Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

АНАЛІЗ ТА ЗАСТОСУВАННЯ МЕТОДІВ ОЦІНКИ КІЛЬКОСТІ ЛЮДЕЙ У ВІДЕОРЯДІ В ЦІЛЯХ МОНІТОРИНГУ МИРНИХ ЗІБРАНЬ
(тема)

Виконав:
студент 2 курсу, групи ІНФм-19-1
Богдан Д. І.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник проф. Машталір С. В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентові Богдану Денису Ігоровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Аналіз та застосування методів оцінки кількості людей у відеоряді в цілях моніторингу мирних зібрань

затверджена наказом по університету від «_23__» __жовтня_____2020 року № 1428Ст.

2. Термін подання студентом роботи до екзаменаційної комісії _23_ __листопада__ 2020 р.

3. Вихідні дані до роботи *Теоретичні відомості про методи оцінки кількості людей, бібліотека PyTorch, мова програмування Python, згорткові нейронні мережі*

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд методів оцінки кількості людей у відеоряді
2. Згорткові нейронні мережі для задачі розрахунку густоти людей
3. Огляд і порівняння існуючих моделей нейронних мереж

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) *Мирні зібрання є одним із засобів відстоювання людиною і громадянином своїх прав, свобод та інтересів у демократичному суспільстві. Для складання реальної картинки суспільства і надання реальних даних про відвідування мирних зібрань необхідно мати інформацію про кількість людей, що приймають у них участь. Підрахунок людей на відео вручну є складним та дуже довготривалим процесом. У зв'язку з цим виникає потрібність в автоматизації цього процесу за допомогою системи оцінки кількості людей у натовпі, який би допоміг організаціям з громадського спостереження у отриманні достовірних даних.*

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на атестаційну роботу	23.10.2020	
2	Аналіз завдання, підбір літератури	23.10.20-24.10.20	
3	Аналіз літератури з досліджуваної проблеми	24.10.20-25.10.20	
4	Аналіз технічних засобів	25.10.20-26.10.20	
5	Розробка методів	26.10.20-02.11.20	
6	Програмна реалізація	03.11.20-10.11.20	
7	Оформлення пояснювальної записки	11.11.20-16.11.20	
8	Перевірка на плагіат	26.11.20	
9	Рецензування	25.11.20	
10	Підготовка презентації та доповіді	26.11.20	
11	Занесення роботи в електронний архів	30.11.20	
12	Попередній захист атестаційної роботи	09.12.20	

Дата видачі завдання _23___ ___жовтня_ _2020_ р.

Студент _____
(підпис)

Керівник роботи _____ ___проф. Машталір С. В._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до атестаційної роботи: 79 с., 30 рис., 33 джерел.

ШТУЧНИЙ ІНТЕЛЕКТ, КОМП'ЮТЕРНИЙ ЗІР, ОЦІНКА КІЛЬКОСТІ ЛЮДЕЙ, ОЦІНКА ГУСТОТИ ТОВПИ, ЗГОРТКОВА НЕЙРОННА МЕРЕЖА, ГІСТОГРАМА НАПРЯМЛЕНИХ ГРАДІЄНТІВ.

Метою дослідження є реалізація та аналіз методів оцінки кількості людей у відеоряді в цілях моніторингу мирних зібрань.

Об'єктом дослідження є сфера громадського спостереження, а саме процес оцінки кількості учасників мирних зібрань.

Проведено дослідження методів оцінки кількості людей у відеоряді на основі розпізнавання образів та на основі оцінки карт щільності згортковою нейронною мережею. В основі згорткових нейронних мереж лежить операція згортки – процес додавання кожного елемента зображення до його сусідів, зважених ядром. В результаті роботи згорток отримується набір деталей зображення від більш абстрактних (криві) до більш конкретних (геометричні фігури).

У результаті роботи здійснена програмна реалізація системи для оцінки кількості людей в відеоряді.

ARTIFICIAL INTELLIGENCE, COMPUTER VISION, PEOPLE DENSITY ESTIMATION, CROWD DENSITY ESTIMATION, CONVOLUTIONAL NEURAL NETWORK, HISTOGRAM OF ORIENTED GRADIENTS.

The aim of the research is to implement and analyze methods of crowd density estimation in a video with the purpose of monitoring of peaceful protests.

The object of research is the field of public observation, namely the process of estimating a number of demonstrators in peaceful protests.

In the scope of this work an analysis of different methods of crowd density estimation based on object recognition and density map estimation by a convolutional neural network was carried out. Convolutional neural networks are based on the convolution operation (kernel, convolution matrix, mask) – process of adding each element of the image to its local neighbours, weighed by the kernel. The result of the operation is a set of details on the image from more abstract (curves) to more concrete (geometric shapes).

As a result of this work an implementation of the system to estimate crowd density.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

CNN – згорткова нейронна мережа

ШНМ – штучна нейронна мережа

SVM – метод опорних векторів

ROI – область інтересу

SOM – саморганізовні карти

ВСТУП

Однією із основних областей розробок сучасної кібернетики є області машинного навчання, розпізнавання образів та комп’ютерного зору.

В останній час велика кількість комерційних проектів включають у себе технології штучного інтелекту. Теорія розпізнавання образів застосовується у обчислювальних машинах, що допомагають поставити медичний діагноз та рекомендоване лікування, в обробці нейробіологічних сигналів, обробці зображень, промисловому контролі тощо. Наприклад, автоматичний аналіз та класифікація мікрофотографій біологічних тканин можуть бути застосовані при діагностиці онкозахворювань. Методами теорії розпізнавання образів можна аналізувати та класифікувати такі клінічні дані, як електрокардіограми та електроенцефалограми. Інше застосування теорії розпізнавання образів – аналіз сигналів у корі головного мозку, що є відкликом на зорові на слухові сигнали.

Існують різні підходи для підрахунку натовпу. Популярним методом є підрахунок шляхом розпізнавання, який виявляє пішоходів за допомогою сканування простору зображення за допомогою детектора, навченого локальним особливостям зображення. Альтернативним підходом є підрахунок шляхом кластеризації, який передбачає скупчення натовпу окремих сутностей, кожна з яких має унікальні, але цілісні закономірності руху які можна згрупувати для апроксимації кількості людей. Інший метод натхненний здатністю людей визначати щільність з першого погляду без підрахунку кількості людей. Цей підхід відомий як підрахунок за допомогою регресії, яка підраховує людей у натовпі, навчаючись зіставляти низькорівневі особливості зображення з щільністю натовпу. Останнім розглянутим підходом є підрахунок з використанням нейронних мереж, що останнім часом набув достатньої популярності.

У цій роботі надані огляд, порівняльна оцінка та критичний аналіз методів комп’ютерного зору для підрахунку натовпу.

1 ІСНУЮЧІ ПІДХОДИ ВИРІШЕННЯ ЗАДАЧІ І ПОСТАНОВКА ЗАДАЧІ

1.1 Огляд існуючих методів

Методи підрахунку натовпів можна розділити на три групи: підрахунок шляхом розпізнавання, підрахунок шляхом кластеризації, підрахунок регресією та підрахунок з використанням нейронних мереж.

1.1.1 Підрахунок шляхом розпізнавання

Підрахунок шляхом розпізнавання потребує знайдення кожної людини на зображенні використовуючи детектор об’єктів [1 – 3].

Даний підхід можна розбити на три групи в залежності від того, за якою ознакою відбувається розпізнавання:

- монолітне розпізнавання;
- розпізнавання за частиною об’єкту;
- розпізнавання за формою.

Монолітне розпізнавання є найбільш простим для розуміння підходом. Класифікатор навчається на наборі зображень людини в повний ріст. Популярними ознаками для розпізнавання людини в повний ріст є вейвлети Хаара або градієнтні підходи такі як гістограма напрямлених градієнтів. Від вибору класифікатора залежать якість розпізнавання об’єктів та швидкодія застосунку. Нелінійні класифікатори дають якісний результат, але є повільнішими за лінійні класифікатори. Навчені класифікатори застосовують до зображення і результати класифікації фільтруються за показником впевненості. Ті об’єкти, що не задовольняють вимозі відкидаються. Монолітне розпізнавання може давати якісні результаті у натовпах із низькою щільністю, але якість результатів падає якщо застосовувати цей підхід до натовпів з

високою щільністю або коли інші об’єкти частково перекривають людей [4] (рис. 1.1).

Розпізнавання за частиною об’єкту є способом вирішення проблеми розпізнавання людей у натовпі з великою щільністю або при наявності об’єктів що частково можуть їх перекривати. Класифікатори у цьому випадку навчаються розпізнавати лише частину людського тіла, наприклад голову та плечі для оцінки кількості людей. Розпізнавання лише за головою часто може бути недостатнім через варіації у формі та зовнішньому вигляді. Результати розпізнавання можуть бути в подальшому покращені слідкуючи за розпізнаними об’єктами впродовж часу і відкидаючи ті об’єкти, що не демонструють чіткий шлях пересування.

Рисунок 1.1 – Розпізнавання пішоходів шляхом:
(а) монолітного розпізнавання, (б) розпізнавання за частиною об’єкту,
(в) розпізнавання за формою

Розпізнавання за формою [5] використовує параметризовані людські силуети що складаються з еліпсів і виконує розпізнавання стохастичним методом шляхом пошуку конфігурації кількості і форм силуетів, що найкраще підходять до маски переднього плану відеоряду.

Всі ці методи [6 – 8] потребують значної обчислювальної потужності та дають гірші результати коли не видно частини об’єкту, що розпізнається.

1.1.2 Підрахунок шляхом кластеризації

Підрахунок шляхом кластеризації припускає те, що переміщення особливостей зображення з часом є рівномірним і траєкторії переміщення можуть бути згруповані разом для отримання незалежно рухомих об’єктів. Підрахунок шляхом кластеризації [9] як правило використовують методи навчання без учителя і припускають те, що траєкторії переміщення об’єктів є рівномірними, що може дати хибні результати якщо об’єкти є нерухомими або декілька об’єктів мають схожі траєкторії і можуть бути зараховані як один.

Варто зазначити, що ці методи не можна застосовувати до статичних зображень – вони працюють лише із безперервними кадрами відеоряду.

1.1.3 Підрахунок регресією

Підрахунок регресією оцінює щільність натовпу базуючись на колективних ознаках натовпів. Оскільки підрахунок регресією не використовує сегментацію або трекінг окремих об’єктів він часто застосовується для підрахунку людей у щільних натовпах та інших оточеннях де можливість розпізнавання індивідуальних об’єктів є значно обмеженою.

Основний принцип роботи таких методів можна описати наступним шляхом: спочатку виділяються ознаки низького рівня, такі як пікселі переднього плану та границі об’єктів у кожному кадрі відеоряду [10]. З цих ознак отримують колективні ознаки, такі як площа переднього плану, кількість границь на зображенні та текстури на зображенні [11, 12]. До цих ознак застосовується модель лінійної регресії що зіставляє їх з можливою кількістю людей, а саме використовується функція що моделює зміни параметру щільності людей в залежності від змін колективних ознак.

Деякі дослідження використовують трекер Канаде-Лукаса-Томасі (KLT), щоб отримати багатий набір низькорівневих відстежуваних функцій,

та кластеризує траєкторію, щоб визначити кількість людей на сцені, яка відстежує місцеві особливості та групує їх у кластери за допомогою байєсівської кластеризації. Іншим тісно пов'язаним методом є [13], який включає ідею постійності ознак у підрахунок за допомогою системи виявлення. Спочатку метод генерує набір особистих гіпотез натовпу на основі виявлення голови. Потім гіпотези ітеративно уточнюються, присвоюючи невеликі ділянки натовпу гіпотезам на основі постійності полів руху та кольору одягу.

Вищезазначені методи уникають контрольованого навчання або явного моделювання особливостей зовнішності, як при підрахунку за допомогою парадигми виявлення. Тим не менше, парадигма передбачає узгодженість руху, отже, помилкова оцінка може виникнути, коли люди, що залишаються статичними на сцені, демонструючи стійкі артикуляції, або два об'єкти, що мають спільні траєкторії ознак з часом.

Одна з найперших спроб вивчити використання методу регресії для оцінки щільності натовпу Девісом та ін. [14]. Спочатку вони отримують особливості низького рівня такі як пікселі переднього плану та грані на кожному кадрі відеоряду. Потім отримуються цілісні властивості такі як площа переднього плану та загальна кількість країв. Отже, модель лінійної регресії використовується для встановлення прямого відображення між цілісними візерунками та реальними людьми. Зокрема, спеціальна функція використовується для моделювання того, як змінюється вхідна змінна (тобто щільність натовпу), коли змінюються цільові змінні (тобто цілісні моделі). З цієї інформації потім можна прогнозувати сподівання щільності натовпу, враховуючи отримані особливості наступного кадру. З часу роботи Девіса та ін. [15], різноманітні методи були запропоновані за тією ж ідеєю з вдосконаленими наборами функцій або більш досконалими моделями регресії, але все ще мають схожий спосіб перетворення інформації. Основними компонентами регресійного метода є представлення ознак, геометрична корекція та моделювання регресії.

Представлення ознак стосується вилучення, відбору та трансформації низькорівневих візуальних властивостей зображення або відео для побудови проміжного входу в регресійну модель. Популярним підходом є поєднання кількох ознак із взаємодоповнюючим характером, щоб сформувати великий банк ознак. Найпоширенішим способом подання даних для оцінки щільності натовпу є сегмент переднього плану, який можна отримати шляхом віднімання фону [16]. З виділеного сегмента переднього плану можна отримати різні цілісні ознаки, наприклад:

– площа – загальна кількість пікселів у сегменті;

– периметр – загальна кількість пікселів по периметру сегмента;

– відношення периметра до площі – відношення між периметром сегмента та площею, яке вимірює складність форми сегмента;

– орієнтація по краю периметра – гістограма орієнтації периметра відрізка.

Різні дослідження продемонстрували обнадійливі результати з використанням особливостей, заснованих на сегментах, незважаючи їх простоту. Однак, під час впровадження слід врахувати кілька моментів. По-перше, щоб зменшити помилкові сегменти першого плану з інших регіонів, можна обмежити аналіз всередині області, що цікавить (ROI – Region of Interest) [17], яка може бути визначена вручну або за підходом до акумуляції переднього плану. По-друге, різні сценарії можуть вимагати різних стратегій видобутку особливостей. Зокрема, динамічне віднімання фону може впоратися з поступовою зміною освітленості, але відчуває труднощі з ізоляцією людей, які тривалий час перебувають не переміщуються; віднімання статичного фону здатне сегментувати статичні об’єкти з фону, але сприйнятливе до зміни освітлення.

Хоча особливості переднього плану враховують загальні властивості сегмента, елементи краю всередині сегмента несуть додаткову інформацію про місцеві та внутрішні структури [18]. Натовпи низької щільності мають форму з грубими краями, тоді як сегменти з щільними скупченнями людей

мають більш складні і гладкі краї. Ребра можна виявити за допомогою крайового детектора, такого як детектор Кенні. Деякі загальні рисові особливості перелічені нижче:

– загальна кількість пікселів краю;

– орієнтація краю – гістограма крайових орієнтацій у відрізку;

– розмір Мінковського – фрактальний вимір Мінковського, який підраховує, скільки заздалегідь визначених елементів структурування потрібно для заповнення країв.

Проблемою, з якою зазвичай зустрічаються при підрахунку за допомогою регресійної системи, є перспективне спотворення, при якому далекі об’єкти здаються меншими, ніж ті, що знаходяться ближче до камери. Як наслідок, особливості (наприклад, область сегмента), витягнуті з одного і того ж об’єкта на різній глибині сцени, мали б величезну різницю у значеннях. Можна значно зменшити вплив перспективних спотворень, якщо розділити простір зображення на різні чарунки, кожна з яких моделюється функцією регресії; помилкові результати можна очікувати лише якщо використовується лише одна функція регресії для всього простору зображення.

Для вирішення цієї проблеми виконується геометрична корекція або нормалізація перспективи, щоб довести сприйманий розмір об’єктів на різній глибині до одного масштабу.

Після вилучення ознак та нормалізації перспективи виконується навчання моделі регресії для прогнозування підрахунку з урахуванням нормалізованих ознак.

Основним недоліком оцінки регресією є залежність навченої моделі від перспективи зображення, тобто якщо застосувати модель, що була навчена на одній перспективі, до зображень із іншою перспективою то результат матиме похибки [19].

1.1.4 Підрахунок з використанням нейронних мереж

Використання штучних нейронних мереж в задачах комп'ютерного зору набуло достатньої популярності останнім часом. У зв'язку з цим останні роботи в області оцінки кількості людей використовують нейронні мережі, що дають гарні результати [20 – 22].

В процесі навчання нейронної мережі вона знаходить сховані ознаки та шаблони ознак, що можуть бути використані при застосуванні мережі до зображення або відеоряду. Це означає, що випадки які дуже важко передбачити та опрацювати, наприклад як існування на зображенні однієї особливості в різних масштабах, також можуть бути використані для підрахунку кількості людей.

Основною перевагою використання нейронних мереж для підрахунку кількості людей є те, що навчена нейронна мережа може бути легко застосована на зображеннях і відеорядах із різними перспективами, при цьому даючи гарний результат.

1.2 Гістограма напрямлених градієнтів

Гістограма спрямованих градієнтів – це методика, що використовується в комп'ютерному зорі і обробці зображень з метою розпізнавання об'єктів. Даний підхід заснований на підрахунку напрямків градієнта яскравості (інтенсивності) в локальних областях зображення і базується на тому факті, що розподіл градієнтів яскравості на будь-якому ділянці зображення дає уявлення про зовнішній вигляд і форму об'єкта, розташованого на цьому ділянці (навіть без урахування точного розташування цих напрямків). Суть методу полягає в тому, що зображення розбивається щільною рівномірною сіткою на області, для кожної з яких будується локальна гістограма напрямків градієнтів яскравості. Для забезпечення інваріантності по відношенню до

висвітлення гістограми піддаються нормалізації за контрастом зі значенням яскравості, що обчислюється за більшим фрагментом. Сукупність побудованих нормалізованих гістограм буде дескриптором об’єкта. Такі дескриптори інваріантні до висвітлення, геометричних і фотометричних перетворень (за винятком орієнтації самого об’єкта). Останнім етапом у розпізнаванні є класифікація дескрипторів за допомогою навчання з учителем.

Реалізація методу гістограми напрямлених градієнтів складається з наступних етапів:

- обчислення градієнту;
- групування напрямків;
- дескриптори блоків;
- нормування блоків;
- метод опорних блоків.

1.2.1 Обчислення градієнту

На цьому етапі для двовимірної поверхні градієнт виводиться шляхом обчислення похідної в горизонтальному на вертикальному напрямку. Для зображення, обчислення градієнта складається, головним чином, з вимірювання для кожного пікселя варіації його навколишніх пікселів. Ця варіація обчислюється шляхом застосування кореляційних масок в горизонтальному та вертикальному напрямках. Маска – це термін, що використовується у кореляційній фільтрації.

Фільтрація – це процес застосування модифікацій до зображення. Для того, щоб застосувати ці зміни, на кожному пікселі використовується фільтр / маска / ядро. Маска, яку також називають ядром і фільтром – це просто матриця, що містить ваги, на які кожен піксель впливає оточуючими його пікселями.

Отже, маска використовується для повторного обчислення значення кожного пікселя на зображенні. Новим значенням є сума сусідніх пікселів з урахуванням ваги маски. Цей тип фільтрації називається кореляційною фільтрацією.

Оскільки зображення можуть бути кольоровими, похідні в горизонтальному та вертикальному напрямках обчислюються для трьох каналів для кожного пікселя. Отже, для зображення розміром 64×128 пікселів, отримуються три матриці 64×128. Кожна матриця описує набір градієнтів для кожного із каналів (RGB). Щоб отримати загальні градієнти, значення для конкретного каналу градієнтів (RGB) усереднюються.

Коли градієнт виведений, для отримання певних подробиць про чіткість меж використовується величина результуючого вектору. Кут $\theta = \arctan(g_y / g_x)$, а величина градієнта становить $g = \sqrt{g_x^2 + g_y^2}$. Кут / напрямок градієнта, як правило, становитиме від $0°$ до $360°$ [15] (рис. 1.2). Виявлення людини на зображенні в основному на основі виявлення країв, що не вимагає знання знаку градієнта. Таким чином, не важливо знати напрямок градієнту на відміну від інтенсивності куту. Іншими словами, градієнти слід перетворити з основи $0° - 360°$ на $0° - 180°$.

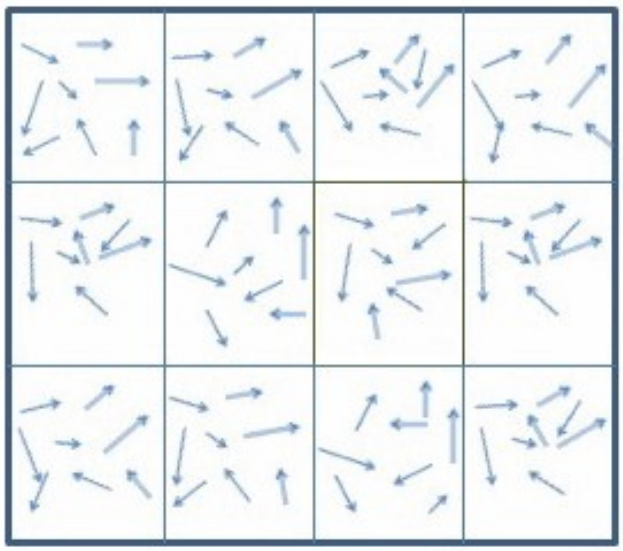

Рисунок 1.2 – Напрямок градієнта від 0 до $360°$

Рішенням для такого перетворення є додавання π до негативних значень кута, оскільки θ отримується в радіанах (для всіх $\theta \leq 0, \theta = \theta + \pi$).

1.2.2 Групування напрямків

Після отримання градієнтів, що належать до кожного з пікселів, зображення розбивається на чарунки розміром 8×8 пікселів. Потім для кожної клітини будуємо чарунок-гістограму.

Гістограма буде структурою даних, що містить 9 чарунків. Діапазон гістограми коливається від $0°$ до $180°$ (рис. 1.3).

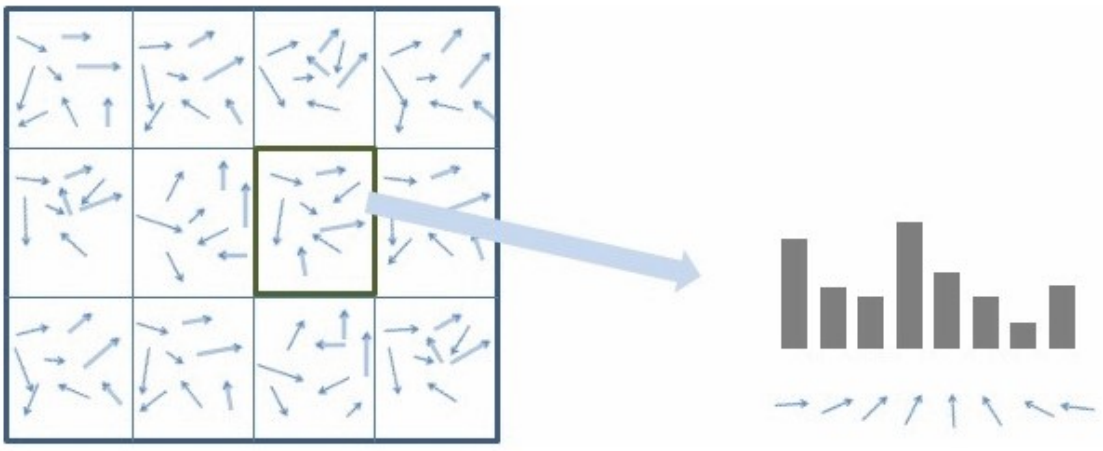

Рисунок 1.3 – Зменшення діапазону гістограми

Діапазон кожного окремого чарунку становить $180 \div 9 = 20°$. Причина наявності дев'яти чарунків заснована на ефективності гістограми на 9 чарунків краща, ніж гістограма із меншою або більшою кількістю чарунків. Менша кількість призведе до пропускання деяких важливих кутів, а більша призведе до надлишку інформації і перенавчання.

Для визначення внеску кожного кута до чарунків використовується лінійна інтерполяція відносно центру. Спочатку ідентифікуються два чарунки, до яких кут, швидше за все, буде відноситися. Потім внесок кута до кожного

із чарунків обчислюється за допомогою лінійної інтерполяції щодо центру цих чарунків. Це дає кращу точність, аніж вибір лише одного чарунку.

1.2.3 Дескриптори блоків

На цьому кроці блок визначається як набір 2×2 чарунків (2 верхні та 2 нижні сусідні чарунки), тому загалом блок має $8 \times 8 \times 4$ пікселів. Блок використовується для визначення чарунків, гістограми яких повинні бути нормалізовані разом. Суть цього полягає у більшій послідовності з точки зору зміни інтенсивності та деяких інших факторів.

Блок проходить по зображенню з кроком 1, як у горизонтальному, так і у вертикальному напрямку. Отже, будуть отримані блоки розміром 7×15. Оскільки в кожному блоці є 4 чарунки, він має 4 гістограми.

Після нормалізації блоків, гістограми всіх блоків $(7 \times 15 \times 4)$ збираються в одну гістограму напрямлених градієнтів, яка має розмір $7 \times 15 \times 4 \times 9 = 3780$.

1.2.4 Метод опорних блоків

Набір дескрипторів (3780 значень) подається на вхід до опорно векторної машини (англ. Support Vector Machine – SVM), який генерує модель. Рішення щодо відношення об’єкту до класу виконується безпосередньо функцією прийняття рішення SVM. В основному для класифікації дескрипторів використовується функція лінійного ядра.

1.3 Штучна нейронна мережа

Штучна нейронна мережа – математична модель і її програмна або апаратна реалізація, побудована по принципу організації і функціонування біологічних нейронних мереж – мереж нейронних клітин живого організму. Це поняття виникло при вивченні процесів, що виникають у мозку і при спробах змоделювати ці процеси. Після розробки алгоритмів навчання отримані моделі стали використовувати у практичних цілях: в задачах прогнозування, для розпізнавання образів, в задачах управління тощо.

ШНМ є системою з'єднаних і взаємодіючих між собою простих процесорів (штучних нейронів). Такі процесори звичайно достатньо прості у порівнянні із процесорами, що використовуються в персональних комп'ютерах. Кожний процесор працює тільки із сигналами, котрі він отримує на вхід і сигналами, що він посилає іншим процесорам. Тем не менш, коли ці прості процесори з'єднують в достатньо великі мережі вони здатні виконувати достатньо складні задачі. Завдання нейромережі полягає в перетворенні вхідного вектору у вихідний вектор, що здійснюється вагою і топологією мережі.

1.3.1 Модель штучного нейрона та його функції активації

Штучний нейрон характеризується своїм поточним станом. Він володіє групою синапсів – односпрямованих вхідних зв'язків, з'єднаних з виходами інших нейронів, присутня така частина, як аксон – вихідний зв'язок штучного нейрона, з якого сигнал (збудження або гальмування) надходить на синапси наступних нейронів. Загальний вигляд штучного нейрона наведено на рисунку 1.4. Штучний нейрон спочатку імітує властивості, що дані біологічному нейрону. Тут множина вхідних сигналів, позначених $x_1, x_2, \ldots, x_N$, надходить на штучний нейрон. Ці вхідні сигнали, в сукупності

позначаються вектором X, приходить до синапсів біологічного нейрона. Кожен синапс характеризується величиною синоптичного зв'язку або його вагою w_i.

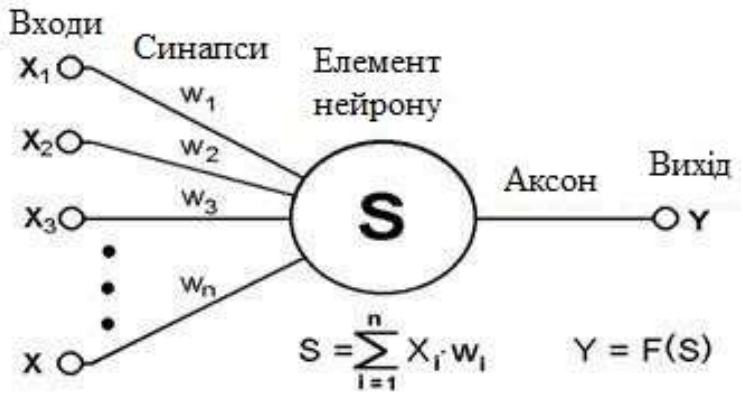

Рисунок 1.4 – Структура штучного нейрона

Для реалізації нелінійності при активації нейрона, його активність, крім різних видів суматорів і систем ваг на входах, визначається функцією одного аргументу – функцією активації. Нейрон в цілому реалізує скалярну функцію векторного аргументу, а вихідний сигнал нейрона визначається видом функції активації і може бути дійсним або цілим. Функція активації застосовується до зваженої суми постсинаптичних сигналів на вході нейрона. Таким чином, активність нейрона повністю визначається його параметрами – вагами і його функцією активації. Існує велика кількість передавальних функцій, що застосовуються на практиці при розробці нейронних мереж, деякі з них служать для реалізації нелінійності системи. Вибір тієї чи іншої функції часто залежить від умов завдання і структури мережі.

Порогова функція Хевісайда є самою простою кусково-лінійною передавальною функцією. Ця функція використовувалась в класичному персептроні і на сьогоднішній день використовується в основному в цілях навчання теорії нейронних мереж.

Лінійна функція активації. При використанні нескладної кусково-лінійної передавальної функції сигнал на виході нейрона є лінійно

зв’язаним із сумою сигналів на його вході. На сьогоднішній день лінійна функція на практиці використовуються дуже рідко.

Сигмоїдальна функція активації. Сигмоїд є монотонно зростаючою всюди диференційованою *S*-образною нелінійною функцією із насиченням. Вона забезпечує посилення слабких сигналів і запобігає насиченню сильних сигналів. Є однією з найбільш поширених передавальних функцій, часто використовується в нейронних мережах по сьогоднішній день. Введення сигмоїдальних функцій було обумовлено недостатньою гнучкістю класифікаторів на основі порогових передавальних функцій і дозволило перейти від жорсткої однорозрядної логіки до більш гнучкої поведінки і адаптивної параметризації нейронних мереж. Найбільш часто використовуваним прикладом сигмоїдальної функції є логістична передавальна функція:

$$\sigma(x) = \frac{1}{1 + e^{-x}}\ , \tag{1.1}$$

де x – вхід нейрона.

ReLU (Rectified Linear Unit). В неглибоких ШНМ використовуються нелінійні функції активації. Часто використовуваними різновидами сигмоїдальних і тангенціальних передавальних функцій є нелінійні, але на практиці навчання глибоких ШНМ такі функції можуть привести до проблем з загасанням або збільшенням градієнтів. Функція ReLU є випрямленою лінійною функцією і на даний момент вважається більш простим і ефективним з точки зору обчислювальної складності варіантом передавальної функції. Похідна ReLU дорівнює або 0, або 1, від чого її застосування запобігає розростанню і загасанню градієнтів, і призводить до проріджування ваг, що позитивно позначається на обчислювальній здатності ШНМ. Передавальна функція ReLU є одним з останніх успіхів в області методів налаштування глибоких нейронних мереж:

$$f(x) = \max(0, x) \ , \qquad (1.2)$$

де x – вхід нейрона.

Сьогодні існує сімейство різних модифікацій ReLU, що вирішують проблеми надійності цієї передавальної функції при проходженні через нейрон великих градієнтів: Leaky ReLU, Parametric ReLU, Randomized ReLU.

1.3.2 Одношарові штучні нейронні мережі

Одношаровою є мережа, що складається з сукупності нейронів, утворення яких формує шар, як зображено на рисунку 1.5. Ліві вершини схеми зображеного нейрону служать для розподілу сигналів, що подаються на вхід. За цими вершинами не закріплено жодних обчислень, тому вони не вважаються шаром і позначені колами, щоб відрізнити їх від обчислювальних нейронів, що позначаються квадратами. Існує сполучення окремою вагою кожного елементу з множини входів X з кожним штучним нейроном. За нейроном закріплена робота подачі зваженої суми входів в мережу. З метою спільності штучні та біологічні мережі мають відсутні з'єднання. Існують з'єднання між виходами і входами елементів в самому шарі. Ваги представлені елементами матриці W. Матриця має n рядків і m стовпців, де n – число входів, а m – число нейронів. Таким чином, обчислення вихідного вектору Y зводиться до матричного множення $Y = X * W$.

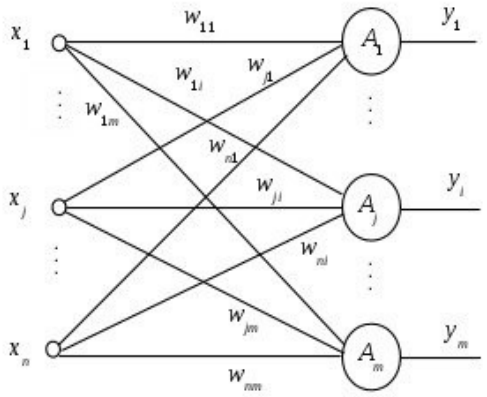

Рисунок 1.5 – Структура одношарової нейронної мережі

1.3.3 Багатошарові штучні нейронні мережі

За сукупністю критеріїв багатошарові архітектури можна розділити на статичні і динамічні. Кожен із класів архітектур нейронних мереж може включати багато підкласів. Нижче будуть приведені основні з них.

До статичних архітектур відносять мережі прямого поширення, у яких реалізований однонапрямлений зв’язок між шарами, відсутні динамічні елементи і зворотний зв’язок, а результат навченої ШНМ однозначно визначається вхідними даними і не залежить від попередніх станів мережі.

Статичні ШНМ прямого поширення:

- Персептрон;
- Саморганізовні карти;
- Когнітрон;
- Неокогнітрон;
- Сучасна згорткова нейронна мережа.

На відміну статичним архітектурам, існують динамічні архітектури ШНМ, реалізуючі рекурентну структуру з використанням зворотних зв’язків, завдяки чому стан мережі в кожен момент часу залежить від попереднього

стану. Рекурентні ШНМ як правило базуються на багатошаровому персептроні.

Динамічні рекурентні ШНМ із зворотними зв’язками:

- нейронна мережа Хопфілда;
- нейронна мережа Коско;
- нейронна мережа Джордана;
- нейронна мережа Елмана.

Персептрон (рис. 1.6) складається з трьох типів елементів: *S*-елементів – елементи що сприймають дані на вході, асоціативних *A*-елементів і реагуючих *R*-елементів на виході. Набір *S*-елементів зв’язаний з *A*-елементом створює асоціацію, і *A*-елемент активується після того, як була досягнута певна кількість сигналів від *S*-елементів. *A*-елемент передає зважений сигнал на сумуючий *R*-елемент і в залежності від того, чи перевищує зважена сума деякий поріг, *R*-елемент видає результат роботи персептрона.

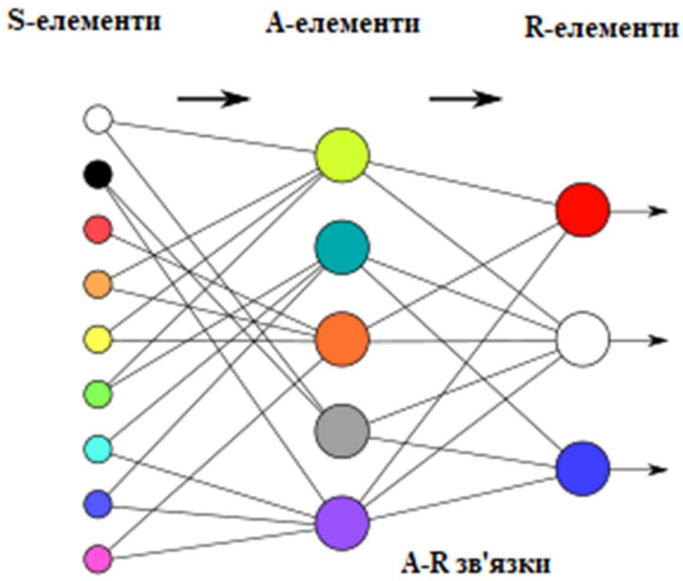

Рисунок 1.6 – Персептрон

Багатошаровий персептрон організується з додатковими прихованими шарами *A*-елементів, розміщеними між *S*-елементами і *R*-елементами.

Навчання елементарного і складного персептрона складається у зміні вагових коефіцієнтів зв'язків *A-R*.

Саморганізовні карти (англ. Self-organizing map – SOM). Саморганізовні карти організуються із шарів Кохонена, що складаються із формальних нейронів – адаптивних лінійних суматорів. Шар Кохонена трансформує найбільший вхідний сигнал в одиничний, а решту – в нуль. Нейронні мережі Кохонена розрізняються за класами вирішуваних ними задач і за способами налаштування вхідних ваг суматорів.

На практиці саморганізовні карти використовуються в задачах кластерного аналізу [23], задачах моделювання і стиснення інформації.

Когнітрон. Когнітрон був розроблений на основі будови біологічної зорової кори, має ієрархічну багатошарову архітектуру. Нейрони між шарами когнітрона зв'язані тільки локально і кожен шар реалізує різні рівні узагальнення: вхідні шари приймають прості образи, у випадку зображень, такі як лінії, крупні однорідні ділянки, їх орієнтацію і локалізацію у просторі вхідних даних, в той час як глибокі шари приймають більш складні абстрактні структури, незалежні від локалізації і інших простих ознак образу.

Когнітрон організується із ієрархічних зв'язаних збуджувальних і гальмівних шарів. Співвідношення збуджувальних і гальмівних шарів на вході нейрона визначає його стан збудження. Існують спрощені моделі когнітрона, що побудовані із одномірних шарів, але спочатку когнітрон конструювався як каскад двомірних шарів. Пресинаптичний простір сигналів визначає виходи попереднього шару або плоскості нейронів, постсинаптичний простір – входи наступного шару або площини. Нейрон когнітрона сприймає не весь постсинаптичний простір сигналів, а лиш його частину, чим і реалізується принцип локальної зв'язності. Область пресинаптичного простору сигналів, утворюючих постсинаптичний простір сигналів, що впливають на стан даного нейрона називається його локальним рецептивним полем (рис. 1.7).

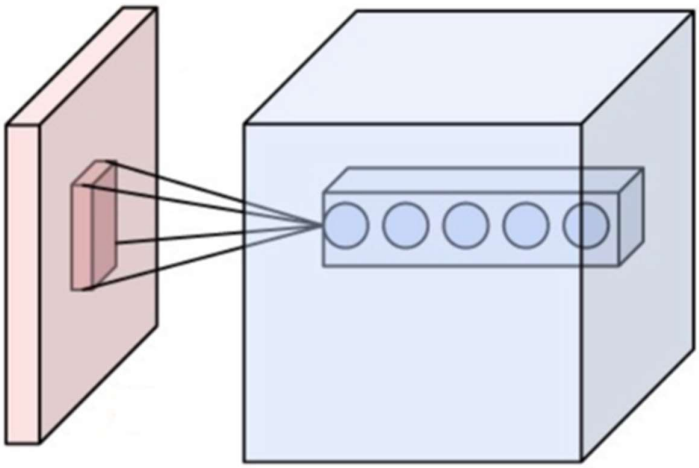

Рисунок 1.7 – Локальне рецептивне поле

Рецептивні поля близьких постсинаптичних нейронів, що називаються зонами конкуренції, перекриваються, і цьому активність даного пресинаптичного нейрона позначається на все більш поширені області постсинаптичних нейронів наступних шарів ієрархії. Розміри зон конкуренції визначають кількість ознак, що сприймаються у просторовій області.

Когнітрон навчається на основі принципу самоорганізації, в класичному випадку реалізується алгоритм навчання без вчителя.

Неокогнітрон. Неокогнітрон є прямим розвитком ідей, що лежать в основі когнітрона і точніше моделює структуру зорової кори головного мозку і є класифікатором, здатним на робастне розпізнавання образів. Кожний шар неокогнітрона складається із площини простих *S*-нейронів і площини складних *C*-нейронів, що також організуються у локальну зв'язність. Локальне рецептивне поле площини *S*-нейронів наступного шару формується пресинаптичними сигналами площини *C*-нейронів попереднього шару. Локальні ознаки образу сприймаються *S*-нейронами, а спотворення локальних ознак компенсується *C*-нейронами. В результаті цього процесу кожен шар після вхідного приймає на вхід все більш узагальнену картину, що була створена *C*-нейронами попередніх шарів. Із кожним рівнем глибини первинні прості ознаки детектуються у все більш складних поєднаннях.

Площину *S*-нейронів можна розглядати як один нейрон, ваги якого визначають ядро згортки, що застосовується до попереднього шару у всіх можливих позиціях. Всі *S*-нейрони реагують на образ, що відповідає ядру згортки в їх рецептивному полі, що дає можливість детектувати його інваріантно до його локалізації.

Навчання неокогнітрона проводиться також без вчителя. Як і у випадку із когнітроном, воно відповідає процедурі виділення набора факторів подібно методу головних компонент і відбувається природним шляхом у процесі самоорганізації мережі.

Неокогнітрон часто застосовується для розпізнавання рукописного тексту, номерів автомобілів і домів.

Сучасні глибокі згорткові нейронні мережі базуються на ідеях, що лежать в основі неокогнітрона і сьогодні застосовуються для вирішення широкого спектру задач: від промислових, корпоративних і дослідних до повсякденних побутових, включно задачі, що вирішуються мобільними пристроями.

Рекурентна нейронна мережа. Рекурентні нейронні мережі організуються в різних архітектурах із різними видами зворотного зв’язку. Наявність зворотних зв’язків дозволяє реалізувати систему, що має асоціативну пам’ять що дозволяє відтворювати послідовності реакцій на певний стимул. Найбільш відомими архітектурами рекурентних нейронних мереж є мережі Хопфілда, мережі Коско, мережі Джордана і мережі Елмана. Широкі варіативні можливості принципової архітектури і складності аналізу структур, що реалізують механізми асоціативної пам’яті призводять до того, що більшість можливостей рекурентних ШНМ на даний момент вивчені достатньо погано.

1.4 Постановка задачі дослідження

Використання методів класичних розпізнавання об’єктів та нейронних мереж є актуальним завданням при вирішенні задачі оцінки кількості людей у відеоряді. Інтелектуальний аналіз відеоданих вважається найбільш складним типом обробки мультимедійних даних, хочу найчастіше він обмежується розпізнаванням послідовності зображень з подальшою сегментацією і зіставленням. [24]

Об’єктом дослідження є сфера громадського спостереження, а саме процес оцінки кількості учасників мирних зібрань.

Метою дослідження є реалізація та аналіз методів оцінки кількості людей у відеоряді в цілях моніторингу мирних зібрань.

В процесі виконання поставленої задачі необхідно:

– розглянути основні принципи та існуючі методи рішення задач машинного навчання, розпізнавання образів і комп’ютерного зору, провести аналіз доступних існуючих систем;

– реалізувати систему розпізнавання мовою високого рівня із використанням декількох методів;

– провести випробування запропонованої системи;

– порівняти обрані методи на реальних даних та зробити висновки щодо використання кожного з методів.

2 ОГЛЯД ВИКОРИСТАНИХ МЕТОДІВ

2.1 Гістограми напрямлених градієнтів

Гістограми напрямлених градієнтів є алгоритмом ковзаючого вікна [25]. Це означає що для будь-якого зображення вікно виявлення 64 пікселів по горизонталі та 128 по вертикалі переміщується за певним шаблоном у всіх місцях і масштабах та дескриптором обчислюється для цього вікна. Вікно сканує вхідне зображення з кроком 32 пікселів по горизонталі та 64 пікселів по вертикалі, як показано на рисунку 2.1.

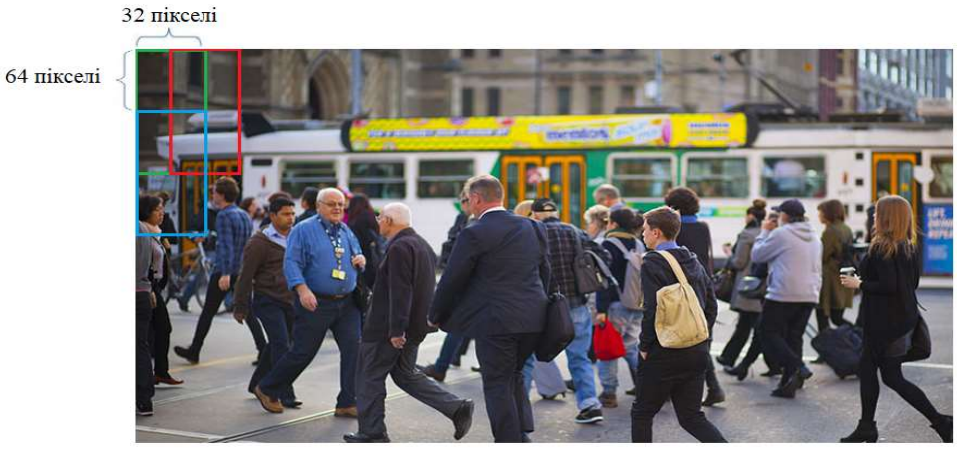

Рисунок 2.1 – Ковзаюче вікно поверх зображення

Для кожного вікна використовується попередньо навчений класифікатор, що присвоює дескриптору відповідний бал. Використаний класифікатор є лінійним класифікатором SVM, а дескриптор базується на гістограмі градієнтних орієнтацій. Часто може виникнути ситуація коли поруч виявляється один і той же об’єкт і вони, як правило, об’єднуються за допомогою підходів немаксимального придушення, щоб отримати рамки з великим значенням впевненості.

На рисунку 2.2 зображено схему роботи алгоритму гістограми напрямлених градієнтів.

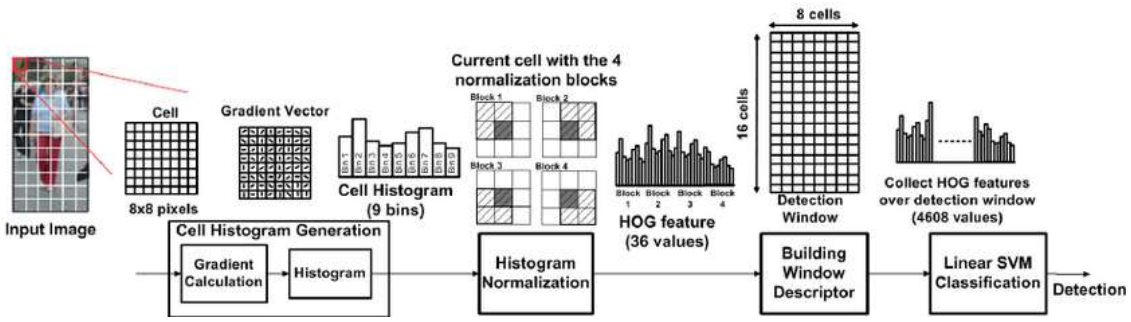

Рисунок 2.2 – Схема роботи гістограми напрямлених градієнтів

Спочатку до зображення по краях додаються зони білого кольору і застосовується гамма-нормалізація. Додавання пустих зон допомагає алгоритм справлятися з випадками, коли людина не повністю міститься на зображенні. Доведено, що гамма-нормалізація покращує ефективність роботи виявлення людей, але це може знизити продуктивність для інших класів об’єктів.

2.1.1 Градієнтний вектор пікселів

Градієнт зображення – це спрямована зміна інтенсивності або кольору зображення. Градієнт зображення є одною з основних особливостей в обробці зображень. Математично, градієнт функції із двома змінними (тут інтенсивність зображення функція) у кожній точці зображення є двовимірним вектором із компонентами, заданими символом похідної в горизонтальному та вертикальному напрямках. У кожній точці зображення вектор градієнта вказує у напрямку максимально можливого збільшення інтенсивності, а довжина вектора градієнта відповідає швидкості зміни в цьому напрямку (рис. 2.3).

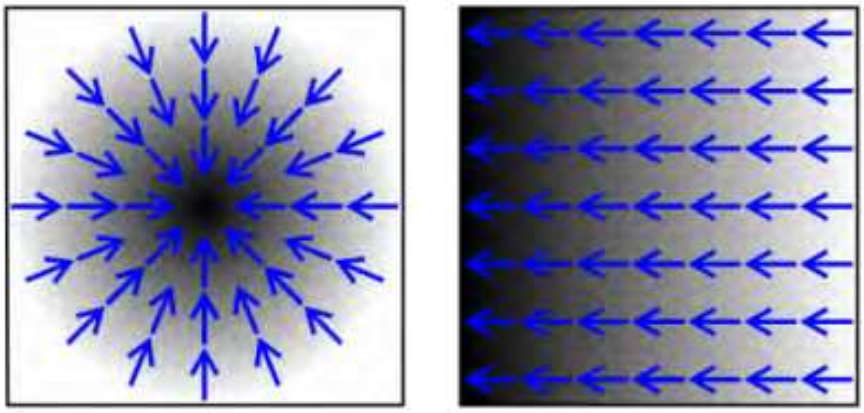

Рисунок 2.3 – Значення та напрямок градієнта

На рисунку 2.3 величини градієнта представлені чорно-білими, де чорний колір представляє більше значення, а відповідний кут – синіми стрілками. Напрямок та величина градієнта отримуються для кожного пікселя із попередньо обробленого зображення. На зображеннях RGB обирається колір з максимальною величиною. Градієнт кожного пікселя обчислюється за згорткою з ядром [-1 0 1].

$$G_x = K_x * I, \quad K_x = [-1, 0, 1]\,, \tag{2.1}$$

$$G_y = K_y * I, \quad K_y = [-1, 0, 1]^T\,, \tag{2.2}$$

$$|G(x,y)| = \sqrt{G_x(x,y)^2 + G_y(x,y)^2}\,, \tag{2.3}$$

де I – значення пікселю.

Напрямок градієнту розраховується за наступною формулою:

$$\tan\big((x,y)\big) = \frac{G_x(x,y)}{G_y(x,y)}\,. \tag{2.4}$$

На рисунку 2.4 вказано приклад застосування функції градієнту до зображення.

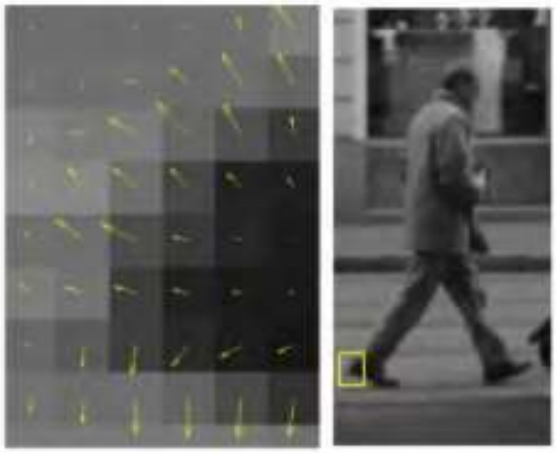

Рисунок 2.4 – Результат застосування функції градієнту до зображення

2.1.2 Виділення гістограми

Гістограма – це точне відображення розподілу числових даних. Вона є оцінкою розподілу ймовірностей неперервної змінної. Гістограми дають приблизну щільність базового розподілу даних.

Вікно виявлення поділено на 8×16 прямокутних областей, що називаються чарунками. Кожен чарунок складається з 8×8 пікселів, які потім дискретизуються в 9 кутових бінів відповідно до їх градієнтної орієнтації [26]. Бін кожного пікселя обчислюється як:

$$bin = (\arctan G_x/G_y) \div 20^{\circ}. \quad (2.5)$$

Кожен піксель вносить зважений голос за відповідний кутовий відсік голосування – це функція величини градієнта в пікселі. Таким чином інформація стискається до 9-мірного простору на клітинку. Кутові біни гістограм є рівномірно розподілені поміж 180°.

Крім того голоси інтерполюються трилінійно між сусідніми центрами бінів як в орієнтації, так і в положенні.

Нехай $h(x)$ позначає значення гістограми для біна з центром у x. Нехай w у точці $x = [x, y, z]$ – вага, яку потрібно інтерполювати. Нехай x_1 і x_2 – це два кутові вектори куба гістограми, що містить x, де в кожній складовій $x_1 \leq x < x_2$. Припустимо, що смуга пропускання гістограми вздовж осей x, y та z дорівнює $b = [b_x, b_y, b_z]$. Трилінійна інтерполяція розподіляє вагу w до 8 сусідніх центрів бінів, як показано нижче:

$$\mathrm{h}(x_1, y_1, z_1) \leftarrow \mathrm{h}(x_1, y_1, z_1) + w\left(1 - \frac{x - x_1}{b_x}\right)\left(1 - \frac{y - y_1}{b_y}\right)\left(1 - \frac{z - z_1}{b_z}\right), \quad (2.6)$$

$$\mathrm{h}(x_1, y_1, z_2) \leftarrow \mathrm{h}(x_1, y_1, z_2) + w\left(1 - \frac{x - x_1}{b_x}\right)\left(1 - \frac{y - y_1}{b_y}\right)\left(\frac{z - z_1}{b_z}\right), \quad (2.7)$$

$$\mathrm{h}(x_1, y_2, z_1) \leftarrow \mathrm{h}(x_1, y_2, z_1) + w\left(1 - \frac{x - x_1}{b_x}\right)\left(\frac{y - y_1}{b_y}\right)\left(1 - \frac{z - z_1}{b_z}\right), \quad (2.8)$$

$$\mathrm{h}(x_2, y_1, z_1) \leftarrow \mathrm{h}(x_2, y_1, z_1) + w\left(\frac{x - x_1}{b_x}\right)\left(\frac{y - y_1}{b_y}\right)\left(1 - \frac{z - z_1}{b_z}\right), \quad (2.9)$$

$$\mathrm{h}(x_1, y_2, z_2) \leftarrow \mathrm{h}(x_1, y_2, z_2) + w\left(1 - \frac{x - x_1}{b_x}\right)\left(\frac{y - y_1}{b_y}\right)\left(\frac{z - z_1}{b_z}\right), \quad (2.10)$$

$$\mathrm{h}(x_2, y_1, z_2) \leftarrow \mathrm{h}(x_2, y_1, z_2) + w\left(\frac{x - x_1}{b_x}\right)\left(1 - \frac{y - y_1}{b_y}\right)\left(\frac{z - z_1}{b_z}\right), \quad (2.11)$$

$$\mathrm{h}(x_2, y_2, z_1) \leftarrow \mathrm{h}(x_2, y_2, z_1) + w\left(\frac{x - x_1}{b_x}\right)\left(\frac{y - y_1}{b_y}\right)\left(1 - \frac{z - z_1}{b_z}\right), \quad (2.12)$$

$$\mathrm{h}(x_2, y_2, z_2) \leftarrow \mathrm{h}(x_1, y_1, z_1) + w \quad \left(\frac{x - x_1}{b_x}\right) \quad \left(\frac{y - y_1}{b_y}\right) \quad \left(\frac{z - z_1}{b_z}\right). \quad (2.13)$$

Дескриптор гістограми напрямлених градієнтів, є об'єднанням усіх блоків, як зображено на рисунку 2.5.

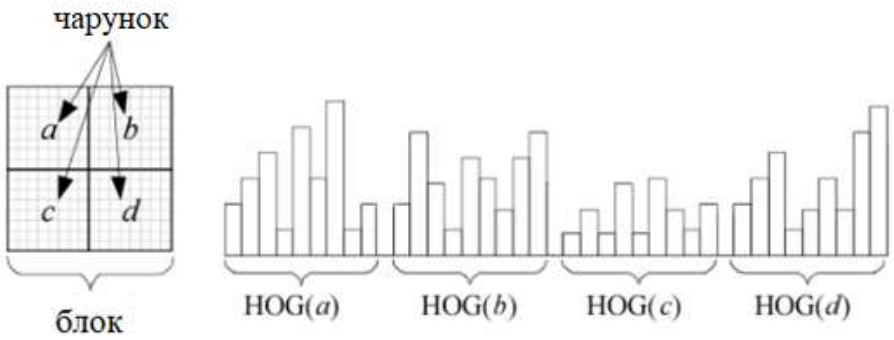

Рисунок 2.5 – Дескриптор гістограми напрямлених градієнтів

Насправді блоки пересікаються таким чином, що кожна відповідь кожної клітини з'являється у кінцевому векторі ознак кілька разів. Крок блоку становить 8 пікселів (1 чарунок), що призводить до чотирикратного охоплення кожного чарунка. Підсумовуючи, кожне вікно виявлення представлене блоками 7×15. Кількість горизонтальних блоків обчислюється як:

$$\frac{\text{Ширина вікна}}{\text{Крок блоку}} - 1, \quad (2.14)$$

а кількість вертикальних блоків як:

$$\frac{\text{Висота вікна}}{\text{Крок блоку}} - 1. \quad (2.15)$$

Блок складається з 2×2 чарунків, а кожен чарунок є гістограмою з 9 бінів, що в сумі дає $(7 \times 15)(2 \times 2) \times 9 = 3780$ ознак.

2.1.3 Нормалізація гістограми

Наступним кроком алгоритму HOG є нормалізація дескрипторів гістограм до виконання класифікації методом опорних векторів (SVM – Support Vector Machines). Нормалізацією називається приведення значень, виміряних на різних масштабах до одного масштабу. Потужності градієнтів можуть змінюватися в широкому діапазоні за рахунок змін тіней, місцевих коливань освітленості та контрастності переднього та заднього планів. Тому локальна нормалізація контрасту має важливе значення. Для цього групи по 2×2 сусідніх чарунків розглядаються як просторові області, які називаються блоками. Кожен блок представлений конкатенацією відповідних чотирьох гістограм чарунків, в результаті чого виходить 36-мірний вектор ознак, який нормується до одиниці довжини, використовуючи норму *L2*.

2.1.4 Класифікація

У машинному навчанні SVM – це контрольовані моделі навчання, які використовуються для класифікації та регресії. Модель SVM – це подання прикладів як точок у просторі, зіставленому так, щоб приклади окремих класів були максимально розділені. Потім нові приклади відображаються в тому самому просторі і, залежно від того, в якій частині простори вони знаходяться, відбувається віднесення до певної категорії (рис. 2.6).

Створений дескриптор HOG використовується для класифікації вікна виявлення до одного із заздалегідь визначених класів. Для цього етапу класифікації використовується лінійну машину опорних векторів (SVM) [27], яка є точною та ефективною з точки зору продуктивності.

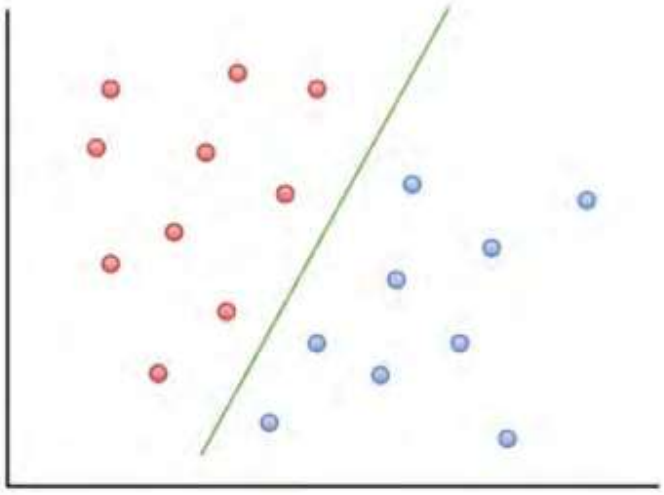

Рисунок 2.6 – Лінійно розділені дані, що відносяться до одного з класів: червоний та синій

2.2 Згорткові нейронні мережі

Згорткові нейронні мережі (CNN – Convolutional Neural Networks) активно використовуються для ефективного розпізнавання зображень. Свою назву вони отримали від операції, що називається згортка (англ. convolution) і часто використовується для оброблення зображень. В даному типі нейромережі використовується три види шарів: згортковий шар (convolution), шар субдискретизації (subsampling, pooling) та повнозв’язний шар (fully-connected) [28]. Послідовність цих шарів визначає якість роботи CNN.

В згорткових нейронних мережах, на відміну від мережі прямого поширення, де кожен вхідний нейрон з’єднується з вихідним нейроном в наступному шарі, для отримання вихідних значень застосовуються згортки над кожним вхідним шаром. В операції згортки використовуються матриця ваг невеликого розміру, яка проходить по всьому поточному шару, формуючи після кожного зсуву сигнал активації для нейрона наступного шару з аналогічною позицією. Ця матриця називається ядром згортки; вона використовується для різних нейронів вихідного шару.

Багатошарові нейронні мережі (рис. 2.7) отримують вхідні дані, після чого трансформують інформацію, проводячи її через ряд прихованих шарів. Кожен прихований шар складається з множини нейронів, де кожний нейрон має сильний зв'язок з усіма нейронами в попередньому шарі і де нейрони в якості одного шару повністю незалежні один від одного і не мають спільних з'єднань. Останній повнозв'язний шар називається вихідним шаром, і в налаштуваннях класифікації він демонструє число класів.

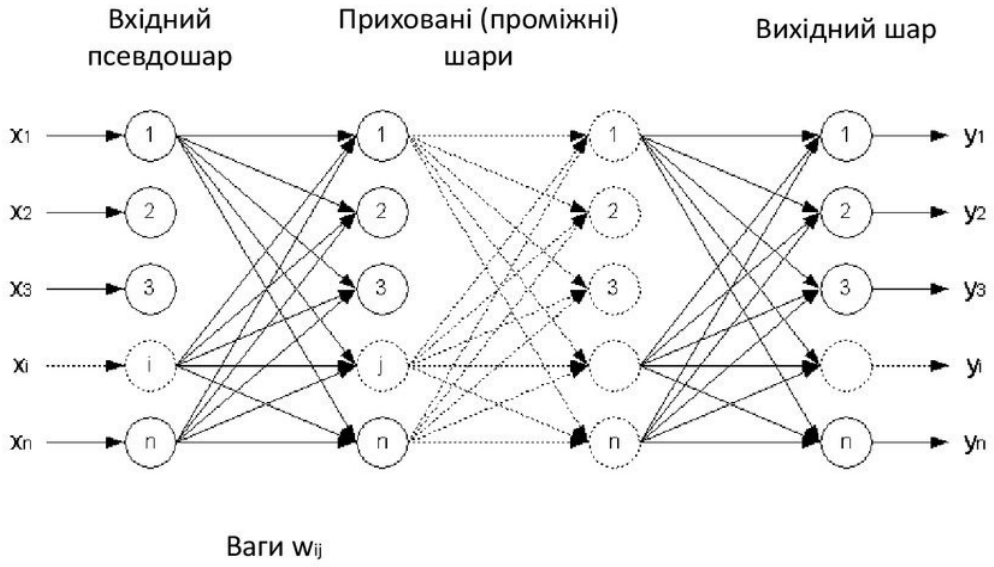

Рисунок 2.7 – Багатошарова нейронна мережа

Звичайні багатошарові нейронні мережі погано масштабуються у випадку з зображеннями великих розмірів [29]. Взявши, наприклад, зображення розміру $[32 \times 32 \times 3]$ (32 – ширина, 32 – висота, 3 – канали кольорів), один повністю підключений нейрон в першому прихованому шарі звичайної нейронної мережі має вагу $32 * 32 * 3 = 3\,072$. Така архітектура погано масштабується для великих зображень. Зображення більшим обсягом, наприклад, $[200 \times 200 \times 3]$, призведе до того, що повністю підключений нейрон буде важити 120 000. Тобто, недоліком повнозв'язності є величезна кількість параметрів, що може швидко привести до перенавчання мережі.

Згорткові нейронні мережі отримують на вхід зображення і вони обмежують побудову мережі більш розумним шляхом. Шари CNN складаються з нейронів, розташованих в 3-х вимірах: ширині, висоті і глибині (рис. 2.8). Нейрони будуть підключені тільки до невеликої області шару. Крім того, результуючий вихідний шар для системи складатиме $[1 \times 1 \times N]$, де N – кількість класів, що система розпізнає, оскільки до кінця побудови CNN зображення перетвориться в єдиний вектор оцінок класу, розташованих за виміром глибини.

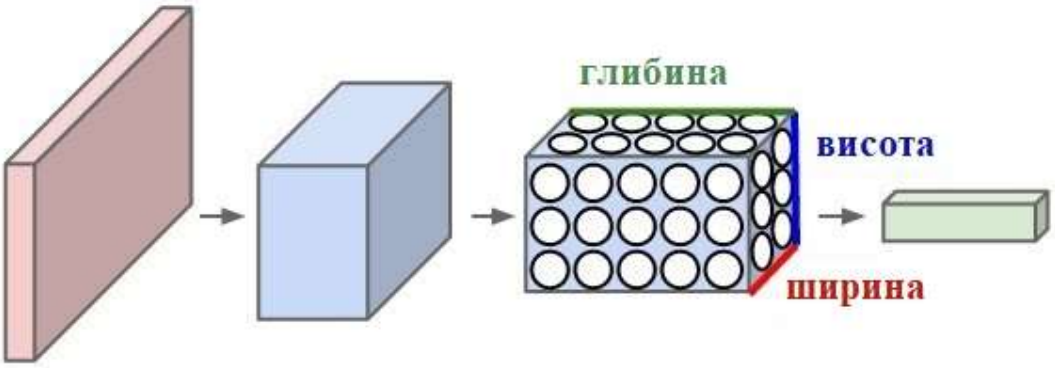

Рисунок 2.8 – Згорткова нейронна мережа

Основу згорткових нейронних мереж складають шари, що характеризуються простим набором задач: вони перетворюють вхідні дані у вигляді 3D-об’єму у вихідний 3D-об’єм з деякою диференційованою функцією, яка може мати або не мати параметри.

2.2.1 Шари згортки

Основним шаром у згорткових нейронних межах вважається шар згортки, що виконує основну роботу даної мережі. Параметром цього шару виступає набір фільтрів для навчання. Кожен фільтр має невеликі просторові розміри (ширину і висоту), але проходить по всій глибині вхідного об’єму.

Наприклад, стандартний фільтр першого шару згорткової нейронної мережі може бути розміром $[5 \times 5 \times 3]$. Проходячи вперед по зображенню виконується ковзання кожного фільтру по ширині і висоті вхідних даних і обчислюється скалярний добуток між матрицею фільтра і входом у будь-яке положення. В результаті складається двомірна активаційна карта, яка надає відгук цього фільтра на кожній просторовій позиції (рис. 2.9). Мережа навчає фільтри, що активуються при виявленні певної візуальної особливості. Такими особливостями можуть бути зона певного кольору, грань певної спрямованості тощо. Після мережа працює з набором фільтрів в кожному шарі згортки і кожен з фільтрів буде формувати двомірну карту ознак. Карти ознак складаються вздовж глибини вхідного об'єму і будуть формувати вихідний об'єм.

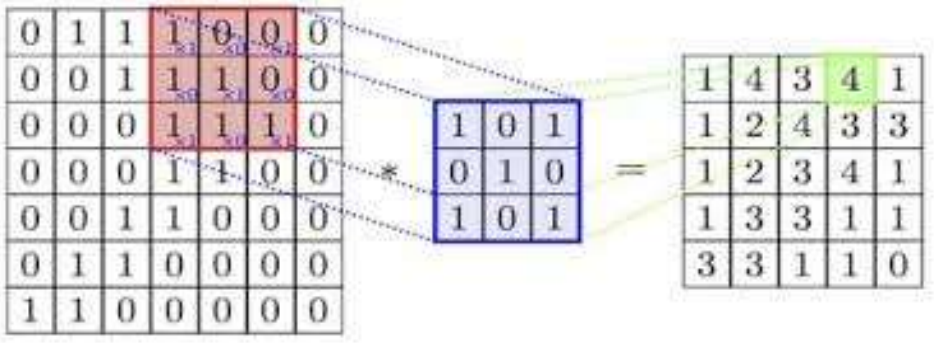

Рисунок 2.9 – Активаційна карта

При роботі з вхідною інформацією високою розмірністю замість зв'язування кожного нейрону із усіма нейронами попереднього шару потрібно застосовувати кожен нейрон тільки до локальної області вхідного об'єму. Просторова протяжність його зв'язку називається рецептивним полем. Ідея рецептивного поля полягає в тому, щоб об'єднати нейрони прихованого шару лише з такими нейронами попереднього шару, які входять в деяку невелику область $n \times n$. Причому для кожного нейрона обирається своя область. Дана область називається локальним рецептивним полем. Задачею рецептивних

полів є виявлення елементарних ознак, таких як кут або край, а їх комбінації дають можливість отримати складніші ознаки.

Нейрони згорткових нейронних мереж як і нейрони звичайних мереж обчислюють скалярний добуток їх ваг і вхідних даних з подальшою нелінійністю, але їх підключення обмежується локальними просторовими вимірами.

2.2.2 Максимальні та усереднені шари підвибірки

В архітектурі згорткових нейронних мереж використовуються шари підвибірки між послідовностями згорткових шарів. Його задачею є поступове зменшення просторових габаритів зображення з метою зменшення кількості параметрів мережі та обчислень загалом, а також контроль перенавчання [30]. Шар підвибірки працює незалежно від зрізу глибини вхідних даних і масштабує обсяг просторово, використовуючи функцію максимуму. Найчастіше використовується шар з фільтрами розміру $[2 \times 2]$ з кроком 2 (рис. 2.10). Подібний шар дає можливість зменшити дискретизацію кожного зрізу глибини входу в 2 рази по ширині і висоті. В цьому випадку кожна операція максимуму буде обирати максимальне значення з чотирьох чисел. При цьому розмір по глибині буде залишатися константним.

У кожному шарі згортки для кожного каналу є свій фільтр, ядро згортки якого обробляє попередній шар за фрагментами. Далі результати застосування різних фільтрів об’єднуються. Результатом цього процесу є шар об’єднання. Операція субдискретизації виконує зменшення розмірності сформованих карт ознак. У архітектурі згорткової нейронної мережі вважається, що інформація про наявність шуканої ознаки є важливішою за знання його точних координат, тому з кількох сусідніх нейронів карти ознак обирається максимальний і приймається за один нейрон карти ознак меншого розміру. Ця операція дає

можливість прискорити подальші обчислення і зробити мережу гнучкою до масштабу вхідного зображення.

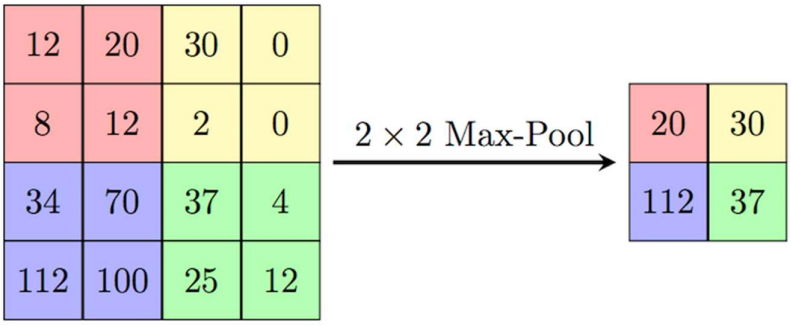

Рисунок 2.10 – Операція підвибірки за максимумом

2.2.3 Техніка dropout

Одною з головних проблем використання глибоких нейронних мереж є перенавчання (англ. overfitting). Перенавчання проявляється у тому, що модель може розпізнавати лише приклади із навчальної вибірки, адаптуючись до навчальних прикладів, замість того щоб вчитися класифікувати приклади, які не брали участі в навчанні, втрачаючи здатність до узагальнення.

Техніка dropout – метод, який використовується для запобігання перенавчання (рис. 2.11). Виключені нейрони не вносять свій внесок в процес навчання ні на одному з етапів алгоритму зворотного поширення помилки (англ. backpropagation), який часто використовується для навчання мереж, тому виключення хоча б одного з нейронів рівносильно ніби навчанню нової нейронної мережі.

Головна ідея техніки dropout – навчання сукупності декількох нейромереж і усереднення отриманих результатів.

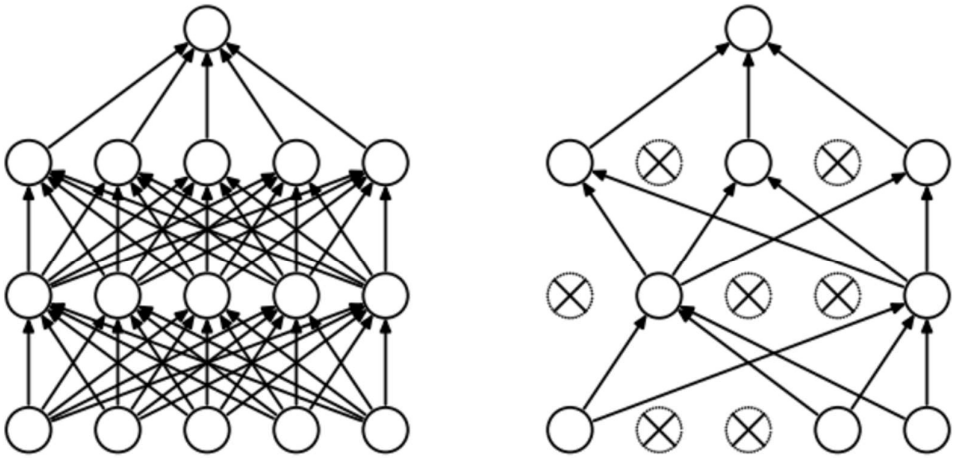

Рисунок 2.11 – Техніка dropout:
(а) нейронна мережа до застосування dropout,
(б) та сама мережа після застосування техніки dropout.

Імовірність виключення кожного нейрона однакова. Застосування техніки dropout на етапі навчання можна представити як змінену функцію активації:

$$out = D * a(h)\,, \tag{2.16}$$

де $a(h)$ – функція активації;

символ $*$ – добуток Адамара.

2.2.4 Техніка batch-normalization

У багатошарових мережах дані, отримані шаром з попереднього шару можуть бути представлені у вигляді будь-якого тензору, координати якого попередньо визначені. Це приводить до того, що параметри в подальших шарах мережі гіршають і виникає необхідність обрати дуже низьку швидкість

навчання, щоб відрегулювати той факт, що функції введення можуть різко змінюватися з часом.

Стає доречним між усіма шарами встановити обрахунки, які б оптимально нормалізували виходи попереднього шару і знижували б тиск на наступний шар, що значно покращило б його роботу в мережі. Спосіб вирішення цієї проблеми полягає в нормалізації міні-партій (англ. batches).

На сьогоднішній день є дві популярні техніки нормалізації: local response normalization та batch-normalization. Обидві використовуються для запобігання зниження швидкості навчання параметрів мережі. Локальна нормалізація відповідності є самостійним шаром мережі. Дана операція нормалізує кожне значення вхідної матриці по каналу.

Пакетна нормалізація застосовує стандартну нормалізацію до отриманих значень, а потім лінійно їх трансформує:

$$y = \frac{x - \mu}{\sqrt{\sigma^2 - \varepsilon}} * \gamma + \beta\,, \tag{2.17}$$

де ε, γ та β є параметрами, що підлягають налаштуванню;

σ та μ, дисперсія та середнє значення залежать від того на якому етапі знаходиться мережа.

Якщо відбувається навчання мережі, то ці значення беруться від значень, отриманих тільки в поточний момент. Під час тестування мережі значення беруться з усієї зібраної статистики.

2.2.5 Класифікатор

В згорткових мережах в повнозв'язних шарах нейрони мають зв'язок з усіма функціями активації з попереднього шару. Активації шарів класифікації можуть бути обчислені за допомогою множення матриць, що супроводжується зміщенням.

Шар класифікації відрізняється від шару згортки тим, що нейрони шару згортки з'єднані лише з локальною областю на вході і що нейрони цього шару можуть спільно використовувати параметри. Однак нейрони в обох шарах підраховують скалярний добуток, тому їх функціональна форма ідентична.

Класифікатор повинен запам'ятовувати всі навчальні дані та зберігати їх для подальших порівнянь із даними з тестового запуску. Це використовує дуже багато ресурсів, оскільки набори даних можуть бути розміром у гігабайтах.

Лінійний класифікатор (рис. 2.12) дає оцінку класу як зважену суму всіх значень пікселів у трьох його кольорових каналах. Залежно від того, які значення встановлено для цих ваг, функція класифікатору має здатність позитивно або негативно оцінювати певні кольори у певних положеннях зображення.

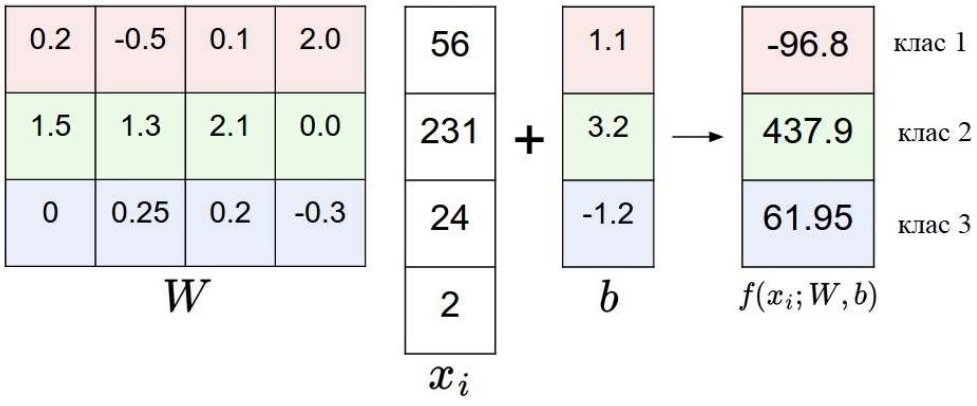

Рисунок 2.12 – Лінійний класифікатор

2.2.6 Softmax

Популярним вибором класифікатору є класифікатор softmax. Він має зовсім іншу функцію втрат та в його основі лежить нормовану експоненційну функцію. Цей класифікатор є узагальненим класифікатором двійкової логістичної регресії. На відміну від методу опорних векторів, який обчислює результати вектору $f(x_i, W)$ як оцінки для кожного класу, класифікатор softmax керований трохи більш інтуїтивним підходом, а також має імовірнісну інтерпретацію. У цьому класифікаторі функція втрат L_i відображення $f(x_i, W) = Wx_i$ залишається незмінною, підключається функція крос-ентропії $L(X, Y)$:

$$L_i = -\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right), \tag{2.18}$$

$$L(X, Y) = -\frac{1}{n}\sum_{n}^{i=1} y^i \ln a\left(x^i\right) + \left(1 - y^i\right) \ln\left(1 - a\left(x^i\right)\right), \tag{2.19}$$

де $X = \{x(1), \dots, x(n)\}$ – набір вхідних прикладів для навчання у вхідному наборі;

$Y = \{y(1), \dots, y(n)\}$ – набір міток для вхідного набору;

функція $a(x)$ – вихідні значення нейронної мережі з вхідним значенням x.

2.3 Методи навчання нейронних мереж

Під навчанням нейронної мережі мається на увазі налаштування її архітектури, ваги нейронів мережі, їх зв’язків. Всі ці параметри впливають на ефективність виконання мережею поставленої задачі. Зазвичай мережа повинна налаштувати свої параметри за поданими навчальними прикладами.

Властивість мереж навчатися за поданими прикладами робить їх більш досконалими у порівнянні з системами, що працюють за прописаними заздалегідь правилами. Серед методів навчання мереж можна виділити два класи: детермінований та стохастичний.

Методи детермінованого класу інтерактивно корегують параметри мережі, спираються на стан поточних параметрів, величини входів, фактичних та бажаних виходів. Прикладом такого методу є метод зворотного поширення помилки.

Методи стохастичного класу змінюють параметри мережі випадковим чином. При цьому збереженими залишаються лише ті зміни, що призвели до покращення. В якості прикладу стохастичного методу можна привести генетичний алгоритм.

Необхідно зауважити, що стохастичні методи навчання можуть потрапити до пастки локального мінімуму (рис. 2.13)

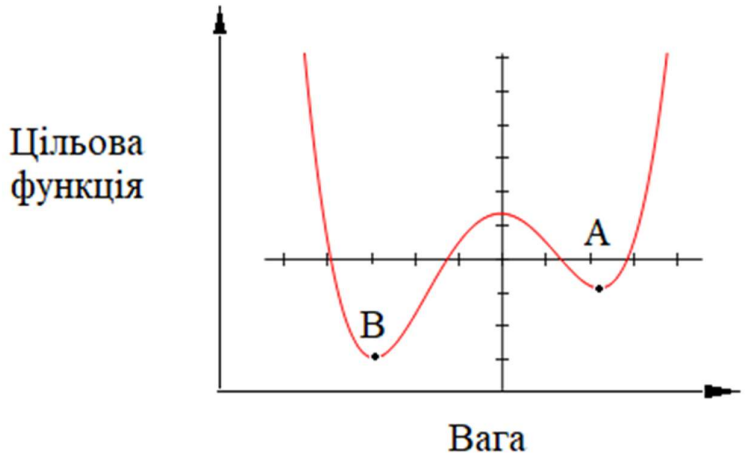

Рисунок 2.13 – Пастка локального мінімуму

Розглянемо зображення рисунку. Нехай початкове значення помилки рівно чи близьке до значення у точці *A*. Якщо випадкові кроки корегування дуже малі, то будь-які відхилення від точки *A* збільшать помилку та не будуть братися до уваги. Таким чином найменше значення в точці *B* ніколи не буде

знайдено. При дуже великих випадкових кроках корегування параметрів мережі помилка буде піддаватися зміні так різко, що ніколи не закріпиться за одним із мінімумів.

Щоб уникнути проблеми локального мінімуму можна поступово зменшувати випадковий середній розмір кроків корегування. Коли середній розмір кроків завеликий, значення помилки буде приймати усі значення з рівною імовірністю. При плавному зменшенні розміру кроків буде дотримано умові при якому значення помилки буде на деякий час затримуватись в точці B. Коли значення кроку буде ще більш зменшено, значення помилки буде зупинятися на короткий проміжок часу і в точці *A*, і в точці *B*. При зменшенні кроку безперервно у кінці буде досягнуто величину кроку, що буде достатньою для подолання локального мінімуму *A*, але не локального мінімуму *B*.

2.3.1 Навчання з вчителем

Визначення методу навчання з учителем можна сформувати так: задача машинного навчання у реалізації функції виведення на підставі відомих навчальних даних.

Структура навчальних даних складається з набору прикладів для виконання навчання. В даному методі навчання кожен приклад представляє собою пару, що складається з вхідного об’єкта (зазвичай вектору) і бажаного вихідного значення (контрольний сигнал). Необхідно прорахувати значення функції на момент входу даних, потім порівняти значення із значенням очікуваного результату, вирахувати помилку і скорегувати параметри мережі на цю помилку.

Алгоритм навчання з вчителем:

Крок 1. Взяті дані для навчання мережі розділити на дві частини. Першою буде навчальна вибірка, другою – тестова вибірка. Дані можливо розділити на співвідношення 70/30.

Крок 2. Прорахувати значення функції для навчальної вибірки та знайти значення функції помилки.

Крок 3. Провести корегування ваг синапсів у мережі.

Крок 4. Повторювати кроки 2 і 3 до тих пір, поки значення функції помилки не буде мінімальним. Повторювати ці дії можна як для кожного екземпляра навчальної вибірки так і для всієї вибірки в цілому. У першому випадку навчання буде повільніше, але матиме більшу точність. У другому випадку показник точності занепадає, але навчання буде відбуватися швидше.

Крок 5. Перевірити всі значення за тестовою вибіркою, щоб зрозуміти наскільки добре система змогла узагальнити дані.

Якщо нейронна мережа навчається з використанням заздалегідь відомих правильних відповідей, то такий алгоритм навчання називається навчанням з учителем.

2.3.2 Навчання без вчителя

Метод навчання без учителя (метод неконтрольованого навчання) має змогу знайти структуру або відносини між різними входами. Головна відмінність від методу навчання з вчителем є те, що в наявності є тільки вхідні дані. Алгоритм навчання без вчителя застосовується тоді, коли відомі тільки вхідні дані. На основі їх мережа навчається видавати найкращі вихідні результати. Поняття «найкращих результатів» визначається самим алгоритмом навчання. Зазвичай алгоритм підлаштовує параметри так, щоб мережа видавала однакові результати для достатньо близьких вхідних значень.

Найбільш важливим неконтрольованим навчанням є кластеризація, яка створює різні кластери введення і зможе вносити будь-які нові дані до відповідного кластеру.

Хоча даний метод і часто використовується у прикладних задачах, він часто піддається критиці вчених через свою біологічну схильність. Важко уявити, що у мозку існує механізм порівняння отриманих результатів з бажаними.

2.3.3 Алгоритм зворотного поширення помилки

Згорткові мережі за своїм типом поширення активаційного сигналу між нейронами є прямими, тому застосування алгоритму зворотного поширення помилки є доречним до таких мереж. Даний алгоритм також називають алгоритм градієнтного спуску через те, що стратегія підбору такого важливого параметру, як вага для кожного нейрону багатошарової мережі базується на градієнтному методі.

Безперервна цільова функція як показник успішності мережі в загальному випадку визначається як квадратична різниця суми між фактичним результатом і очікуваним вихідним значенням.

По суті задача навчання ШНМ зводиться до находження певної функціональної залежності $O = F(I)$, де I – вхідний, а O – вихідний вектори. У загальному випадку така задача при обмеженому наборі вхідних даних має нескінченну множину рішень. Для обмеження простору пошуку при навчанні ставиться задача мінімізації цільової функції помилки ШНМ, що находиться, наприклад, за методом найменших квадратів:

$$E = \frac{1}{2}\sum(t_{k_n} - o_{k_n})^2 , \quad (2.20)$$

де o_{k_n} – значення k-го виходу нейронної мережі;

t_{k_n} – цільове значення виходу;

n – номер останнього шару.

Навчання нейронної мережі проводиться методом градієнтного спуску, тобто на кожній ітерації зміна ваг проводиться наступною формулою:

$$w_{ij_k} = w_{ij_k} + \theta \frac{\Delta E}{\Delta w_{ij_k}} , \quad (2.21)$$

де θ – параметр, що визначає швидкість навчання;

$k = 1..n$ – номер шару.

Алгоритм навчання:

Крок 1. Ініціалізація ваг (ваги усіх синапсів ініціалізуються випадковими невеликими значеннями).

Крок 2. Поки умова припинення роботи алгоритму не виконується, виконуються кроки 2-9.

Крок 3. Кожен вхідний нейрон $(x_i, i = 1, 2, \dots, n)$ відправляє отриманий сигнал x_i усім нейронам у наступному шарі.

Крок 4. Кожен прихований нейрон $(z_j, j = 1, 2, \dots, p)$ сумує вхідні сигнали:

$$z_{in} = v_{0j} + \sum x_i * x_{ij} \quad (2.22)$$

і застосовує функцію активації $z_j = f(z_{in_j})$. Після цього посилає результат всім елементам наступного шару.

Крок 5. Кожен вхідний нейрон ($y_k, j = 1, 2, \dots, m$) сумує зважені вхідні сигнали

$$y_{in_k} = w_{0k} + \sum z_j * w_{jk} \tag{2.23}$$

і застосовує функцію активації, обчислюючи вихідний сигнал: $y_k = f(y_{in_k})$.

Крок 6. Кожен вхідний нейрон ($y_k, j = 1, 2, \dots, m$) отримує цільове значення – те значення, що є правильним для даного вхідного сигналу і розраховує помилку:

$$\sigma_k = (t_k - y_k) * f`(y_{in_k}), \tag{2.24}$$

а також розраховує величину, на яку зміниться вага синапсу w_{jk}:

$$\Delta w_{jk} = a * \sigma_k * z_j . \tag{2.25}$$

Окрім цього, обчислює величину коригування зміщення:

$$\Delta w_{0k} = a * \sigma_k \tag{2.26}$$

і посилає σ_kнейронам у попередньому шарі.

Крок 7. Кожен прихований нейрон ($z_j, j = 1, 2, \dots, p$) сумує вхідні помилки (від нейронів у наступному шарі):

$$\sigma_{in_j} = \sum \sigma_k * w_{jk} \tag{2.27}$$

і розраховує величину помилки, помноживши отримане значення на похідну функції активації:

$$\sigma_j = \sigma_{in_j} * f`\left(z_{in_j}\right), \tag{2.28}$$

а також розраховує величину, на яку зміниться вага синапсу v_{ij}:

$$\Delta v_{ij} = a * \sigma_j * x_i . \tag{2.29}$$

Окрім цього, обчислюється величина коригування зміщення $v_{0j} = a * \sigma_j$. Кожен вхідний нейрон $(y_k, j = 1, 2, \dots, m)$ змінює ваги своїх зв'язків із елементом зміщення і прихованими нейронами:

$$w_{jk} = w_{jk} + \Delta w_{jk} . \tag{2.30}$$

Кожен прихований нейрон $(z_j, j = 1, 2, \dots, p)$ змінює ваги своїх синапсів із елементом зміщення і вихідними нейронами:

$$v_{ij} = \Delta v_{ij} . \tag{2.31}$$

Крок 8. Перевірка умови припинення роботи алгоритму.

Умовою припинення роботи алгоритму може бути як досягнення сумарної квадратичної помилки результату на виході мережі попередньо установленого мінімуму, як і виконання певної кількості ітерацій алгоритму.

В процесі навчання циклічно вирішуються однокритеріальні задачі оптимізації. Для можливості реалізації метода передавальна функція нейронів має бути диференційованою. Оскільки метод є модифікацією класичного методу градієнтного спуску, то він може бути розглянутий як градієнтний спуск поверхнею помилки.

Недоліком алгоритму зворотного поширення помилки є те, що він не дозволяє в загальному випадку досягти глобального мінімуму. Тому і постають основні труднощі навчання нейронних мереж, що полягають в

методах виходу з локальних мінімумів. Головними недоліками градієнтного спуску при навчанні є:

– «параліч» мережі. Значення ваг мережі в результаті корекції можуть досягти дуже великих величин. Оскільки помилка, що посилається назад в процесі навчання, пропорційна похідній стискаючій функції, процес навчання може майже зупинитися. Цьому можна запобігти, зменшуючи крок, але процес навчання при цьому буде відбуватися довше;

– розмір кроку. Якщо значення кроку не змінюється і воно досить мале, то метод буде сходитися занадто повільно. Якщо ж крок занадто великий, то може виникнути параліч мережі. Необхідно змінювати значення кроку: збільшувати до тих пір, поки не припиниться поліпшення оцінки в напрямок антиградієнта і зменшувати, якщо оцінка не поліпшується.

2.3.4 Функція втрат

Застосовані у лінійному класифікаторі функції визначення оцінки приналежності об’єкту до певного класу не надають певного контролю за вхідними даними, адже вони параметризовані значенням ваг нейронів. Тому варто встановити контроль над цими вагами нейронів таким чином, щоб прогнозовані оцінки класів відповідали значенню з навчальних вибірок.

Практичне застосування класифікаторів показало, що не завжди оцінка класу, що дійсно відповідає правильній ідентифікації об’єкта є вірною, тому рішенням стан обчислення не позитивного результату розпізнавання, а навпаки – негативного. Для цього і використовується функція втрат або цільова функція (функція вартості). Значення функції буде високим, якщо робота з класифікації навчальних даних завершилася з помилками і низьким, якщо все працюватиме вірно.

Функція втрат (англ. loss function) також називається функцією помилки або функцією вартості – це функція, яка вираховує дійсне значення для складної події, як передбачення приналежності до класу вхідного вектора.

2.4 Існуючі архітектури згорткових нейронних мереж

У даному розділі будуть описані архітектури згорткових нейронних мереж для обробки зображень.

2.4.1 AlexNet

У 2012 році вперше в історії конкурс з класифікації зображень бази ImageNet на 1000 класів з перевагою у точності розпізнавання майже у два рази виграла згорткова нейронна мережа AlexNet (рис. 2.14). Архітектура включала згорткові шари, max-pooling-шари і повнозв’язні шари на виході мережі. Також використовувались методи dropout і локальної нормалізації. Як функція активації використовувалась функція ReLU. Навчання мережі проходило на двох потужних графічних прискорювачах протягом тижня.

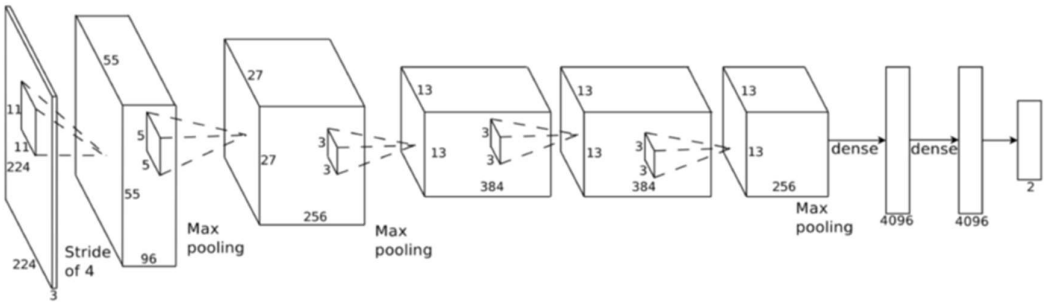

Рисунок 2.14 – Архітектура AlexNet

2.4.2 Inception

Inception-v1 (інша назва – GoogLeNet) – переможець ILSVRC 2014 з top-5 помилкою 6,7%. Автори мережі базувалися на тому, що після кожного шару мережі необхідно зробити вибір – чи буде наступний шар згорткою з фільтром $3 \times 3, 5 \times 5, 1 \times 1$ або шаром субдискретизації. Кожен із таких шарів виконує певну задачу. Фільтр 1×1 виявляє кореляцію між каналами, в той час як фільтри більшого розміру реагують на більш глобальні ознаки, а шар субдискретизації дозволяє зменшити розмір без великих втрат інформації.

Замість того, щоб обирати, який саме шар повинен буде наступним, пропонується використовувати усі шари одразу паралельно один одному, а потім об’єднати отримані результати в один. Щоб запобігти росту числа параметрів перед кожним шаром згортки використовується згортка 1×1, що зменшує число мап ознак. Такий блок шарів назвали модулем Inception (рис. 2.15).

Також в даній архітектурі відмовилися від використання повнозв’язного шару в кінці мережі, використав замість нього шар average pooling, завдяки чому значно зменшилось число параметрів мережі. Таким чином, GoogLeNet, що складається із більш ніж ста базових шарів має майже в 12 разів менше параметрів, аніж AlexNet (близько 7 мільйонів параметрів замість 138 мільйонів).

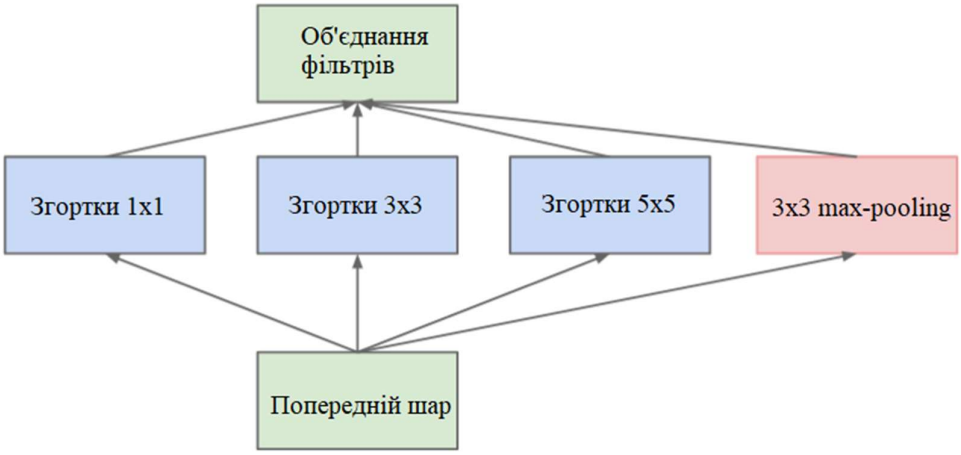

Рисунок 2.15 – Модуль Inception

В наступній ітерації модуля Inception, що називається Inception-v2, автори провели декомпозицію шар з фільтром 5×5 на два шари 3×3. Далі була використана техніка batch normalization, що дала можливість значно підвищити швидкість навчання за рахунок нормалізації розподілення виходів шарів всередині мережі.

Потім автори запропонували Inception-v3. В даній моделі вони запропонували ідею декомпозиції фільтрів розміром $N \times N$ на два послідовних фільтри розмірами $1 \times N$ та $N \times 1$.

2.4.3 ResNet (Residual Network)

Переможцем ILSVRC 2015 з top-5 помилкою в 3,57% став ансамбль із шістьох мереж типу ResNet, розроблений в Microsoft Research.

Автори ResNet помітили, що з підвищенням числа шарів згорткова нейронна мережа може почати деградувати – може зменшитись точність на тестових зображеннях. Так як падає точність і на навчальних зображеннях, то можна зробити висновок, що проблема складається не у перенавчанні мережі.

Було зроблено припущення, що якщо згорткова нейронна мережа досягла своєї межі точності на певному шарі, то всі наступні шари мають перетворитися в тотожні перетворення, але через складності навчання глибоких мереж цього не відбувається. Для вирішення цієї проблеми було запропоновано використання пропускаючих з'єднань (англ. shortcut connections), зображених на рисунку 2.16.

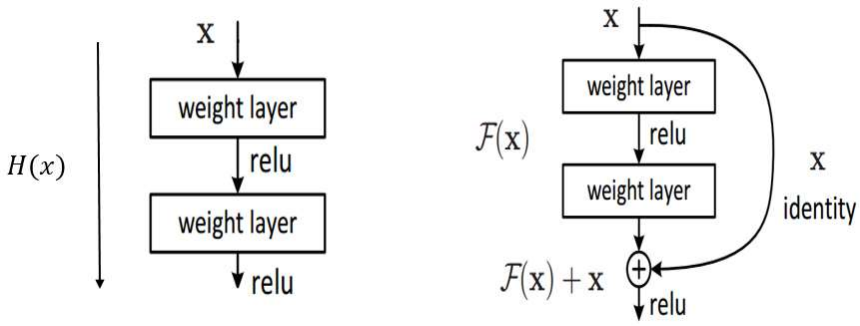

Рисунок 2.16 – Пропускаюче з'єднання

2.4.4 MobileNet

У 2017 році Ендрю Ховард розробив групу невеликих моделей для мобільних застосунків, що називалася MobileNets. Він побудував моделі, що мали різні компроміси між швидкістю і точністю для досягнення різних цілей.

MobileNets архітектура (рис. 2.17) використовує згорткові шари, але замість max-pooling-шарів використовуються згортки з параметром stride рівним 2. Як функція активації використовується функція ReLU.

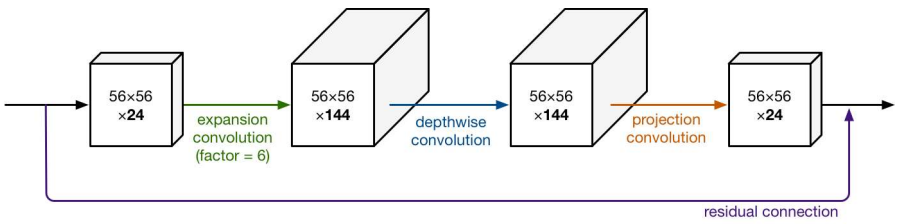

Рисунок 2.17 – Архітектура MobileNet

3 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ

3.1 Установка середовища

Основною операційною системою в даній розробці є система сімейства Linux. В конкретному прикладі був використаний дистрибутив Ubuntu 20.04.

Наступним кроком є установка інтерпретатора мови розробки Python.

3.1.1 Операційна система Ubuntu Linux

З усіх дистрибутивів Linux, вибір припав саме на Ubuntu тому, що дана система користується чималою популярністю серед користувачів.

Одною з позитивних особливостей Ubuntu, що виділяє її від великої кількості інших дистрибутивів Linux в те, що компанія Canonical Ltd надає більш серйозну підтримку своїх продуктів. Також стабільно 2 рази в рік випускаються нові стабільні версії Ubuntu. Через високу популярність Ubuntu існує велика кількість тематичних спільнот, де обговорюються проблеми, пов’язані з цією операційною системою і пропонуються шляхи їх вирішення.

Ubuntu – це одна з багатьох безкоштовних операційних систем на ядрі Linux і одна з найпопулярніших за статистикою. Операційна система має зрозумілий інтерфейс і орієнтована на звичайних користувачів. За замовчуванням в Ubuntu входить набір необхідних прикладних програм для роботи з документами і в Інтернеті. Одним словом, Ubuntu гарна безкоштовна альтернатива операційним системам від Microsoft. Ubuntu включає програмне забезпечення для серверів і робочих станцій. Операційна система Ubuntu підтримує велику кількість видів архітектури, таких як *i386*, *amd64*, *ARM*.

3.1.2 Python

Python є інтерпретованою, спочатку об’єктно-орієнтованою мовою програмування. Він є надзвичайно простий і містить невелику кількість ключових слів, в той же час дуже гнучкий і виразний. Це мова більш високого рівня ніж Pascal, C ++, що досягається, в основному, за рахунок вбудованих високорівневих структур даних (списки, словники, тьюпли).

Безсумнівною перевагою є те, що інтерпретатор Python реалізований практично на всіх платформах і операційних системах. Першою такою мовою був C, проте його типи даних на різних машинах могли займати різну кількість пам’яті і це служило деякою перешкодою при написанні дійсно кросплатформних програм. Python таким недоліком не володіє.

Наступна важлива риса – розширюваність мови, цьому надається велике значення і, як пише сам автор, мова була задуманий саме як розширювана. Це означає, що є можливість вдосконалення мови усіма зацікавленими програмістами. Інтерпретатор написаний на C і вихідний код доступний для будь-яких маніпуляцій. У разі необхідності, можна вставити його в свою програму і використовувати як вбудовану оболонку. Або ж, написавши на C свої доповнення до Python і скомпілювавши програму, отримати «розширений» інтерпретатор з новими можливостями.

Наступна перевага – наявність великого числа приєднуваних до програми модулів, що забезпечують різні додаткові можливості. Такі модулі пишуться на C і на самому Python і можуть бути розроблені усіма досить кваліфікованими програмістами. Як приклад, можна навести такі модулі:

– Numerical Python – розширені математичні можливості, такі як маніпуляції з цілими векторами і матрицями;

– Tkinter – побудова застосунків з використанням графічного інтерфейсу користувача (GUI) на основі широко розповсюдженого на X-Windows Tk-інтерфейсу;

– OpenGL – використання великої бібліотеки графічного моделювання дво- і тривимірних об’єктів.

3.2 Бібліотеки глибокого навчання та роботи з графікою

На сьогоднішній день існує більш десятка бібліотек глибокого навчання, деякі з них мають дуже вузьку специфіку. Найбільш популярними з них є наступні: PyTorch, TensorFlow, Theano, Keras і Caffe.

3.2.1 PyTorch

Бібліотека PyTorch була розроблена як бібліотека для обчислень в наукових цілях і підтримує велику кількість технологій. Бібліотека дозволяє гнучко працювати з нейронними мережами на досить низькому рівні. Для реалізації нейромережі в PyTorch необхідно написати власний цикл навчання, в якому оголошується функція замикання, що обчислює відповідь мережі. Це замикання передається в функцію градієнтного спуску для поновлення ваг мережі. Використання PyTorch не створює серйозних витрат за часом написання програмного коду, крім того, мережі PyTorch мають швидкість класифікації, що перевищує мережі інших фреймворків і найбільш обширну документацію.

PyTorch використовується в основному, щоб навчати моделі швидко і ефективно, тому це вибір великої кількості розробників.

У неї є багато важливих переваг:

– завдяки архітектурі фреймворку, процес створення моделі є досить простим і прозорим;

– режим за замовчуванням «define-by-run» – відсилка до традиційного програмування. Фреймворк підтримує популярні інструменти дебагінгу, такі як pdb, ipdb або дебагер PyCharm;

– він підтримує декларативний паралелізм даних;

– він має багато попередньо навчених моделей і готових модульних частин, які легко комбінувати.

3.2.2 TensorFlow

TensorFlow – це бібліотека програмного забезпечення з відкритим вихідним кодом для задач машинного навчання, розроблена Google. Вона дозволяє створювати і навчати нейронні мережі різної архітектури для виявлення і розпізнавання зразків і пошуку взаємозв'язків. TensorFlow також включає в себе TensorBoard – засіб візуалізації в браузері для оцінки ефективності навчання і мережевих параметрів моделі.

TensorFlow досягає своєї продуктивності завдяки паралелізації задач між центральним і графічними процесорами. Ядро кожної операції реалізовано на C ++ з використанням бібліотек Eigen і cuDNN для кращої продуктивності.

Кожне обчислення в TensorFlow представляється як граф потоку даних, він же граф обчислень. Граф обчислень є моделлю, описує як будуть виконуватися обчислення. Важливо зауважити, що складання графа обчислень і виконання операцій в заданій структурі – два різних процеси. Граф складається з плейсхолдерів (tf.Placeholder), змінних (tf.Variable) і операцій. У ньому виробляється обчислення тензорів – багатовимірних масивів, які можуть бути числом або вектором.

Графи виконуються в сесіях (tf.Session). Існують два типи сесій – звичайні і інтерактивні (tf.InteractiveSession); інтерактивна сесія підходить для

виконання в консолі. Сесія зберігає стан змінних (variables) і черг (queues). Явне створення сесій і графів гарантує належне звільнення ресурсів пам’яті.

У графі кожна вершина має 0 або більше входів і 0 або більше виходів, і являє собою реалізацію операції. Тензори представляють собою ребра графа, а саме масиви довільного розміру (тип масиву вказується під час побудови графа). Особливі вершини, що управляють залежності (control dependencies), також можуть бути в графі: вони вказують, що вихідний вузол для контрольної залежності повинен закінчити виконання до того, як вузол одержувача контрольної залежності почне виконуватися.

Кожна операція має назву і являє собою абстрактне обчислення (наприклад, підсумовування). У операції можуть бути атрибути: наприклад, можливість зробити операцію поліморфною для різних типів тензорів. Ядро – специфічна реалізація операції, яка може бути виконана на певному типі пристрою (центральний або графічний процесор).

Змінна – особливий вид операції, який повертає покажчик на постійно мінливий тензор: така змінна не зникає після одиничного використання графа. Покажчики на подібні тензори передаються численними операціями, які потім змінюють вказаний тензор.

У завданнях машинного навчання, параметри моделі зазвичай зберігають тензори в змінних, які оновлюються на кожному кроці навчання.

Дана робота виконана на єдиному пристрої з використанням CPU.

3.2.3 Theano

Theano існує в першу чергу як розширення мови Python, що дозволяє ефективно обчислювати математичні вирази, що містять багатовимірні масиви. У бібліотеці реалізований базовий набір інструментів для побудови ШНМ. Процес створення моделі і визначення її параметрів вимагає написання об’ємного коду, що включає реалізацію класу моделі, самостійного

визначення її параметрів, реалізацію методів, що визначають функцію помилки, правило обчислення градієнтів, спосіб зміни ваг. Theano є найгнучкішим фреймворком для побудови ШНМ.

3.2.4 Keras

Keras – так звана бібліотека «високого рівня», що працює поверх обчислювальних фреймворків і полегшує написання коду і підвищує його переносимість між різними фреймворками. Дана бібліотека інкапсулює методи і процедури нижчих фреймворків і містить в собі реалізації різних структур для нейронних мереж, полегшуючи їх використання і проектування. Важливою особливістю Keras є можливість роботи з рекурентними і згортковими шарами, а також наявність великої кількості реалізованих додаткових інструментів, необхідних при проектуванні моделей нейронних мереж.

Починаючи з 16 січня 2017 року Keras офіційно була придбана компанією Google і обрана в якості основного високорівневого API для фреймворка Tensorflow.

3.2.5 Caffe

Caffe реалізована на C ++ і має обгортки для Python і Matlab. Топологія ІНV, вихідні дані і спосіб навчання задаються за допомогою конфігураційних protobuf-файлів (технологія і протокол серіалізації даних, що перевершує по ефективності xml і json) в форматі prototxt. Побудова структури мережі виконується з простотою, зручністю і наочністю. Із запропонованих бібліотек є найбільш зручною у використанні, крім того, перевершує інші розглянуті фреймворки в швидкості навчання. Бібліотека Caffe підтримується досить

великою спільнотою розробників і користувачів і на сьогоднішній день є найпоширенішою бібліотекою глибокого навчання широкого кола застосування.

3.2.6 OpenCV

OpenCV – це бібліотека комп'ютерного зору з відкритим вихідним кодом. У 1999 році Гері Бредскі, який працював в корпорації Intel, почав проєкт OpenCV в надії прискорити розвиток комп'ютерного зору і штучного інтелекту, надавши базову інфраструктуру всім працюючим в цій області. Бібліотека написана на C і C ++ і працює на платформах Linux, Windows і Mac OS X. Активно ведеться робота над створенням інтерфейсів до Python, Java, MATLAB та інших мов, а також по перенесенню бібліотеки на платформи Android і iOS для мобільних додатків. Протягом багатьох років розробку OpenCV підтримували корпорації Intel і Google, а особливо компанія Itseez яка виконала основну частину роботи на ранніх етапах розвитку.

При проєктуванні в OpenCV закладалася максимальна обчислювальна ефективність з упором на створення додатків, що працюють у реальному часі. Вона написана на оптимізованому C++ і може користуватися перевагами багатоядерних процесорів.

Одна з цілей OpenCV – надати просту для використання інфраструктуру комп'ютерного зору, яка б дозволила швидко створювати складні додатки. Бібліотека OpenCV налічує понад 500 функцій, що охоплюють багато областей комп'ютерного зору, зокрема контроль якості продукції, медичні зображення, безпеку, призначені для користувача інтерфейси, калібрування камер, стереобачення і робототехніку. В OpenCV включена також універсальна бібліотека машинного навчання (модуль ML). У цій бібліотеці акцент зроблений на статистичному розпізнаванні образів і кластеризації.

3.3 Вибір архітектури нейронної мережі

Для даної роботи була обрана модель VGG-19. Ця мережа характеризується своєю простотою, використовуючи згорткові шари розміру лише 3×3 та max-pooling шари. Два повністю зв'язні шари, кожен з 4096 вузлами, супроводжуються класифікатором softmax.

У VGGNet є два основні недоліки:

– навчання моделі є дуже повільним;

– дуже великий розмір моделі – завдяки своїй глибині та кількості повністю підключених вузлів VGG перевищує 533 МБ для VGG16 та 574 МБ для VGG19.

3.4 Навчання моделі нейронної мережі

Відомо, що згорткові нейронні мережі потребують великого набору даних і ресурсів для ефективного навчання. Наприклад, ImageNet модель була навчена на 1,2 мільйонах зображень протягом 2-3 тижнів використовуючи велику кількість графічних прискорювачів. В більшості випадків такі об'єми даних і ресурсів не є доступними для навчання моделей для власних класів об'єктів. Саме в таких випадках використовується підхід, що називається навчання через перенос (англ. transfer learning).

Навчання через перенос – процес навчання моделі нейронної мережі, базуючись на вже натренованій моделі і застосовуючи вже отримані знання з певного домену. Вважається, що вже натренована на великій кількості загальних даних модель може ефективно виступати загальною моделлю візуального світу. При цьому, перенавчається не вся модель, а лише останній шар, що безпосередньо відповідає за надання певному зображенню класу.

Навчання було виконано з використанням чотирьох широко використовувані набори даних для оцінки кількості людей у натовпах: UCF-QNRF, UCF_CC_50, ShanghaiTech частини A та B.

UCF-QNRF – це останній і найбільший підрахунок натовпу набір даних, що включає 1535 зображень, узятих з Flickr за допомогою 1,25 млн. точкових анотацій. Це складний набір даних оскільки він має широкий діапазон кількості людей на зображеннях, різноманітні роздільну здатність зображень, умови освітлення та ракурси. Навчальний набір містить 1201 зображення, набір для тестування – 334.

ShanghaiTech [31] складається з частини A та частини B. Частина A містить 300 зображень для навчання та 182 зображення для тестування. Всі зображення були зібрані з Інтернету, і більшість із них – це зображення дуже переповнених сцен, таких як мітинги та великі спортивні події. Частина B містить 400 навчальних зображень і 316 зображень для тестування, зроблених на вулицях Шанхаю. Частина A має значно вищу щільність, ніж частина B.

UCF_CC_50 [32] містить 50 сірих зображень з різною роздільною здатністю. Середнє число людей для кожного зображення – 1280, мінімальна та максимальна кількість – 94 та 4532 відповідно.

4 ЕКСПЕРИМЕНТИ ТА РЕЗУЛЬТАТИ

Для проведення експериментів були обрані відео з різною кількістю людей та щільністю натовпу, зняті з різних ракурсів. Метриками для оцінки якості роботи методів були обрані стандартне абсолютне відхилення (MAE – Mean Absolute Error) та стандартне квадратичне відхилення (RMSE – Root Mean Square Error) [33].

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \ , \tag{4.1}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \ , \tag{4.2}$$

де N – кількість експериментів;

y_i – очікуваний результат;

$\hat{y}_i$ – отриманий результат.

MAE показує абсолютну різницю між реальними даними і отриманим результатом, при цьому великі помилки караються не сильно. RMSE використовує квадрат різниці між отриманим результатом і реальними даними. При цьому великі помилки караються сильніше через використання квадрату різниці.

У таблиці 4.1 можна побачити розраховані значення MAE та RMSE.

Таблиця 4.1 – Експериментальні результати

Метод	Гістограми напрямлених градієнтів	Нейронна мережа
MAE	209,25	27,63
RMSE	360,14	42,6

З результатів розрахунку метрик можна зробити висновок, що метод на основі нейронних мереж дає значно кращі результати, аніж метод на основі гістограми напрямлених градієнтів. Це можна пояснити тим, що метод на основі гістограмам напрямлених градієнтів для підрахунку людей використовує розпізнавання голови та облич і класифікатор був навчений на зображеннях, де обличчя може бути видно краще, аніж на тестових даних, або бути зображено у іншій перспективі, мати велику кількість варіацій в деталях. Також у великих скупченнях людей більша частина голови або обличчя може бути перекрита іншою людиною, що завадить розпізнаванню. Приклад цього можна побачити на рисунку 4.1.

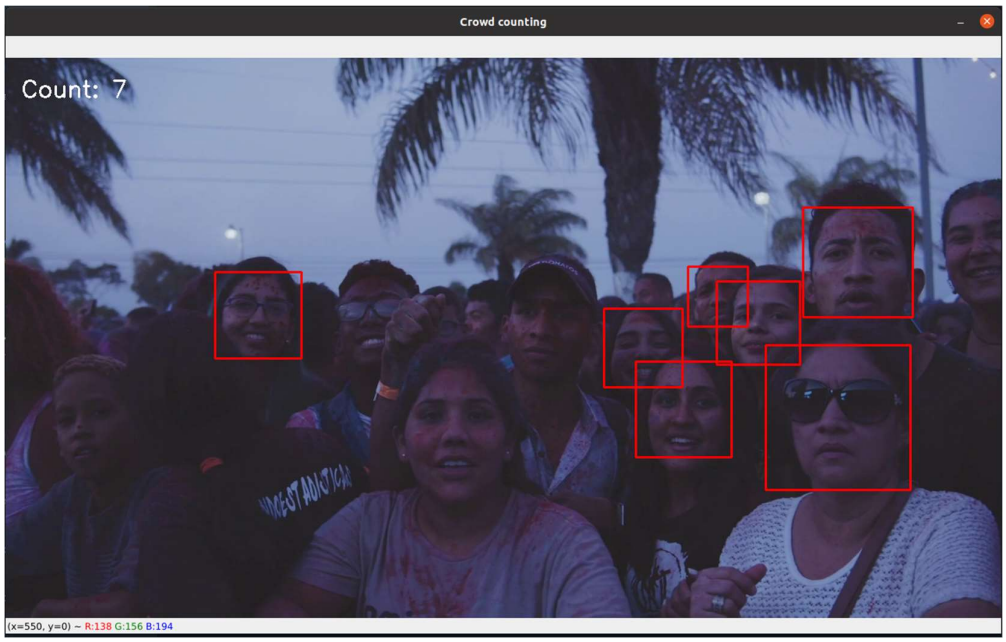

Рисунок 4.1 – Приклад роботи методу на основі HOG на зображенні з оклюзією облич

На тому ж зображенні нейронна мережа дає значно кращий результат (рис. 4.2).

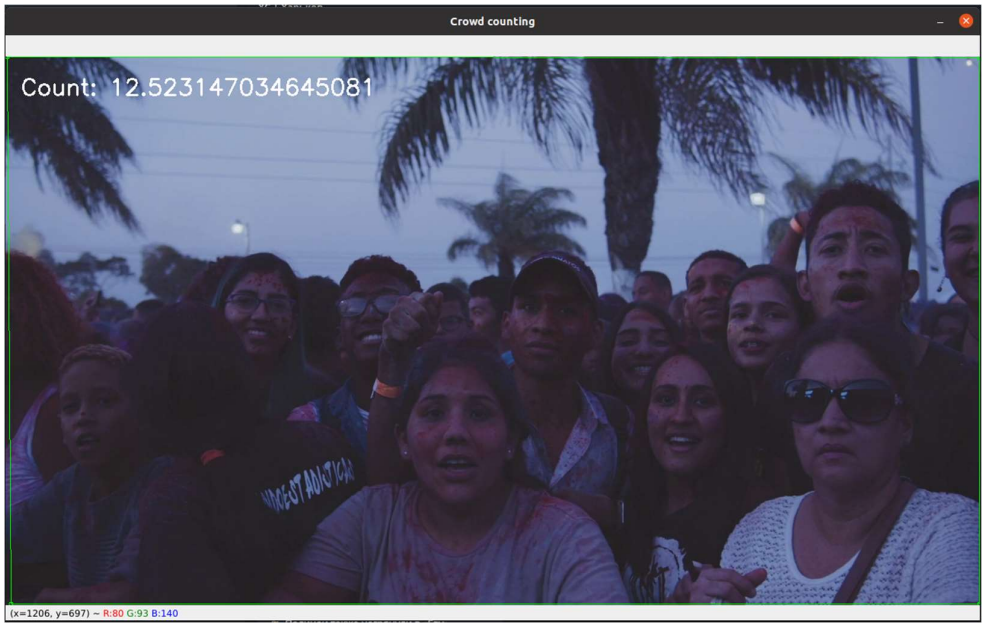

Рисунок 4.2 – Результат роботи методу на основі нейронної мережі на зображенні з оклюзією облич

Також на зображеннях з великою кількістю людей метод на основі нейронної мережі (рис. 4.3) дає набагато кращі результати аніж метод на основі HOG (рис. 4.4).

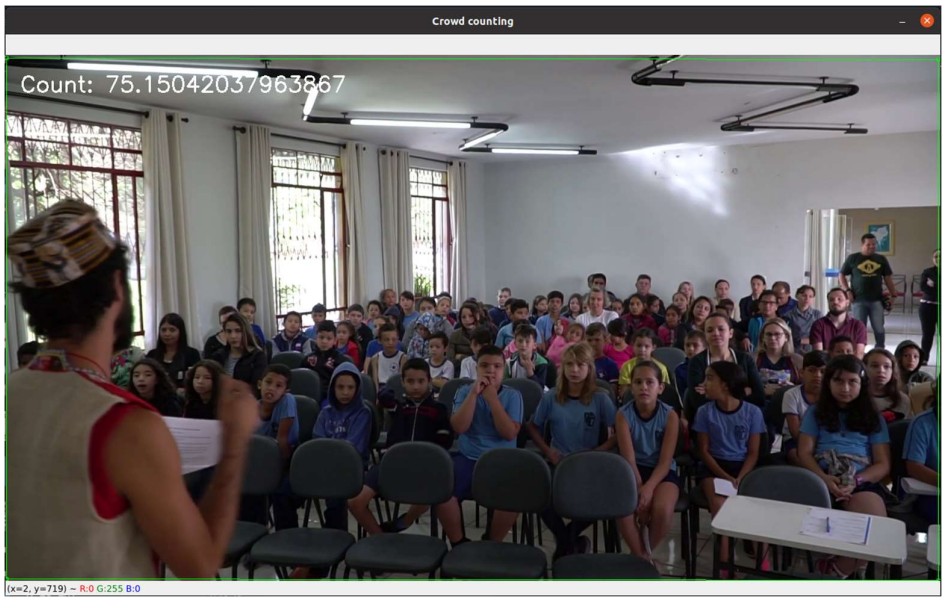

Рисунок 4.3 – Результат роботи методу на основі нейронної мережі на зображенні з великою кількістю людей

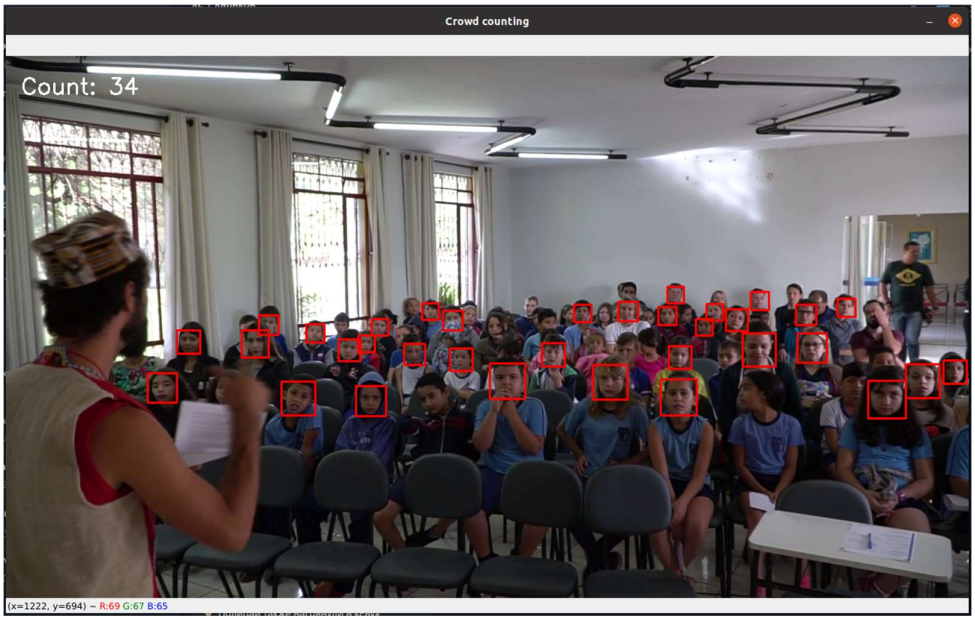

Рисунок 4.4 – Результат роботи методу на основі HOG на зображенні з великою кількістю людей

На зображеннях, де кількість людей невелика, обидва методи показують приблизно однакові результати. Так, на рисунку 4.5 та 4.6 можна побачити результати роботи методів на основі HOG та нейронних мереж відповідно.

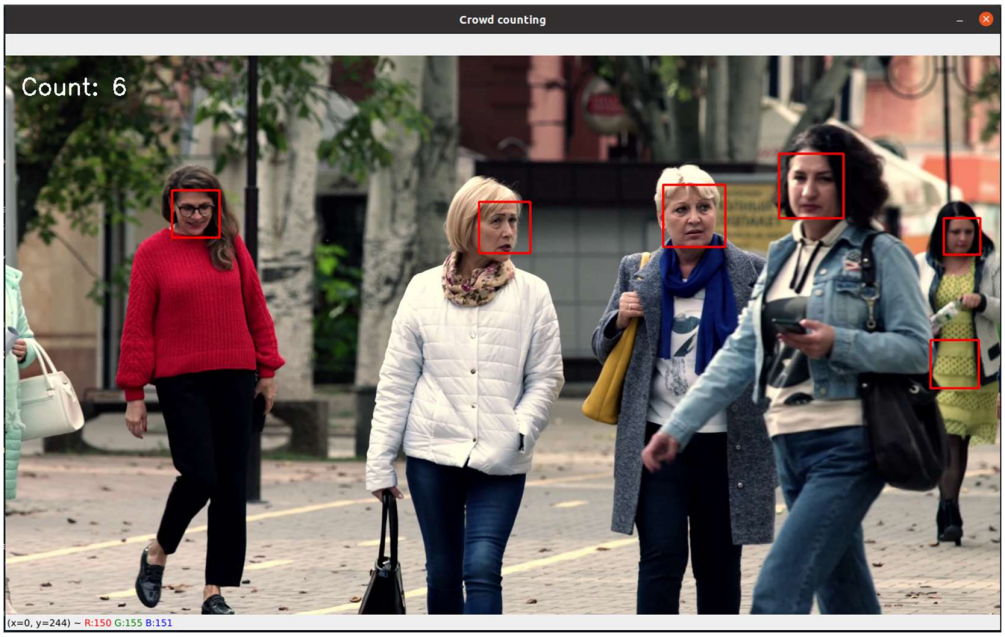

Рисунок 4.5 – Приклад роботи методу на основі HOG на зображенні з невеликою кількістю людей

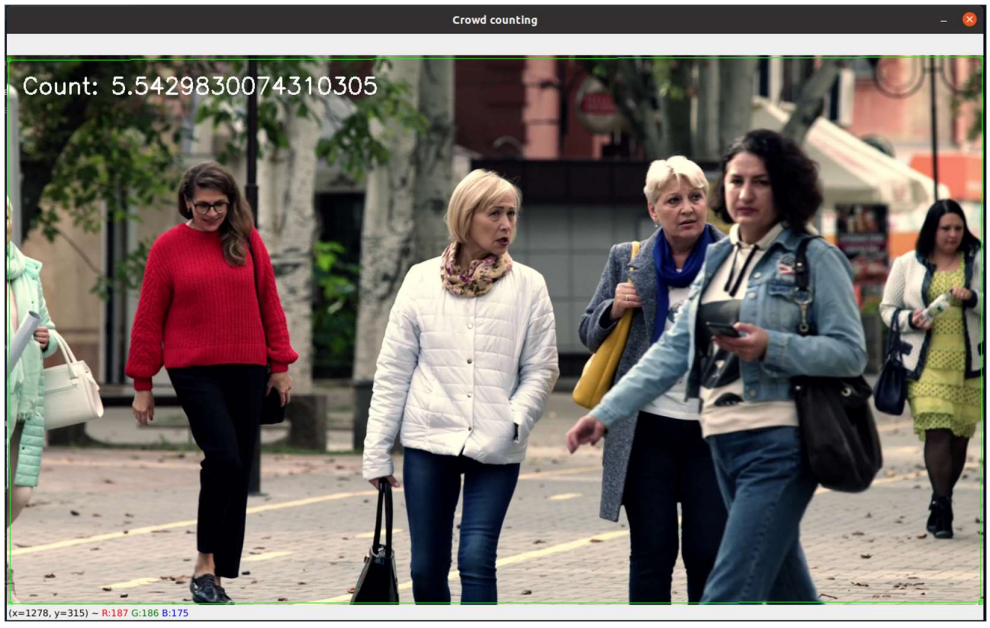

Рисунок 4.6 – Приклад роботи методу на основі нейронної мережи на зображенні з невеликою кількістю людей

ВИСНОВКИ

В результаті реалізації програмного застосунку, що має оцінювати кількість людей у натовпі методами підрахунку шляхом розпізнавання методом гістограми напрямлених градієнтів та підрахунку з використанням нейронних мереж, а також в результаті порівняння та результатів роботів цих методів на різних наборах даних можна зробити висновок, що методи, що базуються на використанні нейронних мереж дають кращі результати через те, що в процесі навчання нейронна мережа знаходить сховані ознаки, що можуть бути використані при застосуванні мережі відеоряду. Це означає, що випадки які дуже важко передбачити та опрацювати також можуть бути використані для підрахунку кількості людей.

Основною перевагою використання нейронних мереж для підрахунку кількості людей є те, що навчена нейронна мережа може бути легко застосована на відеорядах із різними перспективами при цьому даючи гарний результат.

Програмна реалізація згорткових нейронних мереж базується на використанні технологій глибинного навчання представлених у вигляді бібліотек та спеціальних пакетів, що дозволяють розробникам проектувати архітектури мереж та налаштовувати параметри.

Важливим етапом покращення дії класифікації є вибір базової моделі згорткової мережі та обрахунок її показників точності, параметрів в залежності від розміру кроку навчання та використаного алгоритму, аби надалі мати розуміння які саме архітектурні рішення вносити, щоб досягти оптимізації моделі та покращення роботи.

Програмно метод із застосуванням згорткової нейронної мережі було реалізовано за допомогою бібліотеки PyTorch, а реалізація методу із застосуванням алгоритму HOG було реалізовано з використанням бібліотеки OpenCV. Були застосовані знання модульного об’єктно-орієнтованого програмування (ООП).

Результатом роботи став програмний застосунок, що може використовуватися організаціями з громадського спостереження для оцінки кількості людей у натовпах (оцінка кількості учасників мирних зібрань), що використовує згорткову нейронну мережу.

Результати тестування застосунку підтвердили доцільність і ефективність використання згорткових нейронних мереж у цілях оцінки кількості людей у натовпі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Sheng-Fuu Lin, Jaw-Yeh Chen, and Hung-Xin Chao, “Estimation of number of people in crowded scenes using perspective transformation,” IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, vol. 31, no. 6, pp. 645–654, 2001.

2. N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 2005, pp. 886-893 vol. 1, doi: 10.1109/CVPR.2005.177.

3. Meng Wang and Xiaogang Wang, “Automatic adaptation of a generic pedestrian detector to a specific traffic scene,” in Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on. IEEE, 2011, pp. 3401–3408.

4. L. Zhang, M. Shi and Q. Chen, "Crowd Counting via Scale-Adaptive Convolutional Neural Network," 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, 2018, pp. 1113-1121, doi: 10.1109/WACV.2018.00127.

5. D. Moctezuma, C. Conde, I. M. de Diego and E. Cabello, "Person detection in surveillance environment with HoGG: Gabor filters and Histogram of Oriented Gradient," 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), Barcelona, 2011, pp. 1793-1800, doi: 10.1109/ICCVW.2011.6130466.

6. Weina Ge and Robert T Collins, “Marked point processes for crowd counting,” in Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. IEEE, 2009, pp. 2913–2920.

7. Haroon Idrees, Khurram Soomro, and Mubarak Shah, “Detecting humans in dense crowds using locallyconsistent scale prior and global occlusion reasoning,” IEEE transactions on pattern analysis and machine intelligence, vol. 37, no. 10, pp. 1986–1998, 2015.

8. Zhe Lin and Larry S Davis, “Shape-based human detection and segmentation via hierarchical part-template matching,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 4, pp. 604–618, 2010.

9. S. Mashtalir, V. Mashtalir, and M. Stolbovyi, “Video Shot Boundary Detection Via Sequential Clustering”, International Journal “Information Theories and Applications”, Vol.24, No.1, pp.50-59, 2017.

10. R. M. Haralick, K. Shanmugam et al., “Textural features for image classification,” TSMC, no. 6, pp. 610–621, 1973.

11. Bogucharskiy and S. Mashtalir, "Image sequences texture analysis based on vector quantization", Radio Electronics Computer Science Control, no. 2, pp. 94-99, 2014.

12. Mashtalir, S., & Mikhnova, O. (2014). Key Frame Extraction from Video: Framework and Advances. International Journal of Computer Vision and Image Processing (IJCVIP), 4(2), 68-79. doi:10.4018/ijcvip.2014040105

13. A. B. Chan and N. Vasconcelos, “Bayesian poisson regression for crowd counting,” in ICCV. IEEE, 2009, pp. 545–551.

14. Davies, A., Yin, J., Velastin, S.: Crowd monitoring using image processing. Electronics & Communication Engineering Journal 7(1), 37–47 (1995).

15. Tu, P., Sebastian, T., Doretto, G., Krahnstoever, N., Rittscher, J., Yu, T.: Unified crowd segmentation. In: European Conference on Computer Vision (2008)

16. Mashtalir, S., Mashtalir, V.: Spatio-Temporal Video Segmentation. In: Mashtalir V., Ruban I., Levashenko V. (eds) Advances in Spatio-Temporal Segmentation of Visual Data. Studies in Computational Intelligence, vol 876. Springer, Cham. pp. 161-210 (2020) doi: 10.1007/978-3-030-35480 0_4

17. Mashtalir and V. Mashtalir, "Sequential temporal video segmentation via spatial image partitions", 2016 IEEE First International Conference on Data Stream Mining & Processing (DSMP), pp. 239-242, 2016.

18. Mashtalir and O. Mikhnova, "Key frame extraction from video: framework and advances", *Int. J. of Computer Vision and Image Processing*, vol. 4, pp. 68-79, 2014.

19. M. Fu, P. Xu, X. Li, Q. Liu, M. Ye, and C. Zhu, “Fast crowd density estimation with convolutional neural networks,” EAAI, vol. 43, pp. 81–88, 2015.

20. Cong Zhang, Hongsheng Li, Xiaogang Wang, and Xiaokang Yang, “Cross-scene crowd counting via deep convolutional neural networks,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 833–841.

21. Kelly, Colm & Siddiqui, Fahad & Bardak, Burak & Woods, Roger. (2014). Histogram of Oriented Gradients front end processing: an FPGA Based Processor Approach. 10.13140/2.1.1725.9201.

22. M. A. Hassan, I. Pardiansyah, A. S. Malik, I. Faye and W. Rasheed, "Enhanced people counting system based head-shoulder detection in dense crowd scenario," 2016 6th International Conference on Intelligent and Advanced Systems (ICIAS), Kuala Lumpur, 2016, pp. 1-6, doi: 10.1109/ICIAS.2016.7824053.

23. Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, “Single-image crowd counting via multi-column convolutional neural network,” in CVPR, 2016, pp. 589– 97.

24. L. Boominathan, S. S. Kruthiventi, and R. V. Babu, “Crowdnet: A deep convolutional network for dense crowd counting,” in ACM MM. ACM, 2016, pp. 640–644.

25. M. Li, Z. Zhang, K. Huang, and T. Tan, “Estimating the number of people in crowded scenes by mid based foreground segmentation and head-shoulder detection,” in ICPR. IEEE, 2008, pp. 1–4.

26. Mashtalir, S. V., Stolbovyi, M. I., & Yakovlev, S. V. (2019). Clustering Video Sequences by the Method of Harmonic k-Means. Cybernetics and Systems Analysis, 55(2), 200-206.

27. J. Ilao and M. Cordel, "Crowd Estimation Using Region-Specific HOG With SVM," 2018 15th International Joint Conference on Computer Science and Software Engineering (JCSSE), Nakhonpathom, 2018, pp. 1-5, doi: 10.1109/JCSSE.2018.8457384.

28. S. Mashtalir and O. Mikhnova, "Detecting Significant Changes in Image Sequences" in Multimedia Forensics and Security. Intelligent Systems Reference Library, Cham:Springer, vol. 115, pp. 161-191, 2017.

29. S. Albawi, T. A. Mohammed and S. Al-Zawi, "Understanding of a convolutional neural network," 2017 International Conference on Engineering and Technology (ICET), Antalya, 2017, pp. 1-6, doi: 10.1109/ICEngTechnol.2017.8308186.

30. Scherer, Dominik & Müller, Andreas & Behnke, Sven. (2010). Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition. 92-101. 10.1007/978-3-642-15825-4_10.

31. Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma. Singleimage crowd counting via multi-column convolutional neural network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 589–597, 2016.

32. H. Idrees, I. Saleemi, C. Seibert, and M. Shah, “Multi-source multiscale counting in extremely dense crowd images,” in CVPR, 2013, pp. 2547–2554.

33. L. Zeng, X. Xu, B. Cai, S. Qiu and T. Zhang, "Multi-scale convolutional neural networks for crowd counting," 2017 IEEE International Conference on Image Processing (ICIP), Beijing, 2017, pp. 465-469, doi: 10.1109/ICIP.2017.8296324.