

- Енерговитрата - вимагає великої кількості часу на навчання.

- У методі selective search зображення спочатку сегментируется на 2000 регіонів, які потім в ході ітерування за допомогою жадібного алгоритму об'єднуються в більш великі регіони.

- Крім того, самі згорткові мережі теж вимагають обчислювальних потужностей.

- Не може бути використана для відео. Знову ж це випливає з нестачі вище, так як всі проміжні методи енерговитратні, тому кадри просто не встигатимуть оброблятися.

- Selective search не є алгоритмом машинного навчання, тому можуть виникнути проблеми з виявленням потенційних об'єктів на різних зображеннях.

На даний момент модель R-CNN застаріла і не застосовується.

2.2.1.2 Fast R-CNN

Недоліки R-CNN привели авторів у 2015 році до поліпшення моделі. Вони назвали її Fast R-CNN [31]. В її основі лежить наступна архітектура:

- Зображення подається на вхід згорткової нейронної мережі і обробляється за допомогою selective search. У підсумку, маємо карту ознак і регіони потенційних об'єктів.

- Координати регіонів потенційних об'єктів перетворюються в координати на карті ознак.

- Отримана карта ознак з регіонами передається шару RoI (Region of Interest) polling layer.

- Тут на кожен регіон накладається сітка розміром H×W.

- Потім застосовується MaxPolling для зменшення розмірності. Так, усі регіони потенційних об'єктів мають однакову фіксовану розмірність.

- Отримані ознаки подаються на вхід повнозв'язного шару (Fully-conectedlayer), який передається двом іншим повнозв'язним шарам. Перший з функцією активацією softmax визначає ймовірність приналежності класу, другий - кордону (зміщення) регіону потенційного об'єкта (bounding box).

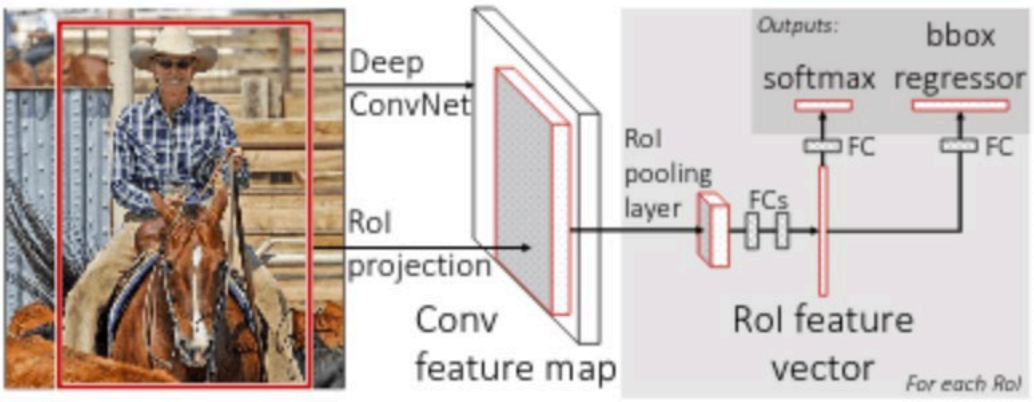


Рисунок 2.6 Архітектура Fast R-CNN

Fast R-CNN показує трохи більш високу точність і великий приріст часу обробки на відміну від R-CNN, так як не потрібно подавати всі регіони на згорткові шари. Але тим не менш, цей метод використовує витратний Selective Search. Тому автори прийшли до Faster R-CNN.

2.2.1.3 Faster R-CNN

Автори продовжили поліпшення над Fast R-CNN і в 2016 запропонували Faster R-CNN. Вони розробили власний метод локалізації об'єкта замість Selective Search - RPN (Region Proporsal Networks) [32]. В основі RPN лежить система якорів. Архітектура Faster R-CNN утворена в такий спосіб:

- Зображення подається на вхід згорткової нейронної мережі.
- Так, формується карта ознак. Карта ознак обробляється шаром RPN.
- Тут ковзне вікно проходиться по карті ознак. Центр ковзного вікна пов'язаний з центром якорів. Якоря - це області, які мають різні співвідношення сторін і різні розміри. Автори використовують 3 співвідношення сторін і 3 розміру.
- На основі метрики IoF (intersection-over-union), ступеня перетину якорів і справжніх розмічених прямокутників, виноситься рішення про поточний регіоні - є об'єкт чи ні.
- Далі використовується алгоритм FastCNN: карта ознак з отриманими об'єктами передаються шару RoI з подальшою обробкою повнозв'язних

шарів і класифікацією, а також з визначенням зсуву регіонів потенційних об'єктів.

2.2.1.4 Single Shot MultiBox Detector

SSD призначений для виявлення об'єктів у режимі реального часу. Faster R-CNN використовує RPN для створення bounding боксів і використовує ці поля для класифікації об'єктів. Хоча це вважається початком мистецтва в точності, весь процес працює зі швидкістю 7 кадрів в секунду. Значно нижче того, що потрібно для обробки в режимі реального часу.

SSD прискорює процес, позбавляючи потреби в RPN. Щоб відновити падіння точності, SSD застосовує кілька вдосконалень, включаючи багато масштабні функції та поля за замовчуванням. Ці вдосконалення дозволяють SSD відповідати точності Faster R-CNN, використовуючи зображення з низькою якістю, що ще більше підвищує швидкість. Згідно з наступним порівнянням, він досягає швидкості обробки в реальному часі і навіть перевершує точність Faster R-CNN. (Точність вимірюється як середня середня точність mAP: точність прогнозів.)

System	VOC2007 test *mAP*	FPS (Titan X)	Number of Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	~6000	~1000 x 600
YOLO (customized)	63.4	45	98	448 x 448
SSD300* (VGG16)	77.2	46	8732	300 x 300
SSD512* (VGG16)	**79.8**	19	24564	512 x 512

Рисунок 2.7 Порівняння якості моделей

Ось так виглядає подібна архітектура нейронної мережи, яка буде використовуватися:

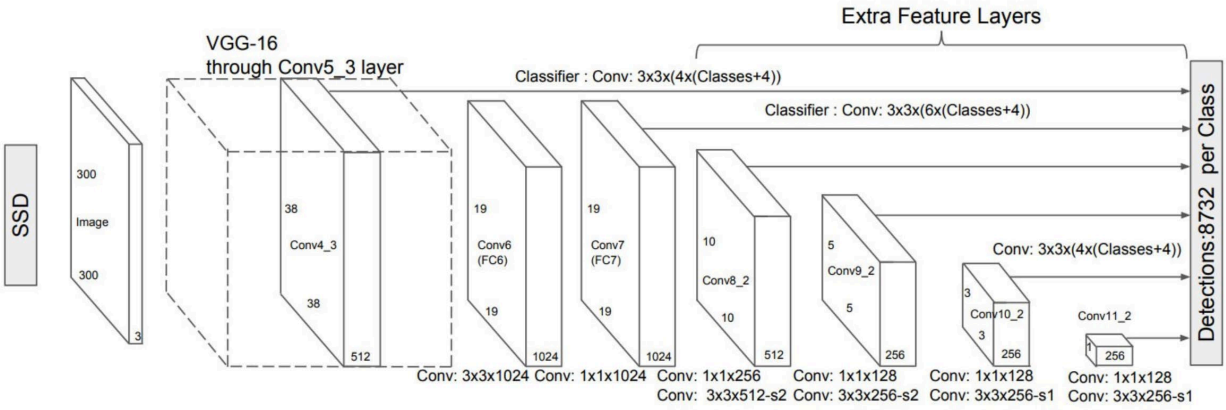


Рисунок 2.8 Архітектра подібної нейронної мережі

3. Розробка програмного модуля інтелектуальної системі ідентифікації осіб без засобів індивідуального захисту

3.1 Розробка нейронної мережі

Як було зазначено раніше у минулому розділі, буде використовуватися така структура моделі як SSD. Створена модель складається з 8 основних згорткових шарів, які ще називають “network backbone”; має 5 шарів по локалізації (корекції меж) та класифікації та загалом має 28 шарів. На вході модель отримує зображення розміром 260 × 260. Кількість каналів у кожному згортковому шарі становить 32, 64 і 128. Загальна кількість параметрів для всієї цієї моделі становить лише 1,015 мільйонів параметрів. Маємо таку структуру нейронної мережі (рис.3.1):

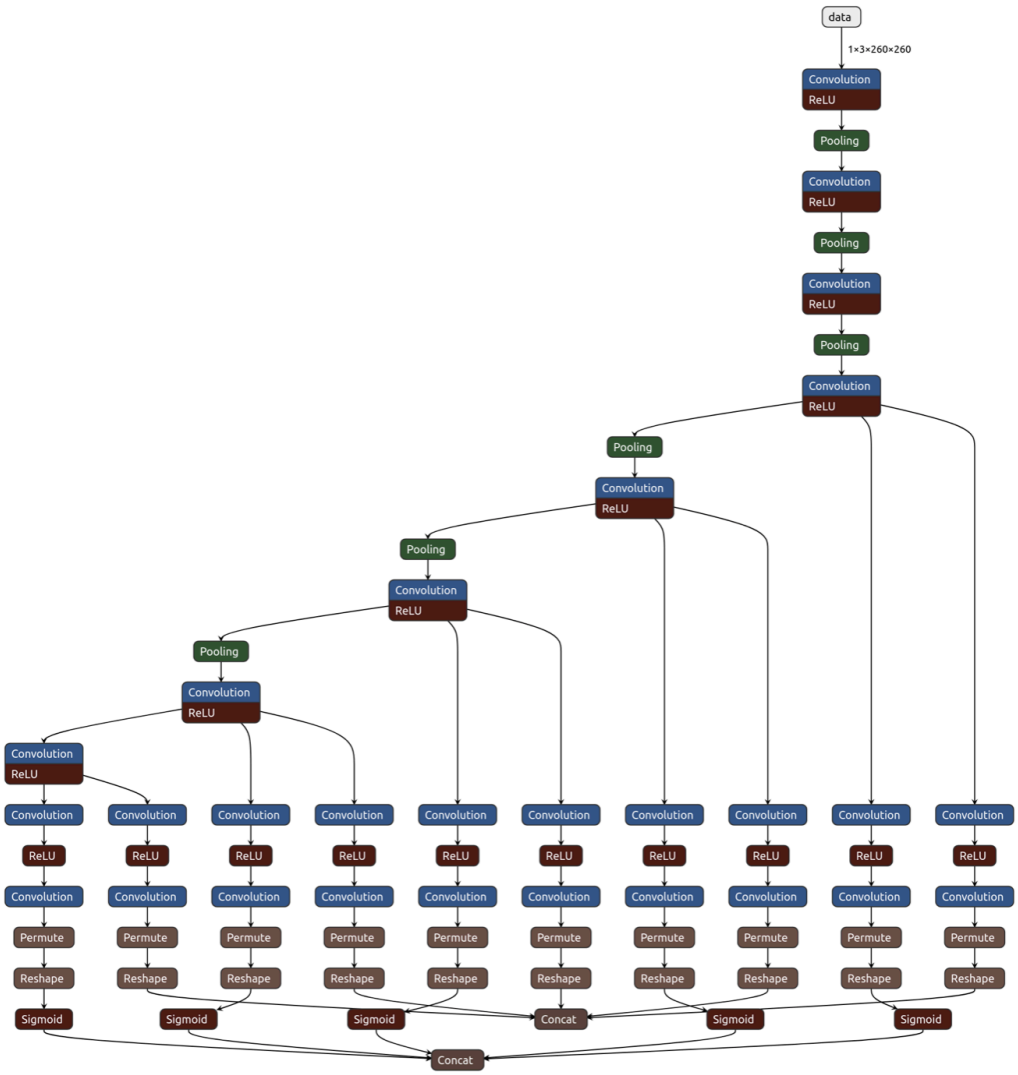


Рисунок 3.1 Структура нейронної мережі

3.2 Навчання нейронної мережі

Для навчання нейронної мережі були використані такі датасети як Wider Face та MAFA. Wider face використовується для виявлення обличчя та містить в собі 32203 зображення, в яких міститься близько 400 тисяч промаркірованих різних обличь з високим ступенем варіативності в масштабі, позі та оклюзії, як показано на прикладах зображень.

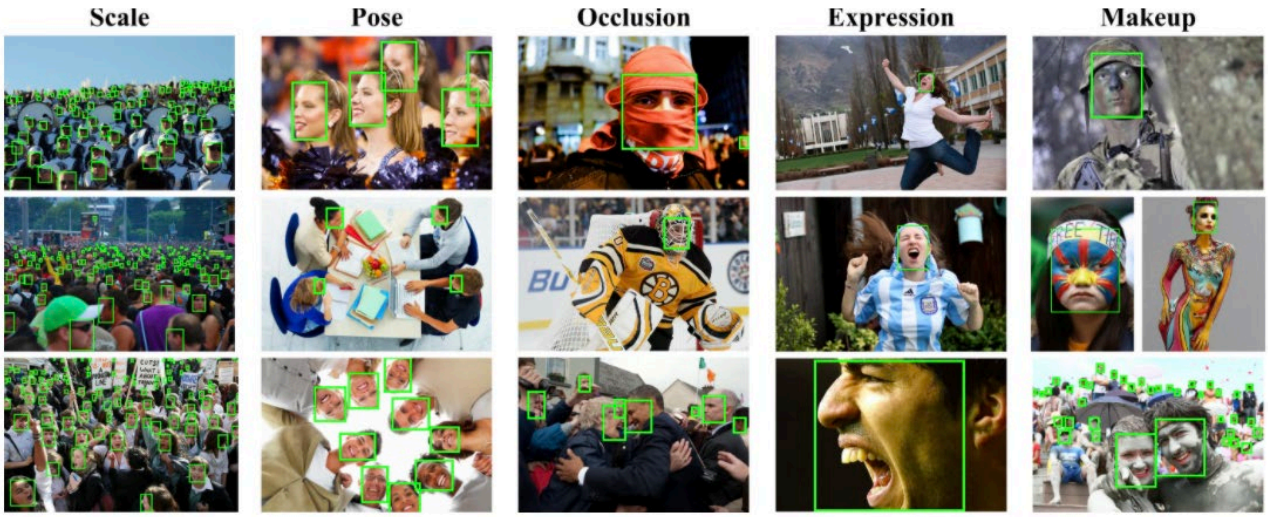


Рисунок 3.2 Приклади зображень Wider Face

З Wider Face було обрано 3894 різних зображень. Цей датасет використовується для класифікації обличь без масок.

Для класифікації обличь в масках було обрано датасет MAFA. Цей датасет є відкритим та був створений за допомогою Ge Shiming з Інституту інформаційних технологій Китайської академії наук. Загалом Mafa – це датасет з закритими обличчями, тобто обличчя можуть бути закриті як і руками, так і якимись сторонніми предметами, такими як маска. Більшість зображень з цього датасету як раз становлять обличчя, закриті масками. Було обрано 4064 таких зображень.

Рисунок 3.3 Приклад зображень MAFA

Далі було випадково поділено набори даних на набір для тренування та на набір для валідації. Нижче приведена статистика по кожному з наборів даних.

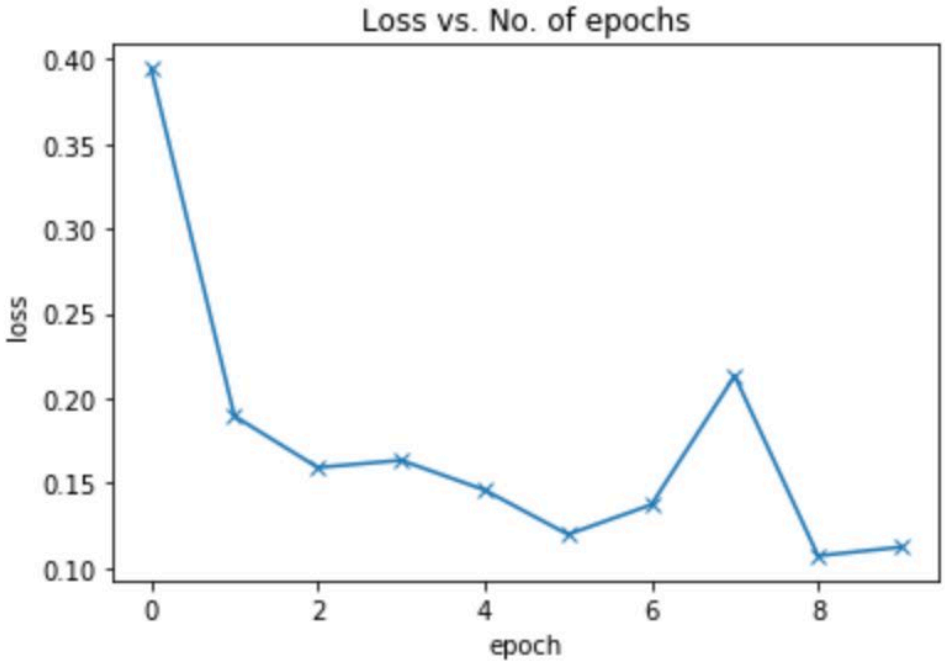
	Wider Face	MAFA	Загалом
Тренування	3114	3006	6120
Валідація	780	1059	1839

Таблиця 3.1 Статистика по зображенням

Після того, як датасет був розподілений на вибірки, починаємо навчання моделі. Для навчання моделі було обрано віртуальну машину з GPU на сервісі Azure для того, щоб забезпечити більшу швидкість навчання . Модель досягає точності близько 96,02%, і, дивлячись на графік, здається малоймовірним, що модель досягне точності вище 99% навіть після тривалого навчання. Це свідчить про те, що, можливо, в подальшому доведеться використовувати більш потужну модель, щоб точніше класифікувати зображення. Це можна зробити, додавши до моделі більше згорткових шарів, або збільшивши кількість каналів у кожному згортковому шарі або за допомогою методів регуляризації.

Модель змогла досягти точності 96,02%, що є гарним показником, оскільки для досягнення цього рівня знадобилося лише 8 епох.

```
[59]: plot_losses(history)
```




```
[60]: plot_accuracies(history)
```

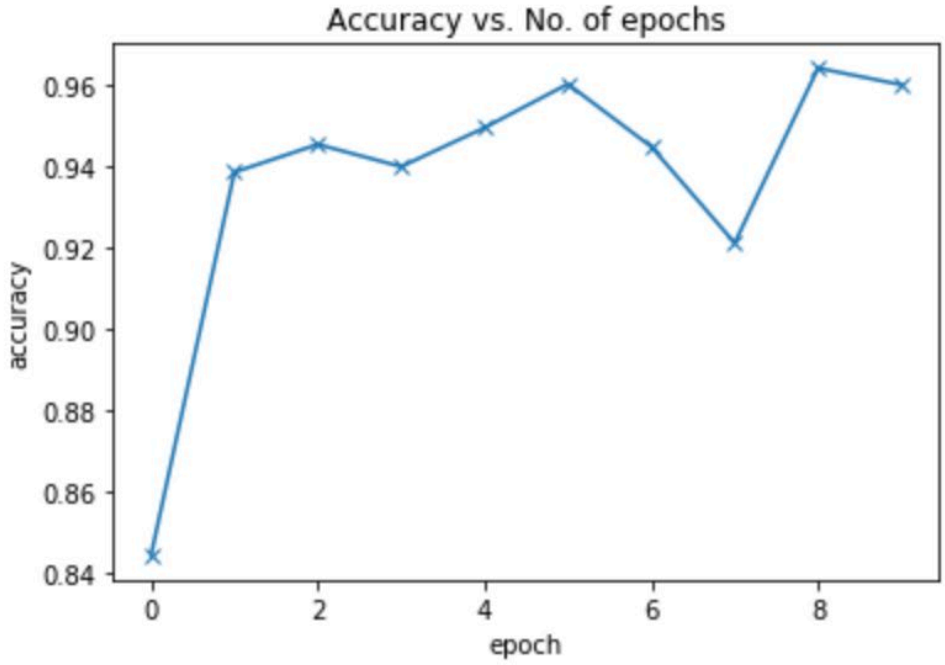


Рисунок 3.4 Точність моделі

3.3 Розробка бази даних

Для швидкої ідентифікації осіб без медичних масок було розроблено базу даних, в якій будуть зберігатися фото людей, яких треба ідентифікувати. Для

цього було створено таку базу даних як PostgreSQL. Базу було розроблено за допомогою Microsoft Azure, а підключення до бази відбувається через термінал та psql – це термінальний клієнт для роботи з PostgreSQL.

Було розроблено таблицю “Users” в базі даних, яка містить такі поля як id, Name та FaceImage. Поле id має такі властивості: тип даних Int, Not null, primary key; поле Name має тип даних VARCHAR; поле FaceImage має тип даних bytea спеціально для зберігання зображень в байтовому поданні.

Таблиця виглядає наступним чином

```
                                  Table "public.idendtities_users"
  Column   |         Type          | Collation | Nullable |                    Default
-----------+-----------------------+-----------+----------+-----------------------------------------------
 id        | integer               |           | not null | nextval('idendtities_users_id_seq'::regclass)
 name      | character varying(50) |           |          |
 faceimage | bytea                 |           |          |
Indexes:
    "idendtities_users_pkey" PRIMARY KEY, btree (id)
```

Рисунок 3.5 Таблиця “Users”

Додатково було розроблено таблицю Violators, яка також містить поля id, name, faceimage, але додатково ще має таке поле як datetime, яке має тип даних timestamp та вказує на те, коли саме було зафіксоване порушення.

```
                                    Table "public.violators"
  Column   |            Type             | Collation | Nullable |                Default
-----------+-----------------------------+-----------+----------+---------------------------------------
 id        | integer                     |           | not null | nextval('violators_id_seq'::regclass)
 name      | character varying(50)       |           |          |
 faceimage | bytea                       |           |          |
 datetime  | timestamp without time zone |           |          |
Indexes:
    "violators_pkey" PRIMARY KEY, btree (id)
```

Рисунок 3.6 Таблиця “Violators”

3.4 Тестування програмного продукту

Для початку було вирішено прокласифікувати звичайні фото людей в медичних масках та без, щоб було розуміння, що все працює так як треба

На першому фото можемо побачити шість людей, четверо з яких знаходяться в медичних масках

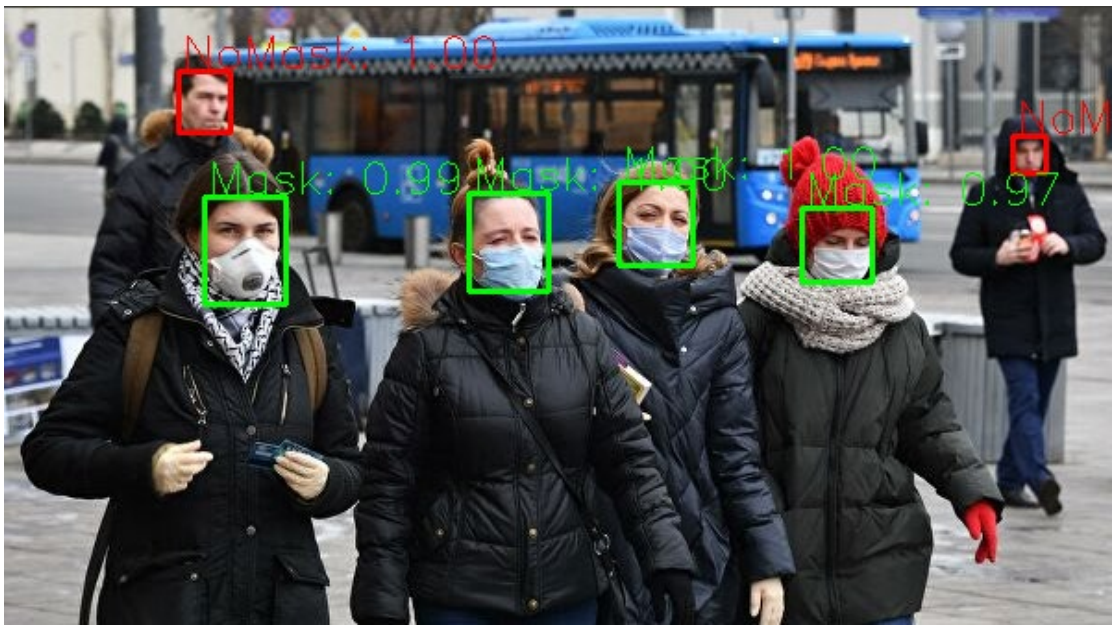


Рисунок 3.7 Перше тестове зображення

На цьому фото можна побачити, що задній фон трохи розмит, але система все одно робить правильно класифікацію з великою точністю.

Друге фото було прокласифіковано для того, щоб відбросити варіанти False Positive, коли людина не одягає маску, а система вивляє маску на зображенні.

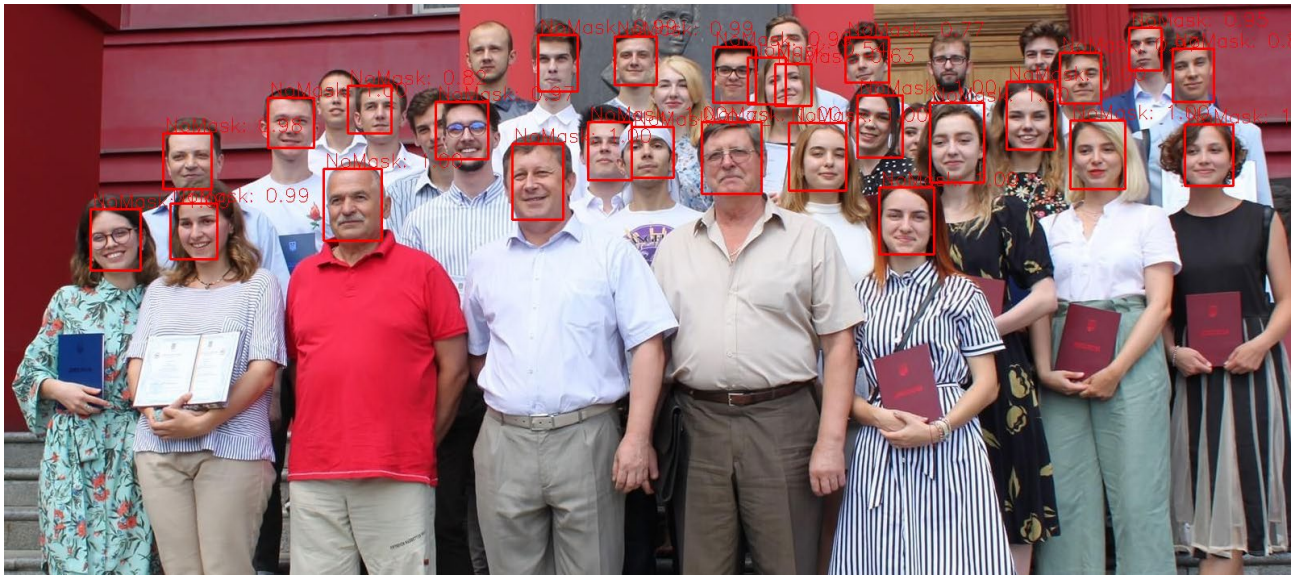


Рисунок 3.8 Друге тестове зображення

На рисунку 3.8 видно, що немає жодного False Positive, але можна побачити, що в останньому ряду є декілька людей, яких система взагалі не розпізнала.

Другим етапом тестування системи було записане відео, яке було подано на вхід системи. Після успішного відпрацювання система зберігає оброблений відеофайл та кадри, на яких людина була без маски.

Для початку хотілося б зазначити, що модель з великою точністю правильно класифікує, що людина в масці при будь-якому повороті голови, що також є великим плюсом. Для того, щоб в цьому переконатися, було виділено 2 фрейми:

Рисунок 3.9 Третє тестове зображення

Далі необхідно протестувати, як модель буде визначати наявність кольорових масок, а саме розових, білих, чорних. Для початку було взято зображення трьох людей у розових масках. Можна побачити, що система виявила маски на їх обличчі з максимальною впевненістю

Рисунок 3.10 Четверте тестове зображення

На рисунку 3.11 зображені люди в білих масках. Система також з досить великою впевненістю виявила маски на обличчях, але можна побачити одне False Negative, що є не зовсім гарним показником, але це все ж таки краще аніж False Positive. Виходячи з цього, треба ще попрацювати з моделлю детекції людини, бо видно, що система «побудувала» не зовсім доречний bounding box.

Рисунок 3.11 П’яте тестове зображення

Останній колір, який, треба перевірити – є чорний. На рисунку 3.12 можна побачити людину, яка кудись дивиться. Система з максимальною впевненістю виявила маску на його обличчі, незважаючи на те, що його голова повернута до іншої сторони.

Рисунок 3.12 Сьоме тестове зображення

Останнім етапом тестування – є тестування ідентифікації людини. Для цього тестове фото було добавлене до бази даних, ім’я користувача було добавлене як Pavlo Holubchykov. Тестове зображення виглядає наступним чином

Рисунок 3.13 Тестове зображення

Таким чином виглядає вікно повідомлень з повідомленнями про користувачів з відсутніми масками. Це месседжер Microsoft Teams, в якому створений спеціальний канал під назвою “face mask detection”, куди надсилаються зображення порушників. Також хотілося б зазначити, що до каналу надсилаються лише ті зображення, в яких confidence більше ніж 90%, бо якщо в реальному часі система виявляє щось як людина без маски, а насправді це щось інше, щоб не було зайвих спрацювань та повідомлень, які будуть лише турбувати адміністраторів чи охорців.

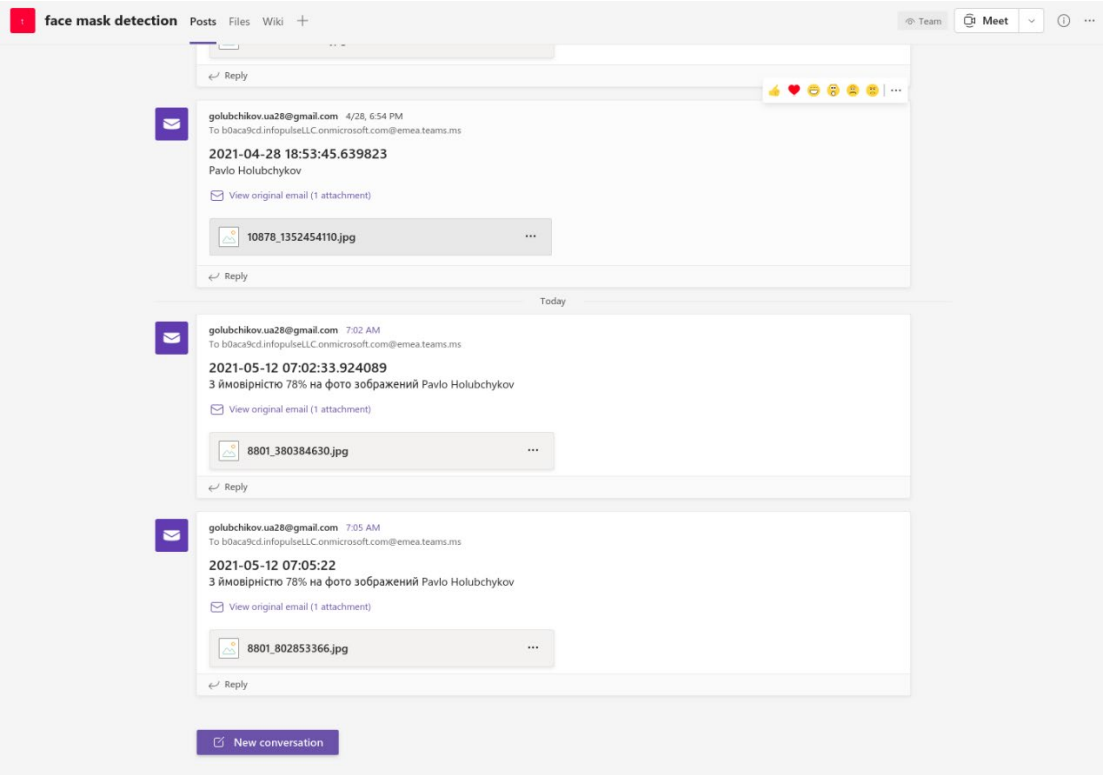


Рисунок 3.15 Інтерфейс месседжера

З останнього повідомлення можна побачити, що система ідентифікувала особу як Pavlo Holubchykov з ймовірністю 78%, що є досить великим показником, бо пів обличчя закриває маска. До речі, на цьому зображенні також можна побачити, що система виявляє, що маски немає навіть, якщо вона вдягнута не належним чином, тобто не закриває повністю ніс.

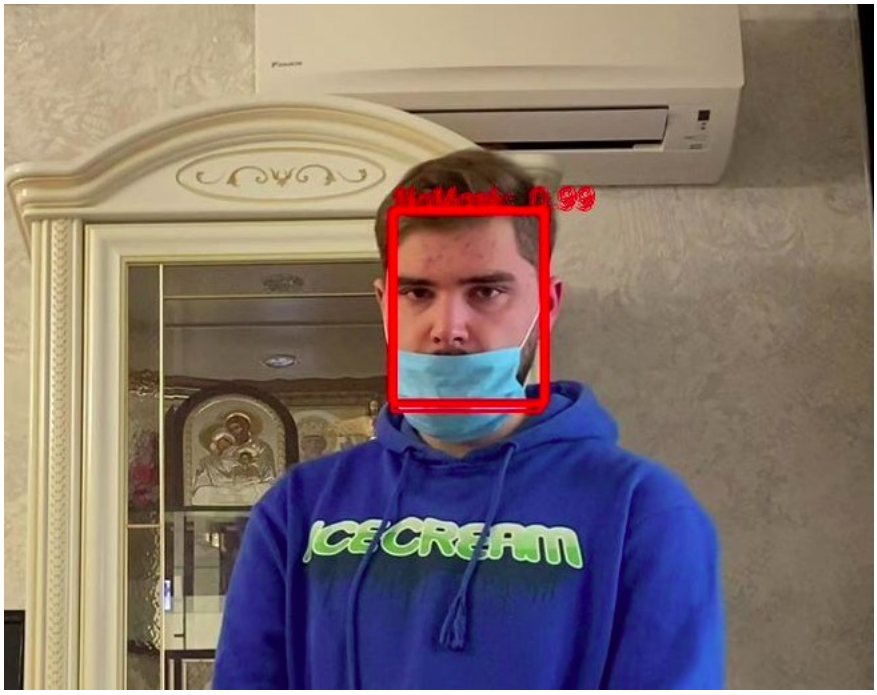


Рисунок 3.16 Восьме тестове зображення

ВИСНОВКИ

У даній роботі була розглянута проблема автоматичної ідентифікації осіб без медичних масок. Актуальність даної проблеми обумовлена зростаючою кількості хворих та постійною появою все більш строгих правил стосовно знаходження в публічних місцях. Було проаналізовано основні розроблені системи по виявленню медичних масок на обличчі людини та розглянуто основні методи, які використовувалися в цих системах.

Було розроблено програмний модуль інтелектуальної системи ідентифікації осіб без медичних масок та виконано поставлені завдання. Створено модель на основі згорткових нейронних мереж та методу Single Shot Multibox Detector (SSD). Розроблена модель показала досить високу точність та швидкість роботи при застосуванні її у реальному часі. Система точно ідентифікує особу без медичної маски та одразу відправляє зображення цієї особи у спеціальний чат.

Було розроблено базу даних для зберігання інформації про людей, яких треба ідентифікувати. Для ідентифікування використовувався сервіс Azure Face Api, який досить швидко обробляє вхідні зображення та дозволяє не втрачати час.

Отже, результатом даної роботи є програмний модуль інтелектуальної системи, що дозволяє (майже без помилок) виявляти медичні маски на обличчі та у разі їх відсутності надсилати в спеціальний месседжер повідомлення про порушника.

Безумовно, метод в подальшому можна вдосконалювати та налагоджувати. В першу чергу, необхідно створити більшу кількість шарів згорткової нейронної мережі для досягнення ще більш високої точності та проаналізувати різні існуючі варіанти функції активації, що також може позитивно впливати на точність.

Понад п'ятдесят країн світу нещодавно ініціювали носіння масок для обличчя в обов'язковому порядку. Люди повинні закривати обличчя в громадських місцях, супермаркетах, громадському транспорті, офісах та магазинах. Розроблене програмне забезпечення передбачено застосовувати

разом з будь-якою існуючою USB та IP-камерою для виявлення людей без маски. Слід зазначити, що отриманий відеопотік в реальному часі може бути реалізований як вебзастосунок або звичайний додаток, щоб оператор у реальному часі за потребою мав змогу подивитися відеопоток та отримувати повідомлення про порушників. Крім того, може бути реалізована також система сигналізації для подачі звукового сигналу, коли хтось без маски заходить у зону. Це програмне забезпечення також можна “підключити” до вхідних воріт або дверей, і тоді до приміщення зможуть входити лише люди в масках для обличчя.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Bengio Y., Lecun, Y. Convolutional Networks for Images, Speech, and Time-Series.1997.URL:https://www.researchgate.net/profile/Yann_Lecun/publication/2453996_Convolutioa_Networks_for_Images_Speech_and_TimeSeries/links/0deec519dfa2325502000000.pdf

2. Bengio Y., Lecun, Y. Convolutional Networks for Images, Speech, and Time-Series.1997.URL:https://www.researchgate.net/profile/Yann_Lecun/publication/2453996_Convolutional_Networks_for_Images_Speech_and_TimeSeries/links/0deec519dfa2325502000000.pdf

3. Khandelwal R. Convolutional Neural Network (CNN) Simplified. 2018. URL :https://medium.com/datadriveninvestor/convolutional-neural-network-cnnsimplified-ecafd4ee52c5

4. Stockman G. C., Shapiro L. G. Computer Vision. Prentice Hall, February 2001. 609 p.

5. Rao D., McMahan B. Natural Language Processing with PyTorch: Build Intelligent Language Applications Using Deep Learning. O'Reilly Media, 2019. 256 p.

6. Springenberg J. T., Dosovitskiy A., Brox T., Riedmiller M. Striving for Simplicity: The All Convolutional Net. arXiv preprint: 1412.6806. 2014.

7. Scherer, Dominik; Müller, Andreas C.; Behnke, Sven (2010). Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition. / 20th International Conference on Artificial Neural Networks (ICANN), Thessaloniki, Greece: Springer, 2010. с. 92–101.

8. Graham B. Fractional max-pooling. arXiv preprint: 1412.6071. 2014.

9. Khan S., Rahmani H., Ali Shakh S. A., Bennamoun M. A Guide to Convolutional Neural Networks for Computer Vision. Morgan & Claypool Publishers, 2018. 207 p.

10.Brownlee J. Loss and Loss Functions for Training Deep Learning Neural Networks, 2019. URL : https://machinelearningmastery.com/loss-and- loss-functions-for-training-deep-learning-neural-networks/

11.Brownee J. How to Choose Loss Functions When Training Deep Learning Neural Networks. 2019. URL : https://machinelearningmastery.com/how-to-choose-loss-functions-when- training-deep-learning-neural-networks/

12.Goodfellow I., Bengio Y., Courville A. Deep Learning (Adaptive Computation and Machine Learning series). The MIT Press, 2016. 775 p.

13.Krizhevsky A., Sutskever I., Hinton G. E. ImageNet Classification with Deep Convolutional Neural Networks. Communications of the ACM, Vol. 60 Issue 6, June 2017 : ACM New York, NY, USA. P. 84-90. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep- convolutional-neural-networks.pdf

14.Gonzalez R. C., Woods R. E. Digital Image Processing. Prentice Hall, Upper Saddle River, New Jersey, 2008. 954 p.

15.Veererajan T. Probability, Statistics And Random Processes. Tata McGraw-Hill, 2002. 693 p.

16.Simmons J. P., Drummy L. F., Bouman C. A., De Graef M. Statistical Methods for Materials Science: The Data Science of Microstructure Characterization. CRC Press, 2019. 514 p.

17.Bottacchi S. Multi-Gigabit Transmission over Multimode Optical Fibre: Theory and Design Methods for 10GbE Systems. John Wiley & Sons, Chichester, UK, 2006. 670 p.

18.De Wos W. H., Munck S., Timmermans J.-P. Focus on Bio-Image Informatics. Springer, 2016. 272 p.

19.Gelman L. Advances in Electrical Engineering and Computational Science. Springer Science & Business Media, 2009. 726 p.

20.Волошин Олексій. Переваги та недоліки мови програмування. 2020. URL: https://blog.ithillel.ua/ua/articles/perevagi-i-nedoliki-movi-python

21.Manjon J. V., Coupe P. MRI Denoising Using Dep Learning. / Patch- Based Techniques in Medical Imaging: 4th International Workshop, Patch-MI 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings. Springer, 2018. p. 12-19.

22.Donoho D. L., Johnstone I. M. Ideal Spatial Adaptation by Wavelet Shrinkage. / Biometrika 81.3, 1994. p. 425-455.

23.Tiagi V. Understanding Digital Image Processing. CRC Press, 2018. 368 p.

24.Cohen R. Signal Denoising Using Wavelets. Project Report. – Department of Electrical Engineering Technion, Israel Institute of Technology, Haifa, 2012.

25.Chang C. G., Yu B., Vetterli M. Adaptive Wavelet Thresholding for Image Denoising and Compression. / IEEE Trasactions on image processing, vol. 9, no. 9, 2000. p. 1532-1536

26.Getreuer P. Rudin–Osher–Fatemi Total Variation Denoising using Split Bregman. / Image Processing On Line, 2012.

27.Goldstein T., Osher S. The Split Bregman Method For L1 Regularized Problems. / SIAM Journal on Imaging Sciences 2(2), 2009. p. 323-343.

28.Chambolle A. An algorithm for total variation minimization and applications. / Journal of Mathematical Imaging and Vision. Springer, 2004. p. 89-90.

29.Tomasi C., Manduchi R. Bilateral Filtering for Gray and Color Images. / Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271). Bombay, 2002.

30.Paris S., Kornprobst P., Tumblin J., Durand F. Bilateral Filtering: Theory and Applications. Now Publishers, 2009. 88 p.

31.Larsson F., Felsberg M. Using Fourier Descriptors and Spatial Models for Traffic Sign Recognition. / In Proceedings of the 17th Scandinavian Conference on Image Analysis, SCIA 2011, LNCS 6688. p. 238-249.

32.Opelt A., Fussenegger M., Pinz A., Auer P. Generic Object Recognition with Boosting. Graz University of Technology, 2004.

33.Classification: Accuracy. / Classification. URL : https://developers.google.com/machine-learning/crash- course/classification/accuracy

34.Huotari J. Why Loss and Accuracy Metrics Conflict? 2018. URL : http://www.jussihuotari.com/2018/01/17/why-loss-and-accuracy-metrics- conflict/

35.Subramanian V. Deep Learning with PyTorch: A practical approach to building neural network models using PyTorch. Packt Publishing, Birmingham, UK, 2018. 238 p.

36.Howard A. G., Zhu M., Chen B., Kalenichenko D., Wang W., Weyand T., Andreetto M., Adam H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv preprint: 1704.04861. 2017.

37.Iandola F. N., Han S., Moskewicz M. W., Ashraf K., Dally W. J., Keutzer K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. arXiv preprint: 1602.07360. 2016.

38.Chollet F. Xception: Deep Learning with Depthwise Separable Convolutions. arXiv preprint: 1610.02357. 2017.