Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Модель та метод розпізнавання
руху людини у режимі реального часу

(тема)

Виконав:

студент II курсу, групи СПм-21-2

Вишнівський Д.В.
(прізвище, ініціали)

Спеціальність

123 «Комп’ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма

Системне програмування
(повна назва освітньої програми)

Керівник: к.т.н. Єрьоміна Н.С.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2023 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системне програмування
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20___ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Вишнівському Данііілу Валерійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Модель та метод розпізнавання руху людини у режимі реального часу

затверджена наказом по університету від “ 03 ” квітня 2023 р. № 318 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 17 травня 2023 р.

3. Вхідні дані до роботи Створити модель для відстежування рухів людини у реальному

4. Перелік питань, що потрібно опрацювати у роботі

1. Розробити модель
2. Розробити програмну реалізацію
3. Оцінити модель та зробити висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) Слайд-презентація – 12 слайдів

6. Консультанти розділів роботи (заповнюється 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

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної області	03.04.23-11.04.23	
2	Формування вимог системи	12.04.23-20.04.23	
3	Розробка та налагодження системи	21.04.23-26.04.23	
4	Тестування	27.04.23-30.04.23	
5	Оформлення матеріалів кваліфікаційної роботи	1.05.23-05.05.23	
6	Подання роботи керівникові та	06.05.23-10.05.23	
	її попередній захист		
7	Подання роботи на рецензування	10.05.23-16.05.23	

Дата видачі завдання 03 квітня 2023 р.

Студент _____
(підпис)

Керівник роботи _____ ст.викл. Єрьоміна Н.С.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 52 с., 7 рис., 1 табл., 1 дод., 21 джерел.

НЕЙРОННІ МЕРЕЖІ, КОМП'ЮТЕРНИЙ ЗІР, ГЛИБОКЕ НАВЧАНЯ, ОЦІНКА ПОЗИ ЛЮДИНИ, ЗГОРТКОВІ НЕЙРОННІ МЕРЕЖІ

Метою кваліфікаційної роботи є реалізація моделі, яка буде взмозі вирішувати актуальну проблему захвату рухів кінцівок людини за веб-камери використовуючи методи глибокого навчання.

У ході виконання кваліфікаційної роботи було проведено аналіз існуючих методів та моделей для розпізнавання кінцівок людини . Крім того розроблено програмне забезпечення для реалізації моделі. Проведені тести та зроблені висновки.

ABSTRACT

Master’s thesis: 52 pages, 7 figures, 1 tables, 1 appendices, 21 sources.

NEURAL NETWORKS, COMPUTER VISION, DEEP LEARNING, HUMAN POSУ ESTIMATION, CONVOLUTIONAL NEURAL NETWORKS

The goal of the qualification work is to implement a model that will be able to solve the actual problem of capturing the movements of human limbs using web cameras using deep learning methods.

In the course of the qualification work, an analysis of existing methods and models for recognizing human limbs was carried out. In addition, software for implementing the model has been developed. Tests were conducted and conclusions were formulated.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AP – середня точність (англ., Average Precision)

AR - середнє запам'ятовування (англ., Average Recall)

CNN - згорточна нейронна мережа (англ., Convolutional Neural Network)

CPU - центральний процесор (англ., Central Processing Unit)

CV – комп’ютерний зір (англ., Computer Vision)

GPU – графічний процесор (англ., Graphic Processing Unit)

HPE – оцінка пози людини (англ., Human Pose Estimation)

MSE – середня квадратична помилка (англ., Mean Squared Error)

MPJPE -середня помилка для кожної кінцівки (англ., Mean Per Joint Position Error)

OKS - подібність ключових точок об’єкт (англ., Object Keypoint Similarity)

PCK - відсоток правильних ключових точок (англ., Percent Correct Keypoints)

ReLU - Зрізаний лінійний вузол (англ., Rectified Linear Unit)

SGD - Стохастичний градієнтний спуск (англ., Stochastic Gradient Descent)

TPU – тензорний процесор (англ., Tensor Processing Unit)

ВСТУП

Human Pose Estimation [1, 2] є фундаментальною областю досліджень у галузі Computer Vision, яка швидко розвивається. Вона спрямована на оцінку положення та орієнтації суглобів людського тіла на зображеннях або відео, що дозволяє комп'ютеру розуміти й аналізувати пози людського тіла.

Здатність сприймати та інтерпретувати пози людини [3, 4] має далекосяжні наслідки для багатьох застосувань, таких як:

Взаємодія людини з комп'ютером: оцінку пози можна використовувати для створення природних інтерфейсів користувача для керування пристроями, програмами чи іграми за допомогою жестів тіла.

Розпізнавання активності: аналізуючи пози тіла людей у сцені, ви можете визначити, якою діяльністю вони займаються, як-от ходьба, біг або сидіння.

Анімація та ігри: оцінку пози можна використовувати для анімації віртуальних персонажів, забезпечуючи реалістичні рухи та взаємодію у відеоіграх або створених комп'ютером фільмах.

Аналіз спорту: тренери та спортсмени можуть використовувати оцінку пози, щоб аналізувати та вдосконалювати свої техніки, пози та рухи під час тренувань або змагань.

Охорона здоров'я та реабілітація: оцінку пози можна використовувати для спостереження за пацієнтами під час фізіотерапії, відстеження їх прогресу та надання відгуків про їхні вправи.

Фітнес і самопочуття [5]: програми можуть використовувати оцінку пози, щоб пропонувати вказівки в режимі реального часу та відгуки про тренування, пози йоги або танцювальні рухи.

Спостереження та безпека. Оцінка пози може допомогти у виявленні незвичайних або небезпечних дій у громадських місцях, наприклад виявлення падінь, бійок або нещасних випадків.

Робототехніка: роботи можуть використовувати оцінку пози людини, щоб розуміти діяльність людини та ефективніше взаємодіяти з людьми, забезпечуючи кращу співпрацю між людьми та роботами.

Мода та роздрібна торгівля: оцінку пози можна використовувати для створення віртуальних примірочних, що дозволяє клієнтам бачити, як одяг підходить на цифровому зображенні себе.

Доповнена реальність. Оцінюючи пози людини, програми доповненої реальності можуть накладати віртуальний вміст на сцени реального світу більш залежно від контексту та інтерактивно.

Існує кілька популярних архітектур глибокого навчання [6, 7] для HPE, таких як Stacked Hourglass Networks [8], Simple Baselines [9], Convolutional Pose Machines, OpenPose та. Ці архітектури використовують згорточні нейронні мережі для оцінки поз людини. Останні досягнення також включають використання моделей на основі трансформерів, таких як Vision Transformers [10], для завдань оцінки пози.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Актуальність обраної теми

Оцінка пози людини є активною сферою досліджень комп’ютерного зору, яка досягла значного прогресуза останні роки завдяки розробці методів глибокого навчання, зокрема згорткових нейронних мереж Актуальність і важливість цієї галузі, зумовлена різноманітністю її застосувань і постійною потребою у вдосконаленні точності, швидкості та надійності.

Деякі фактори, які впливають на актуальність HPE, включають:

Різноманітність застосувань: HPE має численні практичні застосування в різних областях, що робить його важливою сферою дослідження. Приклади включають взаємодію людини з комп’ютером, аналіз руху, спортивну аналітику, відеоспостереження, робототехніку, віртуальну та доповнену реальність, охорону здоров’я, анімацію та створення контенту. Широкий спектр застосувань спонукає дослідників і практиків розробляти більш точні, ефективні та універсальні моделі HPE.

Складні проблеми: HPE за своєю суттю є складною проблемою через такі фактори, як оклюзії, варіації освітлення та зовнішнього вигляду, різноманітні людські пози та присутність кількох людей на одній сцені. Ці виклики спонукають дослідників розробляти нові моделі та методи, щоб подолати такі труднощі та розширити межі того, що можливо в HPE.

Досягнення глибокого навчання. Успіх глибокого навчання комп’ютерного зору приніс велику користь галузі HPE, оскільки такі моделі, як CNN, ResNets і Transformers, значно перевершують традиційні методи. Оскільки методи глибокого навчання продовжують розвиватися, цілком імовірно, що HPE також побачить додаткові покращення в точності та ефективності.

Масштабні набори даних: доступність великомасштабних анотованих

наборів даних для HPE, таких як MS COCO та MPII Human Pose, сприяла розробці та оцінці нових моделей і методів. Ці набори даних дозволяють дослідникам тренуватися та перевіряти свої моделі на різноманітних і складних даних, сприяючи постійному прогресу в HPE.

Вимоги до роботи в режимі реального часу: багато програм HPE вимагають продуктивності в режимі реального часу, що вимагає ефективних моделей, здатних швидко обробляти й аналізувати зображення чи відеопотоки. Ця потреба у швидкості та ефективності заохочує розробку легких моделей і методів оптимізації для забезпечення HPE у реальному часі.

Таким чином, актуальність HPE полягає в його різноманітних додатках, складних проблемах, прогресі в глибокому навчанні, доступності великомасштабних наборів даних і вимогах реального часу. Ці фактори сприяють незмінній важливості та актуальності HPE як в наукових колах, так і в промисловості, стимулюючи продовження досліджень та інновацій у цій галузі.

1.2 Існуючі моделі та рішення

HPE передбачає прогнозування просторового розташування суглобів або ключових точок людського тіла на основі вхідних даних [11, 12], таких як зображення чи відео. Зазвичай він зосереджується на оцінці пози у 2D або 3D:

Оцінка 2D пози[13]: оцінює двовимірні координати (x, y) ключових точок тіла в кадрі зображення або відео. Він широко використовується завдяки своїй обчислювальній ефективності та застосовності для різних програм.

Оцінка 3D пози: оцінює 3D координати (x, y, z) ключових точок тіла. Він надає більше інформації про позу, але вимагає додаткової інформації або припущень, як-от параметри камери, інформація про глибину або тимчасову інформацію між відеокадрами.

За останнє десятиліття HPE досягла значних успіхів, головним чином завдяки поширенню методів глибокого навчання [14, 15] та доступності

великомасштабних анотованих наборів даних. Convolutional Neural Networks з’явилися як потужний підхід до моделювання складних просторових зв’язків між суглобами тіла та досягнення найсучаснішої продуктивності за різними тестами HPE.

Ось основні підходи до HPE:

Підхід «зверху вниз» [16]: включає два основні кроки: виявлення людини та локалізацію ключових точок. По-перше, детектор людини (наприклад, Faster R-CNN, YOLO або SSD) використовується для визначення місцезнаходження людей на вхідному зображенні. Потім модель оцінки пози застосовується до кожної виявленої обмежувальної рамки для оцінки ключових точок.

Підхід «знизу вгору». Цей підхід спочатку передбачає індивідуальні ключові точки тіла для всіх людей на вхідному зображенні. Потім використовується алгоритм групування (наприклад, жадібний висновок, двостороння відповідність), щоб асоціювати виявлені ключові точки з кожною особою [17]. Цей підхід є обчислювально ефективним для сцен із кількома людьми.

HPE є однією з проблем, які вирішуються за допомогою класифікації зображень [18]. Класифікація зображень — це фундаментальне завдання комп’ютерного зору, метою якого є присвоєння попередньо визначеної мітки класу вхідному зображенню на основі наявних у ньому об’єктів або функцій. Типовий алгоритм виконання класифікації зображень складається з кількох кроків, таких як попередня обробка, виділення ознак і класифікація. Останніми роками методи глибокого навчання, зокрема згорткові нейронні мережі, стали основним підходом для завдань класифікації зображень завдяки їхній чудовій продуктивності.

Vision Transformers з’явилися як потужна архітектура для задач комп’ютерного зору, спочатку запропонована Досовицьким та ін. для класифікації зображень. Вони показали чудову продуктивність, часто перевершуючи традиційні згорточні нейронні мережі у різних завданнях.

Основна ідея Vision Transformers полягає в тому, щоб використати механізм самоконтролю з архітектури Transformer, спочатку розроблений для обробки природної мови, і застосувати його до зображень, розглядаючи їх як послідовності токенів.

Нещодавно дослідники почали досліджувати потенціал Vision Transformers для завдань оцінки пози. Щоб адаптувати архітектуру ViT для оцінки пози людини, можна розглянути кілька модифікацій і підходів:

Токенізація: розділіть вхідне зображення на фрагменти, що не перекриваються, і лінійно вбудуйте їх як маркери. Включіть додатковий навчальний маркер, відомий як маркер класу, для агрегування глобальної інформації по всьому зображенню.

Прогноз Heat map: зміна output layer архітектури ViT, щоб передбачити Heat map для кожної ключової точки. Heat map — це 2D карти ймовірностей, які представляють ймовірність присутності ключової точки в кожному пікселі. Остаточне розташування ключових точок можна отримати, знайшовши максимальне значення на кожній тепловій карті.

Багатоетапне уточнення: Включіть багатоступеневу архітектуру для повторного уточнення оцінки пози. Кожен етап може складатися з модуля Vision Transformer, за яким слідує вихідний рівень передбачення. Проміжний нагляд може бути застосований, щоб заохотити модель вивчати кращі проміжні уявлення.

Оцінка пози кількох людей: для завдань оцінки пози кількох людей додайте додатковий вихідний рівень, щоб передбачити присутність людей на зображенні разом із їхніми відповідними ключовими точками.

Кілька останніх робіт успішно застосували Vision Transformers для завдань оцінки пози людини, досягнувши конкурентоспроможності порівняно з традиційними методами на основі CNN. Деякі помітні приклади включають Swin Transformer (Liu та ін.) і VisTR (Wang та ін.), які демонструють потенціал архітектур на основі Transformer для оцінки пози.

Незважаючи на те, що Vision Transformers показала багатообіцяючі

результати в задачах оцінки пози, залишаються проблеми, які необхідно вирішити, такі як обчислювальна складність і потреба у великих обсягах навчальних даних. Поточні дослідження в цій галузі спрямовані на подолання цих проблем і подальше покращення продуктивності для оцінки пози людини та інших завдань комп'ютерного зору.

The Stacked Hourglass Model

The Stacked Hourglass Model, запропонована Ньюелом та ін. у їхній статті — це архітектура глибокого навчання, спеціально розроблена для завдань оцінки пози людини. Ця модель була особливо впливовою в цій галузі, оскільки вона запровадила новий підхід до захоплення та об'єднання багатомасштабної контекстної інформації в структурі глибокого навчання.

The Stacked Hourglass Model складається з кількох модулів, кожен з яких відповідає за прогнозування теплових мап для кожної ключової точки. Загальна архітектура є симетричною, із низкою шарів зниження дискретизації, за якими слідують шари підвищення дискретизації, що нагадує форму пісочного годинника. Ось огляд ключових компонентів та ідей у мережі Stacked Hourglass:

Модуль «The Stacked Hourglas»: модуль призначений для захоплення й обробки контекстної інформації в межах зображення. Він складається з серії convolution і pool layer, які поступово зменшують дискретизацію вхідного сигналу, а потім серії підвищення дискретизації та згорткових шарів, які поступово відновлюють вихідну роздільну здатність. Етапи зменшення та підвищення дискретизації з'єднані через skip connection, що дозволяє моделі вивчати як локальний, так і глобальний контекст.

Intermediate supervision: Stacked Hourglass Network використовує проміжний нагляд між модулями пісочного годинника. Після кожного модуля пісочного годинника модель прогнозує теплову карту для кожної ключової точки. Прогнозовані теплові карти порівнюються з тепловими картами реального стану землі, а втрати поширюються у зворотному напрямку, щоб уточнити прогнози моделі. Цей проміжний нагляд спонукає

модель вивчати кращі проміжні представлення.

Мережа Stacked Hourglass виявилася дуже успішною в задачах оцінки пози людини, досягнувши найсучаснішої продуктивності за кількома тестами, коли її було представлено. Її дизайн і принципи вплинули на багато наступних моделей оцінки пози. У той час як з’явилися нові архітектури, такі як ті, що базуються на Vision Transformers, Stacked Hourglass Network залишається цінним орієнтиром для розуміння та розробки моделей оцінки пози людини.

Модель Simple Baseline for HPE— це підхід, заснований на глибокому навчанні, який зосереджується на простоті та ефективності при досягненні конкурентоспроможності порівняно зі складнішими моделями. Модель Simple Baselines використовує модель ResNet для виділення фічей, а потім кілька деконволюційних шарів для прогнозування теплових карт для кожної ключової точки.

Рисунок 1.1 – Результати роботи моделі HPE

У моделі використовується модель ResNet (наприклад, ResNet-50, ResNet-101 або ResNet-152) для вилучення високорівневих фічей із вхідного зображення. Архітектури ResNet довели свою ефективність для різних

завдань комп’ютерного зору завдяки своїй здатності вивчати глибокі представлення, одночасно пом’якшуючи проблему зникнення градієнта через залишкові з’єднання.

Модель Simple Baseline виділяється своєю простотою та обчислювальною ефективністю, зберігаючи конкурентоспроможність у завданнях оцінки пози людини. Її простий дизайн полегшує реалізацію, розуміння та розширення порівняно зі складнішими моделями, що робить її привабливим вибором для різноманітних програм оцінки пози.

1.3 Постановка задачі

Враховуючи останні досягнення у області Deep Learning, доступність великих датасетів, які необхідні для ефективного навчанні моделі, то здешевлення обчислювальних ресурсів, стало можливим запровадити широке застосування нейронних мереж для вирішення задач Human Pose Estimation. Метою цієї роботи є дослідження існуючих моделей, їх порівняння. Після цього доцільно вибрати модель, яка по сумі показників є кращою та її програмна реалізація. Результат даної роботи допоможе у майбутньому полегшити проектування та реалізацію моделей нейронних мереж, для вирішення задачач Human pose Estimation. Також результати дослідження можливо використовувати у навчанні методам та моделям Computer Vision

2 АНАЛІЗ ТЕХНОЛОГІЙ

HPE є однією з проблем, які вирішуються за допомогою класифікації зображень. Класифікація зображень [19] — це фундаментальне завдання комп’ютерного зору, метою якого є присвоєння попередньо визначеної мітки класу вхідному зображенню на основі наявних у ньому об’єктів або функцій. Типовий алгоритм виконання класифікації зображень складається з кількох кроків, таких як попередня обробка, виділення ознак і класифікація. Останніми роками методи глибокого навчання, зокрема згорткові нейронні мережі (CNN), стали основним підходом для завдань класифікації зображень завдяки їхній чудовій продуктивності.

Ось огляд високого рівня алгоритму для виконання класифікації зображень за допомогою CNN:

Попередня обробка даних: підготовка вхідних даних, виконуючи такі операції, як зміна розміру, нормалізація та доповнення даних. Ці кроки допомагають переконатися, що вхідні зображення мають узгоджений формат, і покращують стійкість моделі до варіацій у даних.

Виділення функцій: CNN складається з кількох згорткових, об’єднаних і активаційних рівнів, які навчаються витягувати значущі ознаки з вхідного зображення. Ці шари охоплюють як низькорівневі функції, такі як краї та текстури, так і високорівневі функції, такі як частини об’єктів і семантична інформація. Результатом етапу виділення ознак є карта ознак великого розміру, яка представляє вхідне зображення в більш компактний та інформативний спосіб.

Повністю зв’язані шари: після виділення об’єктів один або кілька повністю з’єднаних шарів, також відомих як щільні шари, використовуються для обробки високовимірних карт об’єктів. Ці шари допомагають моделі вивчати нелінійні комбінації витягнутих ознак, що може покращити дискримінаційну силу моделі.

Вихідний рівень: останній рівень CNN – це повністю зв’язаний рівень із стільки нейронів, скільки є міток класів. За цим рівнем зазвичай слідує функція активації softmax, яка перетворює вихідні дані нейронів у ймовірності класу. Функція softmax визначається як:

Функція втрат і оптимізація: CNN навчається за допомогою відповідної функції втрат, такої як втрата перехресної ентропії, яка вимірює різницю між прогнозованими ймовірностями класу та основними мітками класу істинності. Параметри моделі оптимізуються за допомогою методів оптимізації на основі градієнта, таких як стохастичний градієнтний спуск (SGD) або адаптивних оптимізаторів, таких як Adam [20], щоб мінімізувати функцію втрат.

Оцінка: коли модель навчена, її можна оцінити на тестовому наборі, щоб виміряти її продуктивність за допомогою таких показників, як точність, точність, запам’ятовування або оцінка F1.

Глибоке навчання — це підмножина машинного навчання, яка зосереджується на алгоритмах на основі штучних нейронних мереж, зокрема глибоких нейронних мереж. «Глибина» означає кількість шарів у мережі – чим більше шарів, тим глибша мережа. Ці шари є взаємопов’язаними вузлами, які імітують нейрони та організовані у вхідний, вихідний і прихований шари. Алгоритми глибокого навчання вчаться видобувати та перетворювати вхідні дані у формат, який є корисним для поставленого завдання.

Глибоке навчання досягло успіху в багатьох областях, включаючи комп’ютерне зір, обробку природної мови, розпізнавання мовлення тощо. У комп’ютерному зорі, наприклад, згорткові нейронні мережі (CNN) використовуються для виконання таких завдань, як виявлення об’єктів, класифікація зображень і оцінка пози людини (HPE).

У контексті HPE моделі глибокого навчання навчені передбачати положення ключових точок на тілі людини (наприклад, лікоть, коліно, плече тощо) за вхідного зображення або відеокадру. Ці моделі вчаться розпізнавати

шаблони, такі як форми та текстури, у необроблених піксельних даних зображень. Згодом вони можуть ідентифікувати ці шаблони, навіть коли людина на зображенні перебуває в іншій позі, одягнена інакше або закрита іншими предметами.

Перевага використання глибокого навчання для HPE полягає в тому, що ці моделі можна навчити наскрізно, тобто вони вчаться отримувати функції з необроблених даних і робити прогнози самостійно, без необхідності ручного проектування функцій. Це особливо корисно в завданнях HPE, де розробка елементів ручної роботи може бути складною через високу варіативність людських поз.

Підсумовуючи, глибинне навчання надає потужний набір інструментів для завдань HPE, що дозволяє точно й ефективно оцінювати пози людини в різноманітних практичних застосуваннях.

ResNet, скорочення від Residual Network, — це особливий тип згорткової нейронної мережі (CNN), який був представлений Каймінго Хе та його колегами з Microsoft Research у їхній статті «Глибоке залишкове навчання для розпізнавання зображень» у 2015 році. ResNet став переможцем конкурсу ImageNet у 2015 році і з тих пір широко використовується в різних задачах комп’ютерного зору завдяки своїй чудовій продуктивності та здатності ефективно навчати дуже глибокі мережі.

Ключовим нововведенням ResNet є впровадження «залишкових блоків» із скороченими підключеннями (також відомими як пропускні підключення). Ці з’єднання дозволяють додавати вихідні дані одного рівня до вихідних даних іншого рівня, розташованого далі в мережі. Це допомагає пом’якшити проблему зникнення градієнтів, яка може виникнути під час навчання дуже глибоких мереж і може перешкоджати здатності мережі навчатися.

У традиційній CNN кожен рівень вивчає нове представлення вхідних даних. Навпаки, у ResNet кожен рівень вивчає залишкову функцію або вид модифікації представлення, отриманого попереднім рівнем. Цей підхід

дозволяє ResNet ефективно навчати мережі, які набагато глибші, ніж це було можливо раніше.

У контексті оцінки пози людини (HPE) ResNet часто використовується як магістраль або екстрактор функцій. Вхідне зображення передається через ResNet, а вихідні карти функцій використовуються для прогнозування ключових точок пози. Наприклад, поширеним підходом є використання карт функцій із ResNet для створення теплових карт для кожної ключової точки, де розташування максимального значення на тепловій карті відповідає прогнозованому розташуванню ключової точки на вхідному зображенні.

Здатність ResNet вивчати складні та ієрархічні характеристики із зображень завдяки його глибині робить його придатним вибором для завдань HPE, де захоплення деталей, як-от співвідношення частин тіла та артикуляції, може бути вирішальним для точного прогнозування пози. Крім того, використання залишкового навчання допомагає ефективно навчати ці глибокі мережі, ще більше покращуючи їх продуктивність у виконанні завдань HPE.

Python — це високорівнева, універсальна та проста у вивченні мова програмування, яка стала одним із найпопулярніших варіантів для різноманітних додатків, зокрема глибокого навчання, аналізу даних, веб-розробки, автоматизації тощо. Його популярність у глибокому навчанні в основному пояснюється його простотою, читабельністю, широкою бібліотечною підтримкою та активним співтовариством. Серед переваг Python можно виділити:

Читабельність і простота: Python має чистий і легко читабельний синтаксис, що полегшує початківцям вивчення та розуміння мови. Ця простота дозволяє розробникам і дослідникам зосередитися на концепціях глибокого навчання та розробці моделей, не занурюючись у складні мовні конструкції.

Широка підтримка бібліотек: Python має багату екосистему бібліотек і фреймворків, які підтримують глибоке навчання, машинне навчання та аналіз даних. Деякі популярні бібліотеки глибокого навчання в Python включають

TensorFlow, PyTorch, Keras і Theano. Ці бібліотеки надають готові функції, рівні та оптимізатори, які полегшують впровадження, навчання та оцінку моделей глибокого навчання. Крім того, Python пропонує бібліотеки для обробки даних (наприклад, NumPy, Pandas) і візуалізації (наприклад, Matplotlib, Seaborn), які корисні для попередньої обробки даних і аналізу продуктивності моделі.

Активна спільнота: Python має велике й активне співтовариство розробників, дослідників і практиків, які роблять внесок у його зростання та розвиток. Ця спільнота постійно працює над вдосконаленням існуючих бібліотек, створенням нових бібліотек і наданням підтримки через форуми, онлайн-платформи та конференції. Як наслідок, Python отримує переваги від постійних інновацій і великої кількості ресурсів, таких як навчальні посібники, публікації в блогах і наукові статті, які полегшують навчання та вирішення проблем.

Сумісність між платформами: Python не залежить від платформи, тобто він може працювати в різних операційних системах, таких як Windows, macOS і Linux. Ця сумісність полегшує розробку та розгортання моделей глибокого навчання на різних платформах.

Взаємодія: Python підтримує інтеграцію з іншими мовами, такими як C, C+, що дозволяє розробникам використовувати оптимізований низькорівневий код для інтенсивних обчислювальних завдань, зберігаючи при цьому простоту та читабельність Python для завдань високого рівня. Ця функція дозволяє бібліотекам глибокого навчання використовувати високопродуктивний код для таких операцій, як маніпулювання тензорами та прискорення GPU, зберігаючи при цьому зручний інтерфейс Python.

Універсальність: Python — це мова програмування загального призначення, яку можна використовувати в різних сферах, включаючи веб-розробку, автоматизацію та наукові обчислення. Ця універсальність робить Python популярним вибором для проектів глибокого навчання, які можуть включати завдання, окрім розробки моделі, такі як збір даних, попередня

обробка та розгортання.

Підводячи підсумок, можна сказати, що простота, читабельність, широка бібліотечна підтримка, активне співтовариство, крос-платформна сумісність, можливість взаємодії та універсальність роблять Python чудовою мовою для глибокого навчання. Його простота у використанні та широкий спектр ресурсів дозволяють розробникам і дослідникам швидко створювати прототипи та впроваджувати моделі глибокого навчання, водночас отримуючи вигоду від постійних інновацій у галузі.

TensorFlow — це бібліотека машинного навчання з відкритим кодом, розроблена Google, яка в основному використовується для глибокого навчання, машинного навчання та чисельних обчислень. Вона розроблена як ефективний, гнучкий і простий у використанні модуль, що робить її популярним вибором для впровадження різних моделей машинного навчання, включаючи нейронні мережі. TensorFlow підтримує обчислення як на ЦП, так і на ГП, дозволяючи розробникам ефективно навчати та запускати великомасштабні моделі глибокого навчання.

TensorFlow надає символічну математичну бібліотеку, яка представляє обчислення у вигляді графа потоку даних, де вузли представляють математичні операції, а ребра представляють тензори (багатовимірні масиви), які перетікають між цими операціями. Цей підхід на основі графів дозволяє TensorFlow оптимізувати обчислення, автоматично обчислювати градієнти для зворотного поширення та розподіляти обчислення між кількома пристроями чи машинами.

Деякі ключові функції TensorFlow включають:

Гнучкість: TensorFlow підтримує різні типи моделей, включаючи глибоке навчання, навчання з підкріпленням і традиційні алгоритми машинного навчання. Його гнучкий дизайн дозволяє розробникам легко перемикатися між різними архітектурами моделей, функціями втрат і оптимізаторами.

Продуктивність: TensorFlow оптимізовано для обчислень CPU і GPU,

забезпечуючи ефективне виконання для великомасштабних моделей. Він також підтримує розподілені обчислення на кількох пристроях або машинах, що дозволяє користувачам масштабувати свої моделі для ще більших наборів даних і швидшого часу навчання.

Автоматична диференціація: TensorFlow автоматично обчислює градієнти для зворотного поширення, спрощуючи реалізацію алгоритмів оптимізації та знижуючи ймовірність помилок у ручних обчисленнях градієнтів.

Eager execution: у TensorFlow 2.0 представлено Eager execution, яке дозволяє користувачам негайно виконувати операції та отримувати конкретні результати, полегшуючи налагодження та розробку моделей. Eager execution робить TensorFlow більш інтуїтивно зрозумілим і схожим на інші популярні бібліотеки глибокого навчання, такі як PyTorch.

Інтеграція Keras: TensorFlow включає API високого рівня під назвою Keras, який є простим у використанні інтерфейсом для створення та навчання моделей глибокого навчання. Keras надає широкий спектр попередньо створених шарів, оптимізаторів і функцій втрати, що полегшує створення та навчання різних типів нейронних мереж.

Візуалізація: TensorFlow надає інструмент візуалізації під назвою TensorBoard, який дозволяє користувачам відстежувати процес навчання, візуалізувати обчислювальні графіки та аналізувати ефективність моделі за допомогою різних показників і графіків.

Велика спільнота та ресурси: у TensorFlow є велика й активна спільнота, яка постійно робить внесок у його розвиток і пропонує підтримку через форуми, навчальні посібники та дослідницькі статті. Ця велика кількість ресурсів полегшує розробникам вивчення та впровадження TensorFlow у свої проекти.

Таким чином, TensorFlow — це потужна та гнучка бібліотека машинного навчання, яка підтримує широкий спектр моделей і пропонує різноманітні функції, такі як ефективні обчислення, автоматичне розрізнення

та зручний API високого рівня через Keras. Його велика спільнота та ресурси роблять його популярним вибором для впровадження моделей глибокого та машинного навчання як у дослідницьких, так і в промислових умовах.

Google Collaboratory, яку часто називають «Google Colab», — це веб-платформа, яка дозволяє писати та виконувати код Python через браузер. Він особливо популярний у спільнотах наук про дані та машинного навчання з кількох причин:

Безкоштовний доступ до GPU/TPU: Google Colab надає безкоштовний доступ до високоякісних графічних процесорів і тензорних процесорів, які особливо корисні для навчання великих нейронних мереж і прискорення інтенсивних обчислювальних завдань у машинному та глибокому навчанні. .

Нульова конфігурація: вам не потрібно налаштовувати Python і необхідні бібліотеки у вашій системі. Google Colab надає середовище з багатьма попередньо встановленими популярними бібліотеками (такими як TensorFlow, PyTorch, Keras і OpenCV). Це робить його чудовим інструментом для людей, які хочуть почати з машинного чи глибокого навчання, не потребуючи клопоту з налаштуванням усього.

Легкий обмін і співпраця. Оскільки Google Colab є продуктом Google, він чудово інтегрований із Google Диском і Google Таблицями. Ви можете легко ділитися своїми блокнотами Colab, співпрацювати в реальному часі та навіть зберігати та завантажувати дані зі свого Диска Google.

Середовище блокнота Jupyter: Google Colab надає середовище, схоже на блокнот Jupyter, що означає, що ви можете писати та виконувати код, відтворювати Markdown і візуалізувати дані за допомогою діаграм і графіків – усе в одному місці. Це інтерактивне середовище чудово підходить як для навчання, так і для викладання.

Масштабованість і доступність: блокноти Google Colab зберігаються в хмарі. Це означає, що до них можна отримати доступ з будь-якої машини з підключенням до Інтернету. Ви можете розпочати роботу над проектом на одному пристрої та легко перейти до нього на іншому.

Підводячи підсумок, Google Colab — це потужний інструмент для виконання коду Python з наголосом на програмах машинного навчання. Він пропонує безкоштовний доступ до графічних процесорів, просте налаштування та спільний доступ, знайомий інтерфейс ноутбука Jupyter, а також масштабованість і доступність хмарних обчислень. Ці функції роблять Google Colab безцінним інструментом для тих, хто працює із завданнями машинного та глибокого навчання.

3 МОДЕЛЬ ТА ЇЇ РЕАЛІЗАЦІЯ

3.1 Опис моделі

Модель для вирішення задачі HPE включає в себе декілька важливих аспектів. Першим важливим кроком є формульовання мети та цілі моделі. У цій роботі були обрані наступні високорівневі вимоги до моделі:

- вхідні дані являють собою відеопотік з однієї вебкамери;
- нейронна мережа повинна класифікувати кінцівки людини, , оцінити їх місцезнаходження та повернути координати та ймовірності;
- у якості вихідних даних модель повинна повернути вхідний відеопотік разом з накладеними лініями для кожної кінцівки.

Далі потрібно визначитись із датасетом, на якому нейронна мережа зможе навчитись оцінювати пози. Набір даних має містити зображення або відео разом із відповідними анотаціями поз. Загальні набори даних для HPE включають MPII Human Pose, COCO Keypoints і Human3.6M. Для цієї моделі було обрано COCO 2017 [21], як стандартний датасет, що використовується у моделях призначенних для HPE.

Попередня обробка даних: очистка і попередня обробка даних, щоб зробити їх придатними для навчання. Це часто включає зміну розміру зображень, нормалізацію значень пікселів і, змінення даних (за допомогою поворотів, перекладів тощо), щоб збільшити розмір і різноманітність набору даних.

Вибір архітектури нейронної мережі. Оскільки проблема HPE включає в себе классифікацію об’єктів на вхідних даних, то доцільно обрати архітектуру ResNet, яка вже є стандартом для вирішеннях таких задач. Також для отримання heatmap у високої роздільної здатності доцільно додати до оригінальної архітектури декілька deconvulation шарів.

Навчання вибраної моделі на попередньо обробленому наборі даних.

Під час навчання модель вчиться зіставляти вхідні зображення з відповідними ключовими точками пози. Цей крок передбачає введення зображень у модель, обчислення втрат (різниця між прогнозами моделі та справжніми значеннями) для налаштування параметрів моделі.

Після навчання йде оцінка продуктивність моделі на перевірочному наборі (дані не використовувались під час навчання). Загальні оціночні показники для HPE включають відсоток правильних ключових точок, подібність ключових точок об’єкта і середню помилку позиції з’єднання

3.1 Вибір датасету для навчання моделі

Набори даних відіграють вирішальну роль у навчанні згорткових нейронних мереж (CNN) і моделей машинного навчання загалом. CNN, як і інші моделі машинного навчання, навчаються на прикладі. Під час навчання вони коригують свої параметри, щоб мінімізувати різницю між їхніми прогнозами та справжніми значеннями для прикладів у наборі даних. Мета CNN полягає не лише в тому, щоб робити точні прогнози на основі даних навчання, але й у тому, щоб добре узагальнювати нові, невідомі дані. Різноманітний і репрезентативний набір даних гарантує, що модель стикається з широким діапазоном прикладів під час навчання, що допомагає їй вивчити більш загальні закономірності, які застосовуватимуться до нових даних. Набори даних також використовуються для оцінки ефективності моделі. Випробовуючи модель на окремій перевірці або тестовому наборі, ми можемо отримати неупереджену оцінку того, наскільки добре модель працюватиме на нових даних.

Бенчмаркінг: набори даних забезпечують стандартний тест для порівняння різних моделей. Навчаючи та оцінюючи різні моделі на одному наборі даних, ми можемо побачити, які моделі працюють найкраще.

З цих причин наявність високоякісного набору даних часто є одним із найважливіших факторів успіху CNN. Це стосується не лише розміру набору

даних, але й таких факторів, як різноманітність прикладів, точність міток і те, наскільки добре набір даних представляє проблемний простір, у якому буде розгорнуто модель.

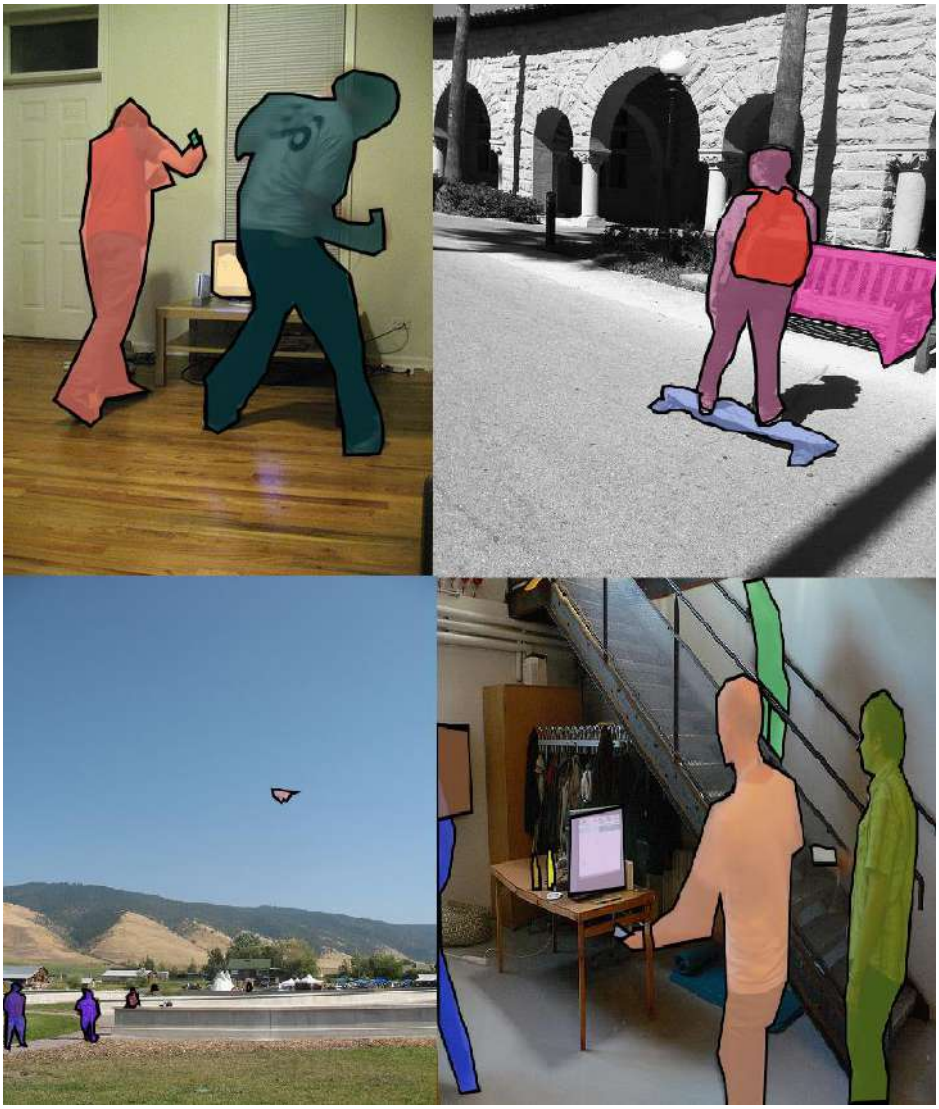

Рисунок 3.1 Приклади зображень з COCO 2017

COCO (Common Object in Context) — це великомасштабний набір даних для виявлення, сегментації та субтитрів. Він був розроблений корпорацією Майкрософт, щоб забезпечити уніфікований набір даних для багатьох завдань комп’ютерного зору з наголосом на виявленні об’єктів у сценаріях реального світу. COCO 2017 — це версія набору даних COCO, що містить дані за 2017 рік. Він містить 118 тис. навчальних зображень, 5 тис.

зображень перевірки та 41 тис. «непозначених» зображень для тестування. Набір даних COCO 2017 складається з:

- зображення складних повсякденних сцен, що містять звичайні об’єкти в їх природному контексті;
- сегментація об’єктів (тобто, які пікселі на зображенні належать якому об’єкту) для підвищення точності;
- п’ять підписів до зображення з чіткими описами;
- анотації ключових точок для зображень людей, корисні для оцінки пози.

Різноманітні сценарії реального світу та докладні анотації COCO роблять його популярним набором даних для навчання та оцінки моделей для цих завдань. Він зазвичай використовується як в наукових колах, так і в промисловості для порівняння продуктивності нових алгоритмів і моделей.

Лістинг 3.1 Завантаження датасету COCO2017

```
    df_train, df_info = tfds.load(name="coco/2017",
split="train", shuffle_files=True, with_info=True)
    df_train = df_train.filter(lambda x, y:
tf.shape(y['bbox'])[0] > 1)
    df_train = df_train.map(functools.partial(to_bgr),
num_parallel_calls=tf.data.experimental.AUTOTUNE)
    df_train = df_train.prefetch(tf.data.experimental.AUTOTUNE)

    df_test = tfds.load(name="coco/2017", split="validation",
shuffle_files=False)
    df_test = df_test.filter(lambda x, y: tf.shape(y['bbox'])[0]
> 1)
    df_test = df_test.prefetch(tf.data.experimental.AUTOTUNE)
```

3.2 Аріхтектура та програмна реалізація нейронної мережі

Як вже було зазначено вище, для реалізації моделі для задача HPE було обрано ResNet. Основними компонентами ResNet є residual blocks. Кожен блок складається з послідовності шарів: двох або трьох згорткових шарів (залежно від версії ResNet), які перемежовуються шарами batch normalisation

та функціями активації ReLU.

Перший згортковий шар у блоці зменшує розмірність вхідних даних (тобто застосовує згортку з невеликою кількістю фільтрів), середні шари обробляють вхідні дані з цією зменшеною розмірністю (зазвичай із згортками 3x3), а останній шар у блок проектує результат назад до початкової кількості каналів.

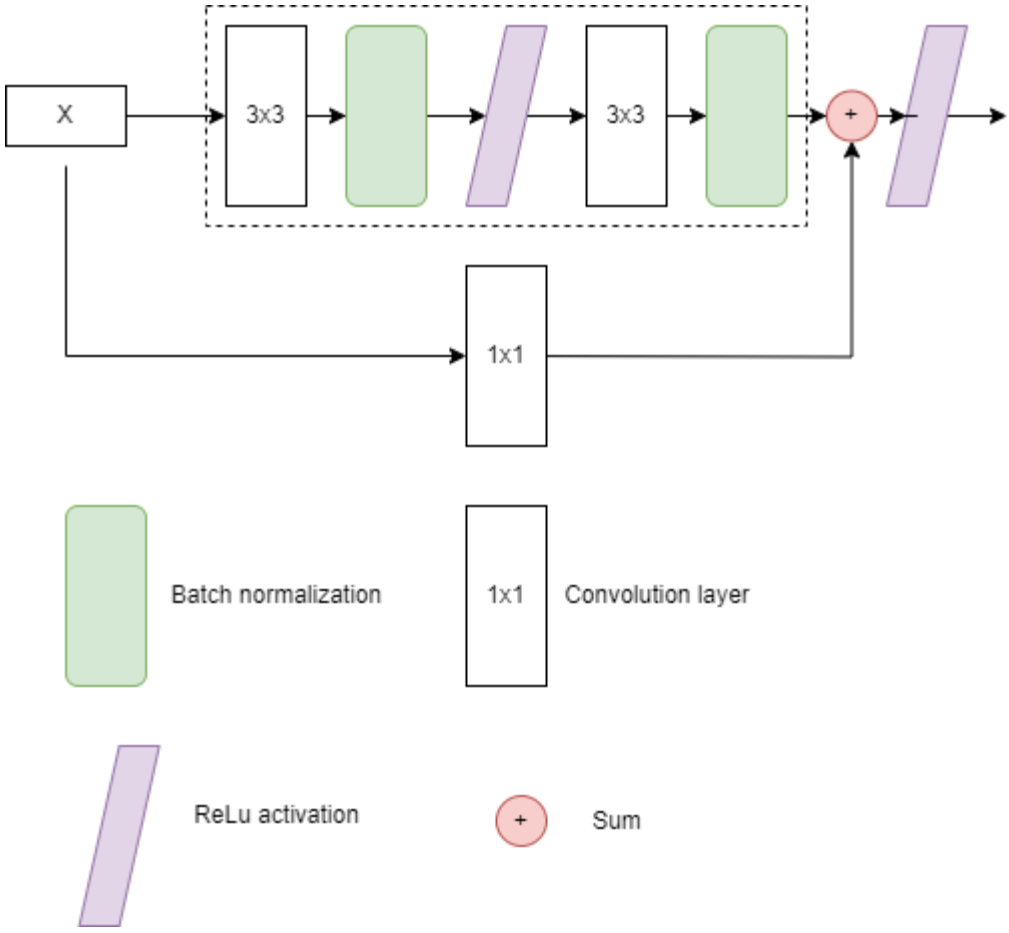

Рисунок 3.2 – Згортковий шар ResNet

Важливо, що існує також з’єднання пропуску, яке переносить вхід блоку безпосередньо на його вихід. Цей пропущений вхід потім поелементно додається до виходу згорткових шарів. Якщо розміри вхідних і вихідних даних не збігаються (через згорткові шари), згортка 1x1 застосовується до пропущених вхідних даних, щоб відповідати розмірам перед додаванням.

Результат цього додавання потім проходить через функцію активації ReLU.

Пропуск з’єднань є ключем до продуктивності ResNet, оскільки вони допомагають пом’якшити проблему зникнення градієнтів у дуже глибоких мережах.

Стеки residual blocks: ResNet складається з кількох стеків цих залишкових блоків. Кожен стек обробляє вхідні дані в іншому масштабі (тобто розмір карт функцій зменшується вдвічі, а кількість фільтрів подвоюється в кожному новому стеку).

Після останнього стеку залишкових блоків застосовується операція global average pooling operation, яка зменшує просторові розміри виходу (від кожного стека залишкових блоків) до 1x1. Це зменшує загальну кількість параметрів і допомагає запобігти переобладнанню.

Лістинг 3.2 Функція з ініціалізацією моделі

```
    def get_model():
        model = models.Sequential()
        input = Input(shape=(train_im.shape[1],
train_im.shape[2], train_im.shape[3])
        model.add(Input(shape=(train_im.shape[1],
train_im.shape[2], train_im.shape[3])))  # cifar 10 images size
        model.add(ZeroPadding2D(padding=(3, 3))(input_im))
        model.add(Conv2D(64, kernel_size=(7, 7), strides=(2,
2))(x))
        model.add(BatchNormalization()(x))
        model.add(Activation(activations.relu)(x))
        model.add(MaxPooling2D((3, 3), strides=(2, 2))(x))
        model.add(res_conv(x, s=1, filters=(64, 256)))
        model.add(res_identity(x, filters=(64, 256)))
        model.add(res_identity(x, filters=(64, 256)))
        model.add(res_conv(x, s=2, filters=(128, 512)))
        model.add(res_identity(x, filters=(128, 512)))
        model.add(res_identity(x, filters=(128, 512)))
        model.add(res_identity(x, filters=(128, 512)))
        model.add(res_conv(x, s=2, filters=(256, 1024)))
        model.add(res_identity(x, filters=(256, 1024)))
        model.add(res_identity(x, filters=(256, 1024)))
        model.add(res_identity(x, filters=(256, 1024)))
        model.add(res_identity(x, filters=(256, 1024)))
        model.add(res_identity(x, filters=(256, 1024)))
        model.add(res_conv(x, s=2, filters=(512, 2048)))
```

```
        model.add(res_identity(x, filters=(512, 2048)))
        model.add(res_identity(x, filters=(512, 2048)))
        model.add(AveragePooling2D((2, 2), padding='same')(x))
        model.add(Flatten()(x))
        out = Dense(len(class_types), activation='softmax',
kernel_initializer='he_normal')(x))
        model = Model(inputs=input, outputs=out,
name='Resnet50')

        return model
```

3.2.1 Функція активації

У штучних нейронних мережах функція активації використовується для введення нелінійності в мережу. Без функцій активації незалежно від того, скільки шарів має нейронна мережа, вона поводилася б як одношаровий персептрон, тому що підсумовування цих шарів дало б вам просто ще одну лінійну функцію.

Функції активації допомагають мережі вчитися на помилках, поширювати їх у зворотному напрямку та регулювати ваги нейронів, щоб робити кращі прогнози в наступних проходах вперед. Вони дозволяють моделі фіксувати складні закономірності в даних шляхом перетворення сумарних зважених вхідних даних від вузла в активацію вузла або вихід для цього вхідного даних.

ReLU— одна з найпоширеніших функцій активації в моделях глибокого навчання. Функція повертає 0, якщо отримує негативне значення, але для будь-якого позитивного значення «x» повертає це значення назад.

Отже, можна резюмувати, що ReLU передає додатні значення як є, а від’ємні значення замінює нулем.

Переваги використання ReLU включають:

Простота обчислення: ReLU дуже простий у реалізації та обчислювально ефективний порівняно з сигмоідом і tanh, оскільки він не має жодних дорогих операцій (наприклад, експонент, множення тощо).

Нелінійність: незважаючи на свою простоту, ReLU забезпечує

необхідну нелінійність, необхідну для таких складних завдань, як розпізнавання зображень.

Вирішення проблеми зникнення градієнта: глибокі нейронні мережі, що використовують функції активації sigmoid або tanh, можуть страждати від проблеми зникнення градієнта, що означає, що мережа відмовляється навчатися далі або працює значно повільно, оскільки градієнти надто малі, щоб зробити будь-яку значну зміну вагових коефіцієнтів. . ReLU допомагає певною мірою пом’якшити цю проблему, оскільки його градієнт дорівнює 0 або 1, залежно від вхідних даних.

Однак ReLU не позбавлений проблем. Однією з проблем є те, що це може призвести до «мертвих нейронів», коли, коли нейрон отримує негативний вхід, він застряє та постійно видає 0. Щоб пом’якшити це, були запропоновані варіанти ReLU, такі як Leaky ReLU та Parametric ReLU, які дозволяють невеликі негативні значення, коли вхід менше 0.

3.2.2 Функція втрат

У машинному та глибокому навчанні функція втрат (або функція вартості) — це метод оцінки того, наскільки добре конкретний алгоритм моделює дані. Якщо прогнози надто сильно відрізняються від фактичних результатів, функція втрат видасть дуже велике число. Поступово, за допомогою певної функції оптимізації, функція втрат вчиться робити менше помилок.

Однією з поширених функцій втрат у машинному навчанні є середня квадратична помилка . Він в основному використовується в задачах регресії та є середнім квадратом різниці між оціненими значеннями та фактичним значенням. MSE є мірою якості оцінювача — він завжди невід’ємний, і значення, ближчі до нуля, кращі.

Математична формула для розрахунку MSE така:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Y_i — фактичне значення даних.

$\hat{Y}_i$— це прогнозоване значення даних, створене моделлю.

«n» — загальна кількість точок даних або екземплярів.

З точки зору його ролі в навчанні моделей машинного навчання, MSE використовується як функція втрат, яку навчальний алгоритм намагається мінімізувати шляхом коригування параметрів моделі. Алгоритми оптимізації, такі як градієнтний спуск, зазвичай використовуються для пошуку параметрів, які дають найменшу MSE.

MSE підкреслює більші помилки над меншими, оскільки помилки зведені в квадрат. Це може бути корисним у багатьох випадках, коли більші помилки мають більший вплив, але це також може призвести до надмірного впливу на модель викидів, якщо вони присутні в даних. Інші функції втрат, такі як середня абсолютна похибка або втрата Хубера, можна використовувати у випадках, коли ця властивість MSE є проблемою.

3.2.3 Deconvolution layer

Для того щоб отримати теплові карти з глибоких та low resolution feature можно примінити декілька deconvolution шарів. Дробово-поступові згортки, також відомі як транспоновані згортки або іноді деконволюції, є типом операцій, що використовуються в згорткових нейронних мережах (CNN), які по суті виконують операцію, протилежну звичайній згортці.

У звичайному згортковому шарі на виход отримуються зважена сума в кожному місці, що може призвести до зниження дискретизації, якщо крок більше 1. З іншого боку, дробово-шаговий згортковий шар або транспонований згортковий шар, підвищує дискретизацію вхідних даних для отримання більшого виходу.

Операція передбачає вставку нулів між вхідними значеннями для

збільшення розміру вхідних даних, а потім застосування звичайної згортки. Ефект полягає в тому, що вхідні значення розподіляються на більшій площі.

Термін «з дробовим кроком» походить від того факту, що ці шари можна вважати такими, що виконують операцію згортання з дробовим кроком. Звичайна згортка з кроком 2, наприклад, зменшує розмір вхідних даних у 2 рази, тому згортка з «кроком» 1/2 логічно збільшить розмір вхідних даних у 2 рази. На практиці це досягається описаною вище операцією вставки нуля та згортання.

Рисунок 3.3 – Теплова карта

3.3 Навчання моделі

Для навчання нейроннної мережі, як вже було зазначено, було обрано датасет COCO2017. Він складається з 3 частин: train, val, test. Перші два

використовуються власне для навчання та налаштування вагів моделі. Оскільки процесс навчання вимагає потужні обчислювальні потужності, було обрано Google Colab у якості середовища для виконная коду. На віртуальну машину завантажено потрібно датасети та аннотації для них.

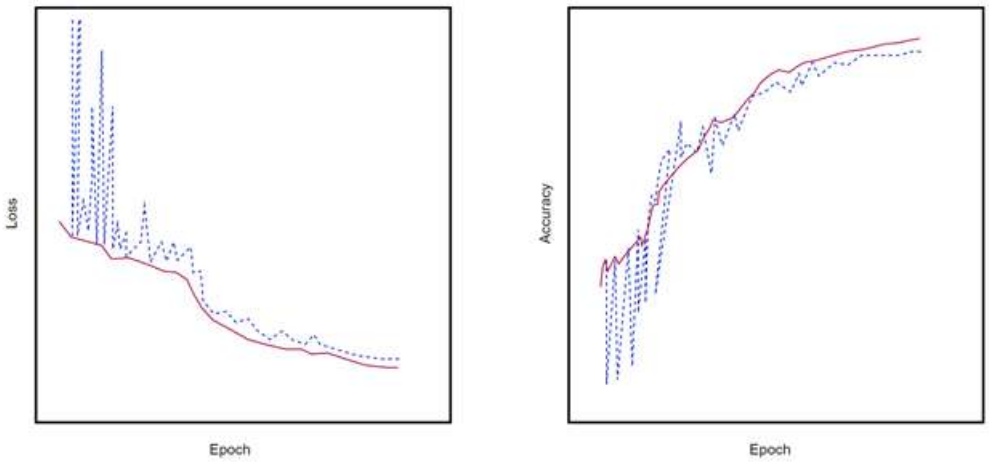

Рисунок 3.4 – Результати навчання

Після цього йде власне єтап навчання, коли модель знаходить оптимальні ваги для нейронів. Навчання виконується певну кількість ітерацій – епох, у межах однієї епохи кожне зображення опрацьовується один раз.

Отримані ваги зберігаються у окремому файлі, що дозволяє використовувати їх при наступному використанні моделі уже на тестовому датасеті та у реальних кейсах.

4 ВІЗУАЛІЗАЦІЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

4.1 Візуалізація

Для отримання вхідних данних та візуалізації роботи використано OpenCV. OpenCV (Open Source Computer Vision Library) — бібліотека комп’ютерного зору та машинного навчання з відкритим кодом. Вона була створена, щоб забезпечити загальну інфраструктуру для програм комп’ютерного зору та прискорити використання машинного сприйняття в комерційних продуктах.

OpenCV зосереджена на додатках у реальному часі та використовує переваги інструкцій MMX та SSE, якщо вони доступні. Вона широко використовується в академічних, дослідницьких та комерційних цілях для розробки передових програм обробки зображе Її інструменти дозволяють використовувати вебкамеру для отримання потоку вхідних данних та перетворювати їх у формат, який приймає модель. Після цього отриманий резульат роботи моделі дозволяє візуалізувати ключові точки на вхідних данних та намалювати зв’язки між ними

Лістинг 4.1 Функції для відображення ключових точок та їх зв’язків

```
def draw_keypoints(frame, keypoints, confidence_threshold):
    y, x, c = frame.shape
    shaped = np.squeeze(np.multiply(keypoints, [y, x, 1]))

    for key_point in shaped:
        ky, kx, kp_conf = key_point
        if kp_conf > confidence_threshold:
            cv2.circle(frame, (int(kx), int(ky)), 6, (0, 255,
0), -1)

def draw_connections(frame, keypoints, edges,
confidence_threshold):
    y, x, c = frame.shape
    shaped = np.squeeze(np.multiply(keypoints, [y, x, 1]))
```

```
    for edge, color in edges.items():
        p1, p2 = edge
        y1, x1, c1 = shaped[p1]
        y2, x2, c2 = shaped[p2]

        if (c1 > confidence_threshold) & (c2 >
confidence_threshold):
            cv2.line(frame, (int(x1), int(y1)), (int(x2),
int(y2)), (0, 0, 255), 4)
```

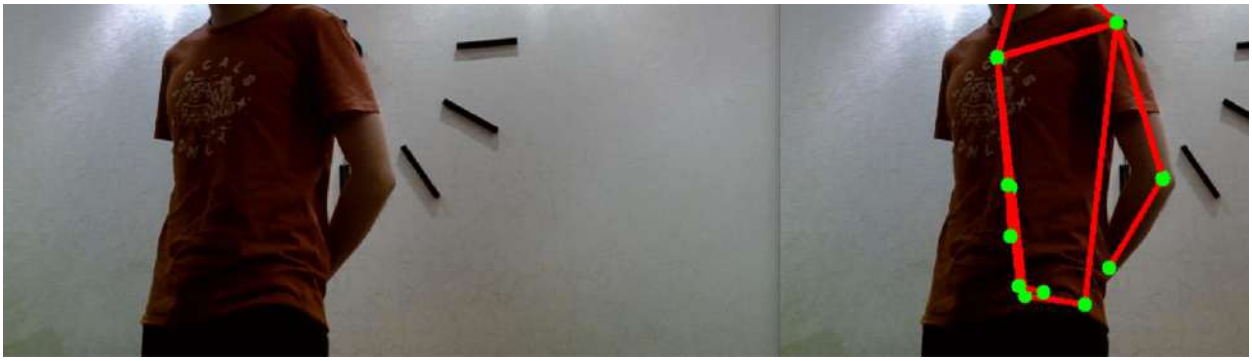

Рисунок 4.1 – Приклад вихідних данних моделі

Рисунок 4.2 – Приклад вихідних данних моделі

4.2 Аналіз результатів

Як можно побачити, результати є досить точними, хоча є проблема точності в умовах зниженого освітлення, а також при збільшенні кута між

людиною та камерою. У контексті оцінки пози людини Average Precision (AP) і Average Recall (AR) є двома показниками оцінки, які зазвичай використовуються для кількісної оцінки ефективності моделі HPE. Вони базуються на показниках точності та запам’ятовування, які широко використовуються в різних задачах комп’ютерного зору.

Точність. Точність вимірює частку справжніх позитивних прогнозів (правильне виявлення ключових точок) серед усіх позитивних прогнозів, зроблених моделлю. У HPE це оцінює, наскільки точна модель у виявленні ключових точок.

Recall вимірює частку справжніх позитивних прогнозів (правильне виявлення ключових точок) серед усіх фактичних позитивних випадків (основні ключові точки істинності) у наборі даних. У HPE він оцінює, наскільки добре модель визначає всі ключові точки, присутні на зображенні.

Щоб обчислити AP для HPE, ми спочатку обчислюємо точність на різних рівнях похибки локалізації ключових точок (порогові значення). Наприклад, прогнози моделі вважаються правильними, якщо відстань між прогнозованими та наземними ключовими точками істини нижче певного порогу (наприклад, відсоток довжини сегмента голови). Потім AP обчислюється як середнє значення точності за різними пороговими значеннями помилки. AP надає єдине значення, яке підсумовує продуктивність моделі з точки зору точності та стійкості до помилок локалізації ключових точок.

Аналогічно, AR для HPE обчислюється шляхом обчислення запам’ятовування на різних рівнях помилки локалізації ключових точок (порогові значення). Потім AR отримується як середнє значення відкликання за різними пороговими значеннями помилки. AR пропонує єдине значення, яке підсумовує продуктивність моделі з точки зору як здатності виявляти ключові точки, так і її стійкості до помилок локалізації.

Як AP, так і AR можна обчислити для окремих ключових точок або усереднити для всіх ключових точок, щоб отримати єдине значення, яке

представляє загальну продуктивність

модель HPE. Високі значення AP і AR вказують на те, що модель є точною та надійною у виявленні та локалізації ключових точок у вхідних зображеннях.

Таблиця 4.1 – Метрики моделі

Назва метрики	Значення
AP	70.4
AP .5	88.6
AP .75	77.8
AP (M)	67
AP (L)	76.9
AR	76.2
AR .5	93
AR .75	83
AR (M)	71.9
AR (L)	82.4

ВИСНОВКИ

Підсумовуючи, оцінка пози людини (HPE) є важливою та складною проблемою в комп'ютерному зорі з численними практичними застосуваннями в різних областях, таких як охорона здоров'я, спортивна аналітика, взаємодія людини з комп'ютером, робототехніка та створення контенту. Поява глибокого навчання, зокрема згорткових нейронних мереж (CNN), значно покращила сучасний рівень HPE, завдяки таким моделям, як Stacked Hourglass Networks, Simple Baselines і Vision Transformers, які демонструють надзвичайну продуктивність як з точки зору точності, так і надійності.

Незважаючи на прогрес, досягнутий за останні роки, HPE все ще стикається з кількома проблемами, зокрема з проблемами оклюзії, варіаціями зовнішнього вигляду, різними позами та кількома людьми в одній сцені. Ці виклики спонукають до постійної розробки нових моделей, методів і архітектур, які можуть вирішити ці труднощі та ще більше просувати сферу HPE.

Майбутні напрямки досліджень HPE можуть включати:

- розробка більш ефективних і легких моделей для забезпечення оцінки пози в режимі реального часу для додатків, які вимагають обробки з низькою затримкою, наприклад робототехніки та доповненої реальності;
- дослідження підходів до неконтрольованого або напівконтрольованого навчання, щоб зменшити залежність від великомасштабних анотованих наборів даних, створення яких потребує багато часу та коштує;
- вивчення нових архітектур і технік, які покращують здатність моделі справлятися з оклюзіями, змінами точки зору та різними позами;
- інтеграція hpe з іншими завданнями, такими як виявлення об'єктів, розпізнавання дій і розуміння сцени, для розробки цілісних систем розуміння

для різноманітних програм;

- покращення можливостей інтерпретації та пояснення моделей hpe, щоб краще зрозуміти їхні прогнози та підвищити довіру до їхніх результатів, особливо в чутливих програмах, таких як охорона здоров’я.

Таким чином, HPE є важливою галуззю комп’ютерного зору, яка швидко розвивається, з моделями глибокого навчання, які демонструють значний прогрес за останні роки. Вирішуючи поточні виклики та досліджуючи нові напрямки досліджень, наукове співтовариство може продовжувати розвивати HPE.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. K. Wang, R. Zhao, and Q. Ji, “Human computer interaction with head pose, eye gaze and body gestures,” in 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018). IEEE, 2018

2. T. B. Moeslund, A. Hilton, and V. Kruger, “A survey of advances ¨ in vision-based human motion capture and analysis,” Computer vision and image understanding, vol. 104, no. 2-3

3. K. Wang, R. Zhao, and Q. Ji, “Human computer interaction with head pose, eye gaze and body gestures,” in 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018). IEEE, 2018

4. T. B. Moeslund, A. Hilton, and V. Kruger, “A survey of advances ¨ in vision-based human motion capture and analysis,” Computer vision and image understanding, vol. 104, no. 2-3

5. S. Jin, L. Xu, J. Xu, C. Wang, W. Liu, C. Qian, W. Ouyang, and P. Luo, “Whole-body human pose estimation in the wild,” in European Conference on Computer Vision, 2020

6. Krizhevsky, A., Sutskever, I., Hinton, G. E. (2017), "ImageNet classification with deep convolutional neural networks", Communications of the ACM, Vol. 60, No. 6

7. Alexander Toshev, Christian Szegedy (2014), “DeepPose: Human Pose Estimation via Deep Neural Networks”, DOI: https://doi.org/10.1109/CVPR.2014.214

8. Alejandro Newell, Kaiyu Yang, Jia Deng (2016), “Stacked Hourglass Networks for Human Pose Estimation”, DOI: https://doi.org/10.48550/arXiv.1603.06937

9. Bin Xiao, Haiping Wu, Yichen Wei (2018), “Simple Baselines for Human Pose Estimation and Tracking”, DOI: https://doi.org/10.48550/arXiv.1804.06208

10. Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby (2020), “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”, DOI: https://doi.org/10.48550/arXiv.2010.11929

11. J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler, “Joint training of a convolutional network and a graphical model for human pose estimation,” in Conference on Neural Information Processing Systems (NeurIPS), 2014

12. Girdhar, R., Gkioxari, G., Torresani, L., Paluri, M., Tran, D., “Detect-and-track: Efficient pose estimation in videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition”

13. Andriluka, M., Pishchulin, L., Gehler, P., Schiele, B. 20(14) ”2d human pose estimation: New benchmark and state of the art analysis. In: IEEE Conference on Computer Vision and Pattern Recognition “

14. Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun (2015), "Deep Residual Learning for Image Recognition", DOI: https://doi.org/10.48550/arXiv.1512.03385

15. Huang, Gao; Liu, Zhuang; Van Der Maaten, Laurens; Weinberger, Kilian Q. (2017). Densely Connected Convolutional Networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), doi:10.1109/CVPR.2017.243

16. Ioffe, S., Szegedy, C., “Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning.”

17. Angjoo Kanazawa, Michael J. Black, David W. Jacobs, and Jitendra Malik, (2018) “End-to-end Recovery of Human Shape and Pose”, DOI: https://doi.org/10.48550/arXiv.1712.06584

18. D. Vyshnivskyi, O. Liashenko, N. Yeromina, Human Pose Estimation system using Deep Learning algorithms, Conrtol, Navigation and Communication Systems Issue 2(72), 2023

19. Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger, Densely Connected Convolutional Networks, 2017 https://arxiv.org/pdf/1608.06993v5.pdf

20. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. ICLR (2015)

21. COCO2017 dataset, available at https://cocodataset.org/?ref=blog.roboflow.com#download