КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет комп'ютерних наук та кібернетики
Кафедра математичної інформатики

Кваліфікаційна робота
на здобуття ступеня бакалавра

за спеціальністю 122 Комп'ютерні науки

на тему:

ВИЗНАЧЕННЯ 3D ПОЗИ ЛЮДИНИ НА ЗОБРАЖЕННІ

Виконала студентка 4 курсу
Магдисюк Вероніка Андріївна _____
(підпис)

Науковий керівник:
Професор, доктор фізико-математичних наук
Терещенко Василь Миколайович _____
(підпис)

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

Студент _____
(підпис)

Роботу розглянуто й допущено до захисту
на засіданні кафедри математичної інформатики
«___» _____ 2022 р.,
протокол № ___
Завідувач кафедри

М.В. Терещенко _____
(підпис)

Київ – 2022

РЕФЕРАТ

Обсяг роботи 47 сторінок, 17 ілюстрацій, 4 таблиці, 23 джерела посилань, 1 фрагмент коду.

MACHINE LEARNING, DEEP LEARNING, ARTIFICIAL NEURAL NETWORK, COMPUTER VISION, POSE ESTIMATION, МОДЕЛЬ МАШИННОГО НАВЧАННЯ, ВИЗНАЧЕННЯ ПОЗИ ЛЮДИНИ З ЗОБРАЖЕННЯ.

Об’єктом розробки є алгоритми та методи визначення поз людей у двовимірному та трьохвимірному просторі, наступна обробка та візуалізація даних. Предметом роботи є програмні застосунки для визначення 2D та 3D поз людини або групи людей з заданого користувачем зображення.

Метою роботи є створення програмного засобу для визначення 2D та 3D поз людини або групи людей та аналіз архітектури та роботи низки моделей машинного навчання для визначення поз людини.

Методи розроблення: комп’ютерне моделювання, Machine Learning, Computer Vision, обробка зображень та послідовності зображень, проектування моделей машинного навчання, аналіз вихідних даних.

Інструменти розроблення: безкоштовне, вільно поширюване онлайн-середовище розробки та розмітки Google Colab, мова програмування Python, розширення та бібліотеки Tensorflow, PyTorch, MediaPipe, GluonCV та інші.

Результати роботи: виконано загальний огляд та збір інформації про Machine Learning та Computer Vision, визначено специфікацію та методи розв’язання задачі визначення пози людини, проведено дослідження різних підходів та архітектур моделей машинного

навчання, реалізовано кілька моделей та програмні застосунки для їх використання, проведено аналіз результатів роботи моделей.

За методами розробки програмний засіб тісно пов’язаний з задачами визначення та переносу пози людини у графічний простір з подальшою обробкою.

Результуючий продукт може використовуватися як основа для застосунків, що пов’язані з фітнесом, медициною, охороною та відеоспостереженням, а також – у захопленні руху, пов’язаному з мультиплікацією та комп’ютерними іграми.

ЗМІСТ

ВСТУП

Оцінка сучасного стану об’єкта дослідження. Pose Estimation (або Keypoint Detection) – це задача машинного навчання, яка полягає у визначенні позиції людини або об’єкта за допомогою прогнозування і встановлення позицій ключових точок: рук, колін, голови тощо.

Актуальність роботи та підстави для її виконання. Моделі для визначення пози людини широко застосовуються у застосунках для фітнесу, відеоспостереження й охорони, у сфері 3D анімації та моделювання. Користувач повинен мати доступ до легковагових і точних моделей, щоб отримати точні результати, не витрачаючи на це багато часу і обчислювальних ресурсів.

Мета й завдання роботи. Метою курсової роботи є розроблення програмного застосунку для визначення пози людини або групи людей. Для виконання завдання потрібно:

а) опрацювати теоретичну складову Machine Learning та Computer Vision, визначити методи та принципи побудови моделей машинного навчання.

б) Провести огляд архітектури та підходів до побудови моделей, визначити найбільш ефективні методи та аспекти.

в) Провести огляд існуючих моделей для визначення пози людини.

г) Реалізувати застосунок для визначення пози людини або групи людей з вхідного зображення або послідовності зображень.

д) Провести тестування застосунку та визначити точність поз.

е) Провести аналіз низки моделей, порівняти результати та зробити висновки.

Об'єкт, методи й засоби дослідження і розробки. Об’єктом дослідження є зображення або відеофайл. Методами розробки є методи машинного навчання та проектування моделей визначення 3D поз людини або

людей. Засобами виконання є мова програмування Python, онлайн-середовище програмування Google Colab, пакети та бібліотеки для Machine Learning: Tensorflow, PyTorch, MediaPipe, GluonCV та інші.

Можливі сфери застосування. Розроблений застосунок може використовуватися для визначення поз людей з фото- та відеофайлів або у режимі реального часу. Також застосунок може стати основою для створення додатків, пов’язаних з рухом людей у спорті та повсякденному житті.

Взаємозв’язок з іншими роботами. Вихідні дані про визначені пози можуть бути використані для ідентифікації дій або експортовані у середовище 3D моделювання та анімації (наприклад, у Blender). Таких реалізованих застосунків є небагато, і вони зазвичай є платними.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ЗАСАДИ МАШИННОГО НАВЧАННЯ ТА ВИЗНАЧЕННЯ ПОЗИ ЛЮДИНИ

1.1 Теорія машинного навчання

Машинне навчання (Machine Learning - ML) — це область досліджень, присвячена розумінню та створенню методів, які «навчаються», тобто методів, які використовують дані для підвищення продуктивності певного набору завдань. Машинне навчання розглядається як частина штучного інтелекту. Основним принципом вирішення задач за допомогою Machine Learning є навчання за рахунок застосування рішень множини подібних задач. Алгоритми машинного навчання будують модель на основі вибіркових даних, відомих як навчальні дані, щоб робити прогнози або приймати рішення без явного програмування на це. [1]

Machine Learning тісно пов’язане з математичною статистикою та аналізом даних. Моделі машинного навчання обробляють масиви вхідних даних (тренувальний набір), які представляються вже відомими рішеннями – числовими, символьними, зображеннями тощо, та виділяють особливі закономірності та елементи (features).

Свій початок машинне навчання бере з галузі штучного інтелекту (Artificial Intelligence - AI). Під час розвитку AI було виявлено проблему навчання з вже існуючих даних. Дослідники підійшли до цього питання за допомогою символьних методів та нейронних мереж.[2]

Машинне навчання, реорганізоване як окрема галузь, почало процвітати в 1990-х роках. Його фокусом стало вирішення саме практичних завдань, а основними інструментами стали методи та моделі, запозичені зі статистики, логіки та теорії ймовірностей.

Основною метою системи є здатність точно виконувати нові, небачені приклади/завдання після того, як вона обробила набір навчальних даних.

Навчальні дані (тренувальний датасет) збираються дослідниками і класифікуються для виконання завдання. Система повинна побудувати загальну модель щодо цього простору, яка дозволить їй створювати достатньо точні прогнози в нових випадках. Перевірка точності відбувається за допомогою тестових даних, коли фактичні результати звіряються з практичними.

Обчислювальний аналіз алгоритмів машинного навчання та їх продуктивності – це галузь теоретичної інформатики, яка відповідає за аналіз коректності та продуктивності роботи начальних машин. Зазвичай, продуктивність та коректність задаються ймовірнісними межами.

Для найкращої продуктивності в контексті узагальнення складність гіпотези повинна відповідати складності функції, що лежить в основі даних. Якщо гіпотеза менш складна, ніж функція, то модель недостатньо підігнала дані (underfit). Але якщо гіпотеза занадто складна, то модель підганяла дані занадто сильно (overfit). Коригуючи модель, можна досягнути оптимальних результатів. Типові способи регуляризації або запобігання overfit включають: штрафні параметри під час тренування (наприклад, зниження ваги) або обрізання зв'язку (пропущені з'єднання, відключення тощо). На додаток до меж продуктивності, теоретики навчання вивчають часову складність і доцільність навчання.[2]

Підходи машинного навчання традиційно поділяються на три великі категорії залежно від характеру сигналу або зворотного зв’язку:

а) Навчання з учителем (Supervised Learning): комп’ютер надає приклади вхідних даних та їх бажаних результатів. Мета полягає в тому, щоб вивчити загальне правило, яке відображає вхідні дані та вихідні дані.

б) Навчання без учителя (Unsupervised Learning): алгоритму навчання не присвоюються ярлики, тому він сам знаходить структуру у вхідних даних. Навчання без нагляду може бути самоціллю (виявлення прихованих закономірностей у даних) або засобом досягнення мети (особливе навчання).

в) Навчання з підкріпленням (Reinforcement Learning): комп'ютерна програма взаємодіє з динамічним середовищем, в якому вона повинна виконувати певну мету. У міру навігації в проблемному просторі програма отримує зворотній зв'язок, аналогічний винагороді, яку вона намагається максимізувати.

Програмні застосунки, що вирішують задачі Machine Learning, широко використовуються у низці галузей промисловості, охороні здоров'я, фінансовій справі, кібер- та безпеці тощо.

Штучні нейронні мережі (Artificial Neural Network - ANN), які зазвичай називають просто нейронними мережами (Neural Network - NN), є обчислювальними системами, що засновані на сукупності пов'язаних одиниць або вузлів, які називаються штучними нейронами. Штучні нейроні отримують сигнали, потім обробляють їх і можуть передавати сигнали до інших нейронів. Сигнал при з'єднанні є дійсним числом, а вихід кожного нейрона обчислюється деякою нелінійною функцією суми його вхідних даних. З'єднання називаються ребрами. Нейрони та ребра зазвичай мають вагу, які коригуються під час навчання. Вага збільшує або зменшує силу сигналу при з'єднанні. Як правило, нейрони об'єднуються в шари. Різні шари можуть виконувати різні перетворення на своїх входах. Сигнали поширюються від першого шару (вхідного шару) до останнього шару (вихідного шару), часто після багаторазового проходження шарів.

NN були натхненні обробкою інформації та розподіленими вузлами зв'язку в біологічних системах. NN мають різні відмінності від біологічного мозку. Зокрема, штучні нейронні мережі, як правило, статичні та символічні,

тоді як біологічний мозок більшості живих організмів є динамічним і аналоговим.[3]

Функціональне навчання мотивується тим, що завдання машинного навчання, такі як класифікація, часто вимагають введення, яке математично та обчислювально зручно обробляти. Проте реальні дані, такі як зображення, відео та сенсорні дані, не піддаються спробам алгоритмічного визначення конкретних ознак. Альтернативою є виявлення таких ознак або уявлень шляхом перевірки, не покладаючись на явні алгоритми.

Deep Learning є частиною більш широкого сімейства методів Machine Learning, заснованих на NN з репрезентативним навчанням. Архітектури глибинного навчання застосовуються в таких областях, як комп'ютерний зір, розпізнавання мовлення, обробка природної мови, машинний переклад, медичний аналіз, кліматологія, програми настільних ігор тощо. Деякі застосунки навіть перевищили результати експертів.[4]

DL відноситься до використання кількох шарів у мережі. Ранні роботи показали, що лінійний сприймач не може бути універсальним класифікатором, але мережа з неполіноміальною функцією активації з одним прихованим шаром необмеженої ширини – може. Deep Learning — це сучасна варіація, яка стосується необмеженої кількості шарів обмеженого розміру, що дозволяє практичне застосування та оптимізовану реалізацію, зберігаючи теоретичну універсальність.

Згорткова нейронна мережа (Convolutional Neural Network – CNN) — це клас штучної нейронної мережі, яка найчастіше використовується для аналізу візуальних зображень. Вони мають застосування в розпізнаванні зображень і відео, системах рекомендацій, класифікації, сегментації зображень, аналізі медичних зображень тощо.

CNN є регуляризованими версіями багатошарових сприймачів. Багатошарові сприймачі це зазвичай повністю пов’язані мережі, тобто кожен нейрон одного шару з’єднаний з усіма нейронами наступного шару. CNN запобігають overfit, використовуючи інший підхід до регуляризації: вони використовують переваги ієрархічного шаблону в даних і збирають шаблони зростаючої складності, використовуючи менші та простіші візерунки, вибиті в їх фільтрах.[5]

1.2 Комп’ютерний зір та його задачі

Комп’ютерний зір (Computer Vision) – це область машинного навчання, яка полягає у навчанні комп’ютерів та систем інтерпретувати та розуміти візуальний світ. Використовуючи цифрові зображення з камер і відео, а також моделі глибокого навчання, машини можуть точно ідентифікувати та класифікувати об’єкти, а потім за потреби приймати рішення з отриманих даних.

Завдання комп’ютерного зору включають методи отримання, обробки та аналізу та розуміння даних у форматі зображень та послідовностей зображень. Розуміння в цьому контексті означає перетворення візуальних образів в цифрові дані за допомогою моделей машинного навчання. У подальшому, ці дані обробляються комп’ютерами та системами і можуть використовуватися для прийняття рішень. [6]

Наукова дисципліна комп’ютерного зору займається теорією штучних систем, які вилучають інформацію із зображень. Дані зображення можуть приймати різні форми, наприклад, відеопослідовності, перегляди з кількох камер, багатовимірні дані з 3D-сканера або медичного скануючого пристрою. Технологічна дисципліна комп’ютерного зору прагне застосувати свої теорії та моделі до побудови систем комп’ютерного зору.

Задачі комп’ютерного зору:

а) 3D реконструкція (Scene Reconstruction): захоплення об’єктів з фото чи відео, їх форм та зовнішнього виду.

б) виявлення об’єктів (Object Detection): виявлення та визначення класу, якому належить об’єкт (людина, тварина, машина тощо).

в) відстеження відео (Video Tracking): виявлення та відстеження рухомого об’єкта (або кількох об’єктів) у часі.

г) розпізнавання об’єктів (Object Recognition): пошук та ідентифікація об’єктів на зображенні або відео.

д) Оптичне розпізнавання символів (OCR): відновлення тексту та цифр у печатному форматі з зображення.

е) оцінку тривимірної пози (3D Pose Estimation): виявлення та встановлення трансформації об’єкта або пози людини з заданого зображення чи 3D сканування.

ж) оцінку руху (Motion Estimation): процес визначення векторів руху, які описують перехід від одного 2D зображення до іншого. Це використовується при стисканні відеофайлів.

з) візуальне обслуговування (Visual Servoing): управління роботами за допомогою інформації з візуальних сенсорів.

и) моделювання 3D сцени (3D Scene Modeling): створення 3D об’єктів у графічному просторі з вхідних візуальних даних.

к) відновлення зображення (Image Restoration): відновлення якості зображень, які є пошкодженими, розмитими або мають багато шуму.

Організація системи комп’ютерного зору сильно залежить від застосування. Деякі системи є окремими додатками, які вирішують конкретну задачу вимірювання або виявлення, тоді як інші складають підсистему більшої конструкції. Конкретна реалізація системи комп’ютерного зору також залежить від того, чи визначена її

функціональність заздалегідь, чи якусь її частину можна вивчити чи змінити під час роботи. Багато функцій є унікальними для програми. Проте є типові функції, які є в багатьох системах комп’ютерного зору:

а) Отримання зображення – цифрове зображення створюється одним або кількома датчиками зображення, які, окрім різних типів світлочутливих камер, включають датчики дальності, томографічні пристрої, радари, ультразвукові камери тощо. Залежно від типу датчика, отримані дані зображення є звичайним двовимірним зображенням, тривимірним об’єктом або послідовністю зображень.

б) Попередня обробка – перш ніж метод комп’ютерного зору можна буде застосувати до даних зображень, щоб отримати певну частину інформації, зазвичай необхідно обробити дані, щоб переконатися, що вони задовольняють певним припущенням, які випливають із методу.

1) Повторна вибірка.

2) Зниження шуму.

3) Підвищення контрастності.

4) Масштабування.

в) Вилучення об’єктів – функції зображення різного рівня складності витягуються з даних зображення. Типовими прикладами таких функцій є:

1) Лінії, краї та хребти.

2) Локалізовані точки інтересу.

г) Виявлення/сегментація – на цьому етапі обробки приймається рішення про те, які точки або ділянки зображення є релевантними для подальшої обробки. Приклади:

1) Вибір конкретного набору цікавих точок.

2) Сегментація однієї або кількох областей зображення, які містять конкретний об’єкт, що цікавить.

3) Сегментація зображення на архітектуру вкладеної сцени, що включає передній план, групи об’єктів, окремі об’єкти або частини помітного об’єкта.

4) Сегментація або спільна сегментація одного або кількох відео на серію масок переднього плану для кожного кадру.

д) Високорівнева обробка – на цьому кроці вхідним сигналом зазвичай є невеликий набір даних, наприклад набір точок або область зображення, яка, як передбачається, містить певний об’єкт. Решта обробки стосується, наприклад:

1) Перевірка того, що дані задовольняють припущенням на основі моделі та конкретної програми.

2) Оцінка специфічних для програми параметрів, таких як поза або розмір об'єкта.

3) Розпізнавання зображень – класифікація виявленого об’єкта за різними категоріями.

4) Реєстрація зображення – порівняння та поєднання двох різних видів одного об’єкта.

е) Прийняття остаточного рішення.

1.3 Задача визначення пози

Pose Estimation — це завдання комп’ютерного зору, яке полягає у визначенні позиції людини в графічному форматі. Ця техніка зазвичай застосовується для прогнозування місцезнаходження та повороту частин тіла або положення суглобів людини.

Pose Estimation визначає та класифікує пози частин тіла і суглобів людини на зображеннях або відео. Загалом, методика, заснована на моделі, використовується для представлення та визначення пози людського тіла в 2D та 3D просторах. Pose Estimation це спосіб отримати набір координат, які описують позу людини шляхом визначення

суглобів та частин тіла, таких як зап’ястя, плечі, коліна, очі, вуха, щиколотки та руки.

Моделі мають різні підходи до визначення частин тіл, виявлення позицій, побудови скелету пози. Вони сприймають на вхід зображення або послідовність зображень, визначають положення частин тіла, будують взаємозв’язки між суглобами, повертають їх як вихідні дані та точність визначення кожної точки.

Стандартна модель глибокого навчання для вирішення задачі Human Pose Estimation має в основі CNN. Архітектура CNN складається з двох ключових компонентів — кодера (оцінювач) і декодера (детектор).

Кодер витягує конкретні функції, які називаються ключовими точками, з вхідного зображення. Кількість ключових точок залежить від підходу і зазвичай коливається від 17 до 33. Прикладами ключових точок можуть бути лівий лікоть, праве коліно або основа шиї. Кодер запускає вхід через набір блоків звуження згортки і витягує ознаки всіх людей на зображенні.

Потім декодер створює теплову карту ймовірності ключової точки. Це допомагає уточнити прогнози кодера та оцінити, наскільки ймовірно, що вилучені об’єкти будуть розташовані в позначених областях зображення.

Таким чином, CNN відображає попередньо оцінені ключові точки на теплову карту та виводить ті з найвищою ймовірністю. Потім він групує пов’язані ключові точки в структуру тіла, схожу на скелет. [7]

РОЗДІЛ 2. МАШИННЕ НАВЧАННЯ ЯК ПІДХІД ДО ВИРІШЕННЯ ЗАДАЧ КОМП'ЮТЕРНОГО ЗОРУ

2.1 Проектування моделей машинного навчання

Моделі для визначення поз можуть мати різні підходи, структури та архітектури. У цьому підрозділі буде розглянуто їх класифікації та характеристики. [8]

Основною класифікацією моделей є розмірність отриманих поз:

а) 2D моделі визначають знаходження ключових точок відносно заданих даних у 2D (зображення або відео). Результати роботи представляються як масив координат кожної точки на осях X та Y.

б) 3D моделі, окрім визначення положення ключових точок у X та Y, визначають близькість точки до камери і тим самим – положення по осі Z.

2D Pose Estimation є більш швидким та менш ресурсозатратним, адже просто знаходить частини тіла відносно зображення. 3D Pose Estimation зважає на положення камери, а тому спроектувати трьохвимірну позу складніше.

Іншою базовою характеристику моделі є здатність до виявлення певної кількості поз:

а) Single Person Pose Estimation – вони ж – моделі, що визначають одну людину та її позу. У випадку кількох людей на зображенні, визначення пози проводиться для першої або найкраще виявленої людини.

б) Multi Person Pose Estimation - вони ж – моделі, що визначають кілька людей та їх пози. Зазвичай, обмеження на кількість людей нема, але більша кількість людей та поз займає більше часу та ресурсів.

Класифікація за основою моделей, тобто, на яких геометричних об’єктах вони базовані:

а) Скелетна модель (кінематична) – найширше застосована, вона визначається набором ключових точок, які визначають позицію суглобів та частин тіла. Використовується для 2D та 3D Pose Estimation.

б) Контурна модель (планарна) – складається з контуру тіла та широт частин тіла та суглобів. Представляється як форма людського тіла у 2D, де частини тіла описані прямокутниками та границями.

в) Об’ємна модель – представляє собою 3D модель людського тіла.

Методи визначення поз можна розділити на 2 основні групи: «знизу вверх» та «зверху вниз». Методи першої групи визначають спочатку суглоби або частини тіла, а потім за отриманими положеннями будують позу. Методи другої групи визначають тіло та обмежувальні рамки частин тіла, у яких ведеться пошук самих частин та кінцівок.

Як демонструє підхід, методи «знизу вверх» краще визначають складні та групові пози, але зокрема можуть видати похибкові результати через те, що вони не використовують дані про структуру тіл та оточення. Ці дані широко застосовуються у методах «зверху вниз», що запобігає таким помилкам. Але ці методи погано визначають складні та групові пози, адже важливу роль у визначенні грає детектор тіл, у випадку некоректного спрацювання якого, некоректно визначаються і пози. [9]

Різні моделі часто працюють з різними видами скелетів людини та координатними системами. Зазвичай, скелет складається з 17 чи 18 ключових точок, таких як, ноги, кисті рук, плечі та ін. Але, наприклад, модель BlazePose працює з 33 точками. Про окремі моделі і їх особливості мова піде у розділах 2.3, 3.1 та 3.2.

2.1.1 Архітектура та шари

Однією з найбільш уживаних моделей для Pose Estimation є Mask R-CNN. Вона бере за основу Fast R-CNN, яка в свою чергу є модифікацією R-CNN. Тому, по порядку.[10]

Convolution Neural Network вже згадувалася у розділі 1.1. Region-Based CNN (R-CNN) – клас нейронних моделей, які визначають обмежувальні рамки для вхідних даних та обробляють їх. Faster R-CNN – це покращена версія архітектури R-CNN з двома етапами:

а) Region Proposal Network (RPN). RPN — це нейронна мережа, яка визначає кілька об’єктів на заданому зображенні або регіоні.

б) Fast R-CNN визначає та вилучає особливості з використанням RoIPool (об’єднання регіонів інтересів) з кожного поля-кандидата і виконує класифікацію та регресію обмеженої рамки. RoIPool — це операція для вилучення невеликої карти об’єктів.

Mask R-CNN була створена за допомогою Faster R-CNN. У той час як Faster R-CNN має 2 виходи для кожного об’єкта-кандидата, мітку класу та зміщення обмежувальної рамки, Mask R-CNN є додаванням третьої гілки, яка виводить маску об’єкта. Вихід додаткової маски відрізняється від виводів класів і блоків, що вимагає вилучення набагато більш тонкого просторового макета об’єкта.

Mask R-CNN є розширенням Faster R-CNN і працює шляхом додавання гілки для передбачення маски об'єкта (область інтересу) паралельно з існуючою гілкою для розпізнавання обмежувального прямокутника.

OpenPose і PersonLab (також відомі як PoseNet) є варіантами архітектури кодера-декодера з особливостями. На додаток до виведення теплових карт, модель також виводить уточнення теплових карт у вигляді коротких, середніх і далеких зміщень. Теплові карти визначають широкі області зображення, де, ймовірно, буде знайдена ключова точка,

а зміщення направляють прогнози в межах області до більш точного остаточного прогнозу.

2.1.2 Тренування та валідація

Спроектована модель повинна пройти тренування та тестування на коректність. Це реалізується за допомогою підготовлених для визначеного завдання наборів даних (датасетів). Датасети збираються групами дослідників та обробляються згідно задачі: класифікуються, визначаються обмежувальні рамки та маски, положення ключових точок тощо. Ці дані використовуються для навчання мережі, її тестування та валідації – якість створеної моделі визначається часом виконання задачі та точністю.

Майже найбільшим сайтом з наборами даних є Kaggle. Тим паче, Google Colab підтримує швидке та зручне завантаження даних звідти.

Набір даних MPII Human Pose включає близько 25 тис. зображень, які містять понад 40 тис. людей з визначеними суглобами та частинами тіла. Загалом датасет охоплює 410 видів діяльності людини, і кожне зображення було отримано з відео з Youtube і має визначення цієї діяльності.[11]

COCO (Microsoft Common Objects in Context) – це широкомасштабний набір даних для виявлення, сегментації, виявлення ключових точок і субтитрів. Набір даних складається з 328 тисяч зображень. Перша версія набору даних MS COCO була випущена в 2014 році. Він містив 164 тис. зображень, розділених на навчальні (83 тис.), перевірочні (41 тис.) і тестові (41 тис.). У 2015 році було випущено додатковий тестовий набір із 81 тис. зображень, включаючи всі попередні тестові зображення та 40 тис. нових зображень. У 2017 році розподіл навчання/підтвердження було змінено з 83тис./41тис. на 118тис./5тис. У новій версії використовуються ті самі зображення та анотації. [12]

Набір даних містить анотації для:

а) виявлення об'єктів: обмежувальні рамки та маски сегментації для кожного екземпляра з 80 категоріями об'єктів.

б) субтитрів: описи зображень природною мовою.

в) виявлення ключових точок: містить понад 200 тис. зображень і 250 тис. осіб, позначених ключовими точками (17 можливих ключових точок).

г) сегментації зображень – маски сегментації з 91 категорією об’єктів.

д) паноптичної сегментації: повна сегментація сцени з 80 категоріями об’єктів (наприклад, людина, велосипед, слон) і підмножиною з 91 категорій речей (трава, небо, дорога).

е) DensePose: понад 39 тис. зображень і 56 тис. екземплярів людей, позначених анотаціями DensePose.

Leeds Sports Pose Dataset (LSP) – це набір даних, що включає в себе 2000 зображень з визначеними позами людей у категоріях спортивних занять. Кожне зображення було анотовано 14 ключовими точками. Лівий і правий суглоби послідовно позначаються з точки зору, орієнтованої на людину.[13]

Розширений варіант цього датасету - Leeds Sports Pose Extended Training Dataset, який містить 10 000 зображень, а також поз, які важко оцінити.

2.2 Огляд існуючих рішень та систем комп’ютерного зору

Google, напевно, є чи не найрозвинутішою компанією, яка досліджує та удосконалює інтернет та комп’ютерні технології. Серед галузей розробок – також і Machine Learning та Computer Vision. Потужності та дослідження Google призвели до появи низки доступних рішень та інструментів для будь-яких користувачів.

Зручним способом взаємодії з потужним інструментарієм є використання API, тому компанією було розроблено Vision API та AutoML Vision. [14]

Vision API дозволяє користувачу швидко та надійно провести аналіз зображення, серед функціоналу – OCR, Object Detection та Recognition, Face Detection та Recognition, визначення категорій, контексту та безпечності. Демонстрація роботи доступна у браузері.

AutoML Vision – це інструмент та середовище для тренування моделей Computer Vision. Він дозволяє завантажити датасет для тренування або створити власний, тренувати та валідувати заготовлену модель та використовувати її для подальшого створення додатків.

Компанія Intel, окрім комп'ютерних складових, також розробляє програмні рішення, зокрема – і для Computer Vision. OpenVINO Toolkit – одне із них. Це інструментарій для розробки програм штучного інтелекту та комп'ютерного зору, які працюють на різних типах обладнання – для розумних камер, засобах промислового виробництва, робототехніки, інтелектуальних серверів, транспорту тощо. [15]

NVIDIA – провідний розробник відеокарт та застосунків для Computer Vision. Ними постійно проводяться дослідження та покращення власних бібліотек, фреймворків та застосунків – для автомобілів, медицини, комп'ютерних ігор, відеообробки та ін. Крім того, NVIDIA пропонує рішення і для задачі Pose Estimation. [16]

2.3 Фреймворки, бібліотеки та моделі для Python

Для користувачів та дослідників в області AI та ML важливо мати доступ до потужної обчислювальної техніки та ресурсів. Google Colab – це хмарне середовище програмування мовою Python, яке взяло за основу розмітку Jupyter. Colab надає користувачам інтерфейс для виконання обчислень на

потужних серверах. Тому це середовище ідеальне для виконання задач машинного навчання.

Мова Python заточена під виконання масивних обчислень, складних задач та алгоритмів. Тому саме Python часто стає інструментом для моделювання та реалізації мереж машинного навчання.

Одним із найбільш використовуваних платформ для розробки моделей машинного навчання та комп'ютерного зору є Tensorflow. Tensorflow пропонує функціонал для створення моделей машинного навчання з нуля – моделювання шарів, тренування, підгонка, валідація та розгортання.

Tensorflow пропонує рішення і для Pose Estimation – 2D модель MoveNet з двома варіантами: швидшим (Lightning) і якіснішим (Thunder). Як продемонструвало тестування, коректність моделі Thunder перевищила 70% (див. таблицю 1) [17].

Модель	**Розмір (MB)**	**Точність**	**Затримка (ms)**		
			Pixel 5 - CPU 4 threads	**Pixel 5 - GPU**	**Raspberry Pi 4 - CPU 4 threads**
MoveNet.Thunder (FP16 quantized)	12.6	72.0	155	45	594
MoveNet.Thunder (INT8 quantized)	7.1	68.9	100	52	251
MoveNet.Lightning (FP16 quantized)	4.8	63.0	60	25	186
MoveNet.Lightning (INT8 quantized)	2.9	57.4	52	28	95
PoseNet(MobileNetV1 backbone, FP32)	13.3	45.6	80	40	338

Таблиця 1 - точність та затримка роботи моделей MoveNet.

MoveNet — це модель «знизу-догори», яка використовує теплові карти для точної локалізації ключових точок людини. Архітектура (див. рисунок 2.1) складається з двох компонентів: для визначення особливостей і для передбачення поз.

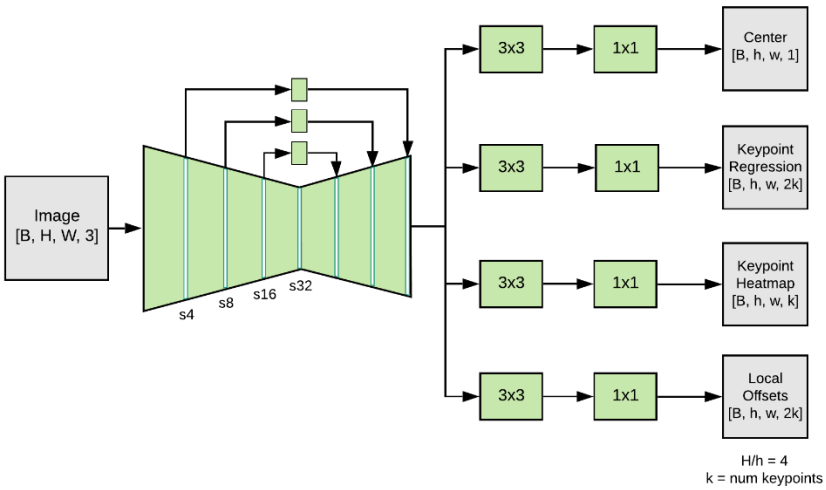


Рисунок 2.1 – архітектура моделі MoveNet.

Визначення проводиться у 4 етапи:

а) Теплова карта центру людини передбачає геометричний центр кожної особи.

б) Поле регресії ключових точок передбачають повний набір ключових точок для особи, що використовується для групування.

в) Теплова карта ключових точок людини передбачає розташування всіх ключових точок, незалежно від осіб.

г) Двовимірне поле зміщення для кожної ключової точки передбачає локальні зсуви від кожного пікселя карти вихідних функцій до точного розташування кожної ключової точки.

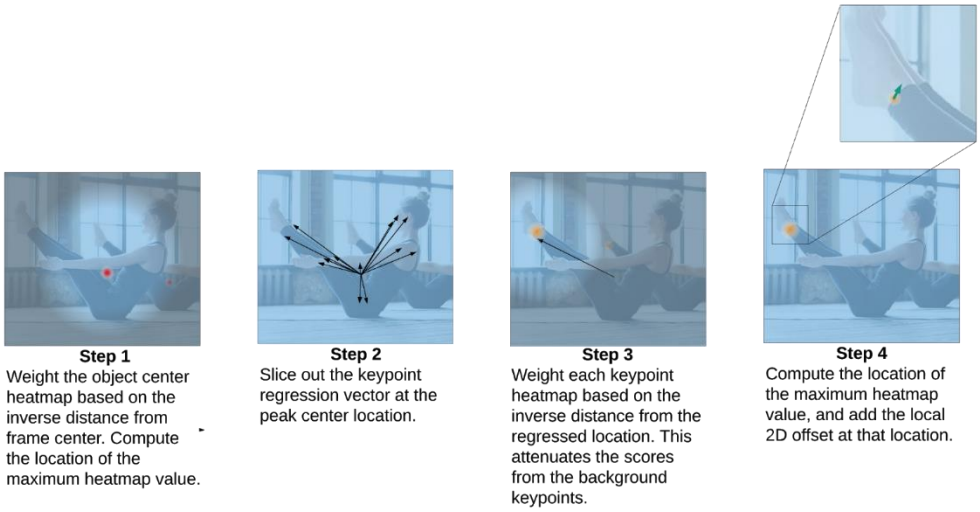


Рисунок 2.2 – демонстрація етапів визначення пози моделлю MoveNet.

Keras — це API глибокого навчання, написаний на Python, який працює поверх платформи машинного навчання TensorFlow. Keras містить безліч реалізацій часто використовуваних будівельних блоків нейронної мережі, таких як шари, цілі, функції активації, оптимізатори та безліч інструментів, які полегшують роботу з графічними та текстовими даними для спрощення кодування, необхідного для написання глибокого коду нейронної мережі.

На додаток до стандартних нейронних мереж, Keras підтримує згорткові та рекурентні нейронні мережі. Він підтримує інші загальні рівні утиліти, такі як вилучення, нормалізація пакетів і об'єднання.

PyTorch – це фреймворк машинного навчання з відкритим вихідним кодом, заснований на бібліотеці Torch, який використовується для таких додатків, як комп’ютерне зір і обробка природної мови, розроблений лабораторією дослідження штучного інтелекту Facebook (FAIR).

Розробка моделей машинного навчання реалізується за допомогою бібліотеки TorchVision – ця бібліотека містить набори даних для тренування, архітектури та трансформації для моделей.[18]

Серед доступних моделей є такі як:

а) AlexNet

б) VGG

в) ResNet

г) SqueezeNet

д) DenseNet

е) Inception v3

ж) ShuffleNet v2

з) MobileNetV3

и) EfficientNet та ін.

MediaPipe — це фреймворк для створення та використання моделей машинного навчання для обробки медіаданих, таких як відео, аудіо тощо. Цей фреймворк був розроблений Google та використовувався внутрішньо у їх сервісах, але згодом випущений для загального користування. MediaPipe пропонує швидкі та точні вирішення задач Computer Vision, в тому числі – і для Pose Estimation.

GluonCV – набір інструментів, алгоритмів та натренованих моделей для Deep Learning. Цей пакет містить засоби для створення швидких і точних рішень та застосунків з Machine Learning, зокрема – для Computer Vision та Pose Estimation.

Alpha Pose — це точна система для Multi Person Pose Estimation, яка здатна до виявлення поз у реальному часі. Ця модель використовує підхід «згори вниз» - за допомогою моделі детектора людей визначаються рамки для кожної особи, пози яких визначаються в окремому порядку. Як детектор часто використовується Yolo v.3, а власне модель-визначник базується на ResNet-101 і тренується даними з набору COCO. Її можна легко реалізувати за допомогою GluonCV. [19]

РОЗДІЛ 3. ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАСТОСУНКУ

3.1 Реалізація застосунку для визначення пози людини у 2D

Застосунок для 2D Multi Person Pose Estimation було створено у середовищі Google Colab і використано фреймворк Tensorflow та Keras.[20]

Як модель було обрано архітектуру OpenPose: програмно відтворено її шари (див. рисунок 3.1) та завантажено претреновані ваги.

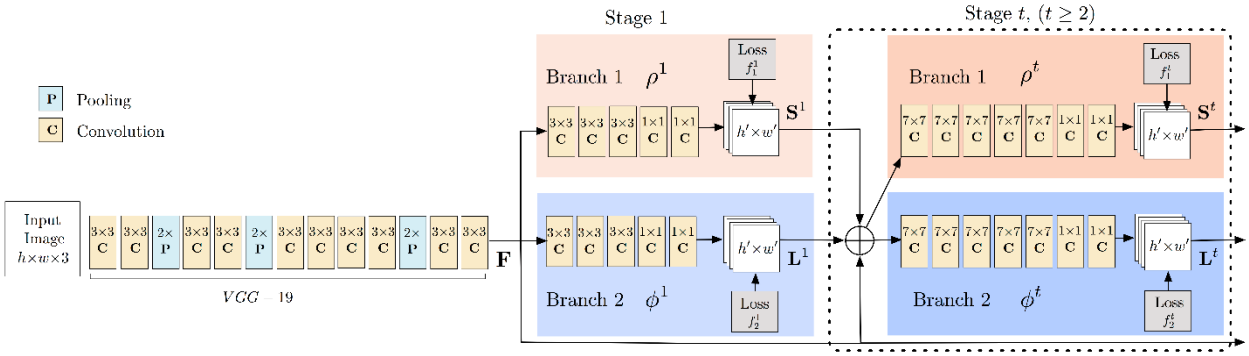


Рисунок 3.1 - архітектура моделі OpenPose.

OpenPose спочатку виявляє частини (ключові точки), що належать кожній людині на зображенні, а потім призначає частини окремим особам. Мережа OpenPose спочатку виявляє особливості за допомогою шарів VGG-19. Потім об’єкти подають у дві паралельні гілки згорткових шарів. Перша гілка передбачає набір з 18 карт, кожна з яких представляє певну частину скелета поза людини. Друга гілка передбачає набір з 38 полів спорідненості (Part Affinity Fields – PAFs), які представляють ступінь асоціації між частинами.

Шари моделі:[21]

а) Шари VGG-19: ця частина відповідає за попередню обробку зображення та визначення особливостей.

1) Conv2D(x, 64, 3, "conv1_1")
2) Activation('relu') (x)
3) Conv2D(x, 64, 3, "conv1_2")
4) Activation('relu') (x)
5) MaxPooling2D (x, 2, 2, "pool1_1")
6) Conv2D (x, 128, 3, "conv2_1")
7) Activation('relu') (x)
8) Conv2D (x, 128, 3, "conv2_2")
9) Activation('relu') (x)
10) MaxPooling2D (x, 2, 2, "pool2_1")
11) Conv2D (x, 256, 3, "conv3_1")
12) Activation('relu') (x)
13) Conv2D (x, 256, 3, "conv3_2")
14) Activation('relu') (x)
15) Conv2D (x, 256, 3, "conv3_3")
16) Activation('relu') (x)
17) Conv2D (x, 256, 3, "conv3_4")
18) Activation('relu') (x)
19) MaxPooling2D (x, 2, 2, "pool3_1")
20) Conv2D (x, 512, 3, "conv4_1")
21) Activation('relu') (x)
22) Conv2D (x, 512, 3, "conv4_2")
23) Activation('relu') (x)
24) Conv2D (x, 256, 3, "conv4_3_CPM")
25) Activation('relu') (x)
26) Conv2D (x, 128, 3, "conv4_4_CPM")
27) Activation('relu') (x)

б) Stage t = 1: на цьому етапі створюються два відгалудження, перше з яких прогнозує позицію частин тіла – будує теплокарти, які визначають

ймовірність знаходження ключових точок, а друге – будує карти полів спорідненості, які визначають зв’язки між суглобами. На першому етапі мережа створює початковий набір карт виявлення S і набір полів спорідненості L.

1) Conv2D (x, 128, 3, "conv6_1_CPM_L%d" % branch)
2) Activation('relu') (x)
3) Conv2D (x, 128, 3, "conv6_2_CPM_L%d" % branch)
4) Activation('relu') (x)
5) Conv2D (x, 128, 3, "conv6_3_CPM_L%d" % branch)
6) Activation('relu') (x)
7) Conv2D (x, 512, 1, "conv6_4_CPM_L%d" % branch)
8) Activation('relu') (x)
9) Conv2D (x, num_p, 1, "conv6_5_CPM_L%d" % branch)

в) Stage t ≥ 2: на кожному наступному етапі, передбачення з обох відгалуджень об’єднуються з особливостями і використовуються для створення більш точних прогнозів. У OpenPose кінцевим етапом є 6.

1) Conv2D (x, 128, 7, "Mconv1_stage%d_L%d" % (stage, branch))
2) Activation('relu') (x)
3) Conv2D (x, 128, 7, "Mconv2_stage%d_L%d" % (stage, branch))
4) Activation('relu') (x)
5) Conv2D (x, 128, 7, "Mconv3_stage%d_L%d" % (stage, branch))
6) Activation('relu') (x)
7) Conv2D (x, 128, 7, "Mconv4_stage%d_L%d" % (stage, branch))
8) Activation('relu') (x)
9) Conv2D (x, 128, 7, "Mconv5_stage%d_L%d" % (stage, branch))
10) Activation('relu') (x)
11) Conv2D (x, 128, 1, "Mconv6_stage%d_L%d" % (stage, branch))
12) Activation('relu') (x)

13) Conv2D (x, (38/19 – для відповідних відгалуджень), 1, "Mconv7_stage%d_L%d" % (stage, branch)).

Модель результуючого скелетону містить 18 ключових точок (див. рисунок 3.2) - це формат вихідної пози COCO. Формат виводу OpenPose це список із 18 кортежів, що містять положення точки по осі X та Y на зображенні та оцінку впевненості.

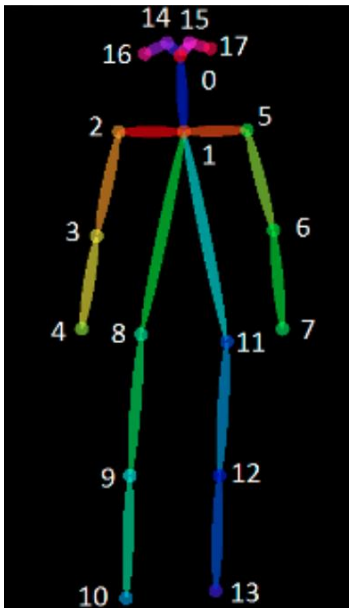


Рисунок 3.2 - будова вихідного скелетону OpenPose.

Розглянемо роботу моделі на прикладі наступного зображення (див. рисунок 3.3).

Рисунок 3.3 - тестове зображення для демонстрації 2D Pose Estimation.

В результаті прогнозування утворюються теплові карти та карти зв’язків. На рисунку 3.4 показана теплова карта для усіх точок ший кожної людини, а на рисунку 3.5 - PAF для лівої ноги та коліна.

```
plt.imshow(oriImg[:,:,[2,1,0]])
plt.imshow(heatmap_avg[:,:,1], alpha=.5)
fig = matplotlib.pyplot.gcf()
cax = matplotlib.pyplot.gca()
fig.set_size_inches(12, 12)
fig.subplots_adjust(right=0.93)
cbar_ax = fig.add_axes([0.95, 0.15, 0.01, 0.7])
_ = fig.colorbar(ax2, cax=cbar_ax)
```

Фрагмент коду 1 - відображення теплокарти для обраної точки.

Рисунок 3.4 - теплокарта для точок шиї.

Рисунок 3.5 - поля спорідненості для лівої ноги та коліна.

Рисунок 3.6 демонструє усі знайдені точки. Для визначення точної позиції точки застосовується визначення піків теплокарт. Потім за допомогою отриманих прогнозів будуються всі пози (див. рисунок 3.7).

Рисунок 3.6 - знайдені ключові точки.

Рисунок 3.7 - побудовані за точками і зв’язками скелетони.

3.2 Реалізація застосунку для визначення пози людини у 3D

Можливість створення та тренування власної моделі є обмеженою – це потребує дуже багато ресурсів та часу, що у даному дослідженні виявилося перепоною. Тому для створення застосунку було використано MediaPipe та модель BlazePose.[22]

BlazePose – модель для 3D Pose Estimation, розроблена у рамках MediaPipe. Вона дозволяє визначати координати 33 ключових точок однієї людини у трьохвимірному просторі (див. рисунок 3.8). BlazePose визначає та сегментує людину та визначає її частини тіла. Ця модель також заточена під визначення поз з відео, тому для визначення наступних поз орієнтується на попередні дані.

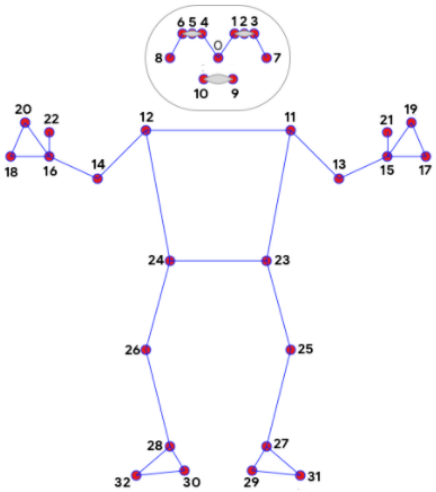


Рисунок 3.8 - будова скелетону людини для Blaze Pose.

Задача застосунку [23]:

а) Завантажити та налаштувати модель.

б) Завантажити зображення.

в) Визначити положення ключових точок.

г) Зобразити ключові точки у 2D просторі (поверх заданого зображення).

д) Зобразити ключові точки у 3D просторі (координатному просторі).

Ініціалізація моделі відбувається наступним чином: with mp_pose.Pose(static_image_mode=True, model_complexity=2, enable_segmentation=True, min_detection_confidence=0.6) as pose.

а) static_image_mode відповідає за те, чи є вхідні зображення послідовністю – якщо так, то модель спрощує виявлення поз на наступних зображеннях за допомогою вже виявленої попередньої.

б) model_complexity визначає точність та швидкість визначення поз (чим більша складність – тим точніше і повільніше).

в) enable_segmentation дозволяє отримувати маску сегментації – виділену частину зображення, де є людина.

г) min_detection_confidence – мінімальна впевненість під час визначення людини, щоб вважати її успішною.

Допоміжною бібліотекою для роботи з зображеннями є OpenCV – її функціонал підтримує завантаження та обробку зображень, а BlazePose працює саме з зображеннями даного формату.

За визначення пози відповідає функція process, яка приймає на вхід зображення, а повертає набір координат у локальному та глобальному форматі та видимість: x, y, z та visibility. Положення по осям X та Y нормалізоване відносно ширини і висоти зображення, а положення по Z визначає близькість до камери. mp_drawing.draw_landmarks(image, results.pose_landmarks, mp_pose.POSE_CONNECTIONS, landmark_drawing_spec = mp_drawing_styles.get_default_pose_landmarks_style()) відображає на зображенні отримані ключові точки та зв’язки у 2D просторі.

Відображення у 3D просторі можна реалізувати за допомогою matplotlib та mpl_toolkits, задавши 3D графік та побудувавши на ньому відрізки. Для спрощення візуалізації точки було трансформовано: ax.plot([landmark[i].x,

landmark[e[1]].x], [landmark[i].z, landmark[e[1]].z], [1-landmark[i].y,1-landmark[e[1]].y]).

3.3 Тестування застосунку

Для тестування застосунку використано 3 зображення (див. рисунок 3.9). Ці фотографії зображують людей у складних позах.

Рисунок 3.9 - фотографії людей у складних позах.

Результати роботи застосунку показані на рисунку 3.10 - зліва розташовані зображення з визначеними точками у 2D просторі, а справа побудовано визначені 3D скелетони.

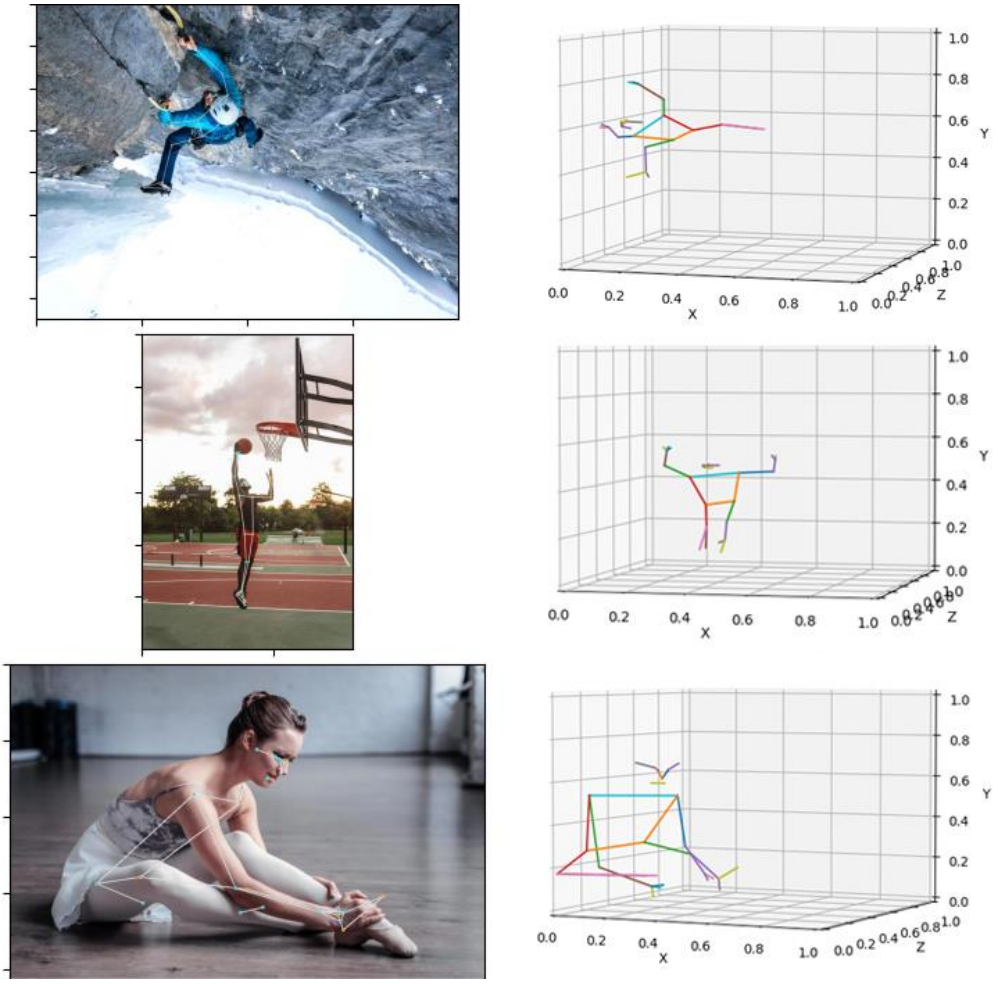


Рисунок 3.10 - результати роботи.

З отриманих результатів можна провести аналіз та зробити висновки:

а) У всіх випадках значення по осі Z є занадто великими відносно X та Y – тобто модель не зовсім точно визначає відстані. Ці значення можна нормалізувати, помноживши на деяке число.

б) На третьому зображенні ліва нога на рука танцівниці видні погано. Через це програма намагалася «вгадати положення», тому вихідний скелетон має кілька похибок.

Загалом було отримані хороші результати роботи. За отриманими координатами можна визначити кути, які, наприклад, визначатимуть пози та дії.

РОЗДІЛ 4. ПОРІВНЯЛЬНИЙ АНАЛІЗ РОБОТИ РІЗНИХ МОДЕЛЕЙ

4.1 Постановка та опис завдання аналізу

Задля кращого розуміння роботи різних моделей та пошуку найоптимальнішого варіанту було проведено порівняльний аналіз роботи 5 моделей Pose Estimation:

а) OpenPose,

б) AlphaPose,

в) BlazePose,

г) MoveNet Lightning,

д) MoveNet Thunder.

Інформація про ці моделі наведена у розділах 2.3 та 3.1-3.2.

OpenPose та AlphaPose є різними за принципом роботи, MoveNet Lightning та MoveNet Thunder відрізняються точністю та швидкістю, тоді як BlazePose є 3D моделлю. Також, відмінністю цих моделей є різний формат результатів – і координат, і власне скелетону. Тому порівняння координат буде проведено для наступних ключових точок: носу, верхніх та нижніх кінцівок.

Аналіз було проведено практичним способом:

а) У мережі Інтернет було знайдено 4 зображення людей, які знаходяться у складних позах, частини тіл деяких людей є прихованими або відсутні на зображеннях.

б) Створено демонстраційні застосунки (у випадку OpenPose та BlazePose було використано реалізовані застосунки у розділах 3.1 та 3.2 відповідно) для визначення поз за допомогою вищезгаданих моделей.

в) Результати роботи моделей та застосунків – зображення з позначеними ключовими точками та json-файли з координатами цих точок, - використано як вхідні дані у програмі-аналізаторі.

г) Через відмінності у форматі даних координати було нормалізовано та підігнано під єдиний формат.

д) Координати частин тіла зберігаються у вкладених списках, і статистичні показники (середні значення та дисперсія) окремо розраховуються для кожної частини тіла.

е) За показниками роботи моделей було обрано найкращу та визначено суму відстаней від її ключових точок до точок інших моделей.

4.2 Аналіз роботи та результати

Застосунки для визначення поз отримували 4 зображення, на кожному зображенні знаходилася одна людина у складній позі, на деяких зображеннях одна чи кілька кінцівок приховані. Нехай ці зображення класифікуються наступним чином:

а) «Фермер» - складність визначення пози полягає у тому, що ліва рука та нога чоловіка приховані, і рослинність частково приховує праве коліно.

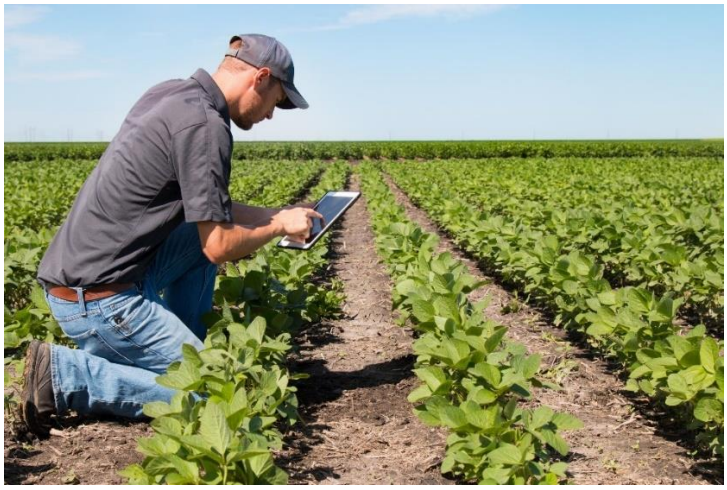


Рисунок 4.1 - зображення «Фермер» для визначення пози.

б) «Ді-джей» - фотографію було зроблено по пояс, а отже, ніг не видно, і права рука схована.

Рисунок 4.2 - зображення «Ді-джей» для визначення пози.

в) «Стрибок» - фотографію було зроблено в русі, голову і праву ногу практично не видно.

Рисунок 4.3 - зображення «Стрибок» для визначення пози.

г) «Меч» - перспектива зображення, одяг дівчини та його колір запобігають точному визначенню позиції частин тіла.

Рисунок 4.4 - зображення «Меч» для визначення пози.

Результуючі зображення з отриманими ключовими точками та зв’язками знаходяться у додатку А. Серед них можна виявити найкращі та найгірші пози (див. таблицю 2). Як продемонстровано, найкраще з визначенням поз впоралися OpenPose та AlphaPose, а найгірше – MoveNet Lightning.

Зображення	Найкраще визначення	Найгірше визначення
Фермер	OpenPose, AlphaPose	Lightning, Thunder
Ді-джей	AlphaPose	BlazePose, Lightning, Thunder
Стрибок	OpenPose	Lightning
Меч	BlazePose	Lightning

Таблиця 2 - результати попереднього аналізу визначених поз.

Для визначення точності окремих ключових точок скористаємося поняттям дисперсії. Дисперсія - це міра того, наскільки значення у вибірці розсіяні. Велике значення дисперсії вказує на те, що дані розкидані, а мале – що згруповані навколо середнього значення.

У таблиці 3 наведені мінімальні та максимальні значення дисперсії та ці точки; за цими результатами можна зробити висновки, які частини тіла визначалися точно, а де моделі допускали помилки.

	Мін. по X	Макс. По X	Мін. по У	Макс. По У
Фермер	r_hand: 3.15e-06	r_foot: 0.0011	r_hand: 2.788e-05	l_knee: 0.0048
Ді-джей	l_hand: 9.5e-05	r_knee: 0.0048	l_shoulder: 0.00047	l_knee: 0.056
Стрибок	nose: 0.0003	l_hand: 0.0295	l_elbow: 6.8e-05	l_knee: 0.0032
Меч	r_shoulder: 0.000135	l_knee: 0.0059	r_elbow: 8.79e-05	l_hand: 0.0083

Таблиця 3 - мінімальні і максимальні значення дисперсій.

Найкращі результати були продемонстровані на зображенні «фермер» - визначені моделями точки є найменш розсіяними. Якщо побудувати позу за середніми значеннями точок, то вони знаходимуться на коректних позиціях або будуть віддалені на невелику відстань від них (див. рисунок 4.5).

Рисунок 4.5 - поза, отримана на середніми значеннями точок.

Раніше було визначено, що саме AlphaPose визначає ключові точки найточніше, а отже – проведемо порівняльний аналіз позицій точок, які отримані за допомогою AlphaPose та інших моделей. Для розрахунку точності було використано метод суми відстаней між точками (див таблицю 4).

	OpenPose	BlazePose	Lightning	Thunder
Фермер	0.165	0.425	0.92	0.825
Ді-джей	0.164	2.402	2.61	2.778
Стрибок	0.341	1.547	1.012	1.798
Меч	0.102	0.552	1.769	1.054

Таблиця 4 - суми відстаней між ключовими точками.

ВИСНОВКИ

Під час виконання роботи було опановано теоретичні та практичні засади машинного навчання та його розділів – глибинного навчання та комп’ютерного зору, проведено дослідження моделей нейронних мереж для завдання визначення пози людини та огляд інструментів і застосунків для моделювання мереж і визначення поз за допомогою мови програмування Python та її надбудов.

Результатом роботи стали застосунки для визначення поз у 2D та 3D просторі з заданих користувачем зображень. Ці продукти продемонстрували точність у роботі та можуть використовуватися як основа для, наприклад, класифікатора дій або для переносу у графічний простір для подальшої роботи.

ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Burns E. What Is Machine Learning and Why Is It Important? [Електронний ресурс] / Ed Burns // SearchEnterpriseAI. – Режим доступу: https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML

2. Contributors to Wikimedia projects. Machine learning - Wikipedia [Електронний ресурс] / Contributors to Wikimedia projects // Wikipedia, the free encyclopedia. – Режим доступу: https://en.wikipedia.org/wiki/Machine_learning

3. Contributors to Wikimedia projects. Artificial neural network - Wikipedia [Електронний ресурс] / Contributors to Wikimedia projects // Wikipedia, the free encyclopedia. – Режим доступу: https://en.wikipedia.org/wiki/Artificial_neural_network

4. Contributors to Wikimedia projects. Deep learning - Wikipedia [Електронний ресурс] / Contributors to Wikimedia projects // Wikipedia, the free encyclopedia. – Режим доступу: https://en.wikipedia.org/wiki/Deep_learning

5. Contributors to Wikimedia projects. Convolutional neural network - Wikipedia [Електронний ресурс] / Contributors to Wikimedia projects // Wikipedia, the free encyclopedia. – Режим доступу: https://en.wikipedia.org/wiki/Convolutional_neural_network

6. Contributors to Wikimedia projects. Computer vision - Wikipedia [Електронний ресурс] / Contributors to Wikimedia projects // Wikipedia, the free encyclopedia. – Режим доступу: https://en.wikipedia.org/wiki/Computer_vision

7. Human Pose Estimation: Approaches, Use Cases, Implementation Tips – ITRex [Електронний ресурс] // ITRex. – Режим доступу: https://itrexgroup.com/blog/human-pose-estimation-use-cases-implementation-tips/

8. A Comprehensive Guide on Human Pose Estimation - Analytics Vidhya [Електронний ресурс] // Analytics Vidhya. – Режим доступу: https://www.analyticsvidhya.com/blog/2022/01/a-comprehensive-guide-on-human-pose-estimation/

9. Pose Estimation Guide | Fritz AI [Електронний ресурс] // Fritz AI | Machine Learning for iOS, Android, SnapML. – Режим доступу: https://www.fritz.ai/pose-estimation

10. Mask R-CNN: A Beginner's Guide - viso.ai [Електронний ресурс] // viso.ai. – Режим доступу: https://viso.ai/deep-learning/mask-r-cnn/

11. MPII Human Pose Database [Електронний ресурс] // MPII Human Pose Database. – Режим доступу: http://human-pose.mpi-inf.mpg.de/#dataset

12. COCO - Common Objects in Context [Електронний ресурс] // COCO - Common Objects in Context. – Режим доступу: https://cocodataset.org/#home

13. Leeds Sports Pose Dataset [Електронний ресурс]. – Режим доступу: http://sam.johnson.io/research/lsp.html

14. Vision AI | Derive Image Insights via ML | Cloud Vision API | Google Cloud [Електронний ресурс] // Google Cloud. – Режим доступу: https://cloud.google.com/vision

15. Computer Vision and Machine Vision Solutions [Електронний ресурс] // Intel. – Режим доступу: https://www.intel.com/content/www/us/en/internet-of-things/computer-vision/vision-products.html

16. Tag: 3D Pose Estimation | NVIDIA Technical Blog [Електронний ресурс] // NVIDIA Technical Blog. – Режим доступу: https://developer.nvidia.com/blog/tag/3d-pose-estimation/

17. Pose estimation | TensorFlow Lite [Електронний ресурс] // TensorFlow. – Режим доступу: https://www.tensorflow.org/lite/examples/pose_estimation/overview

18. torchvision – Torchvision 0.12 documentation [Електронний ресурс] // PyTorch. – Режим доступу: https://pytorch.org/vision/stable/index.html

19. Predict with pre-trained AlphaPose Estimation models – gluoncv 0.11.0 documentation [Електронний ресурс] // GluonCV Toolkit. – Режим доступу: https://cv.gluon.ai/build/examples_pose/demo_alpha_pose.html

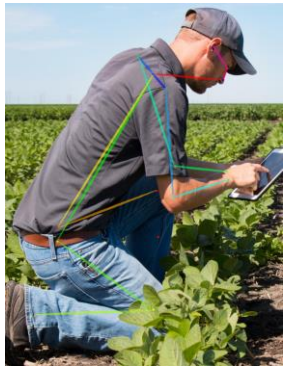
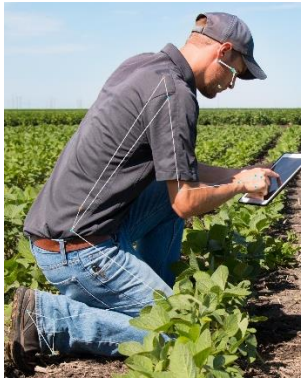
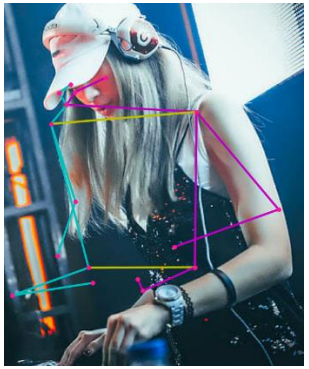
20. PoseEstimation/OpenPose.ipynb at main · daisysermech/PoseEstimation [Електронний ресурс] // GitHub. – Режим доступу: https://github.com/daisysermech/PoseEstimation/blob/main/OpenPose.ipynb

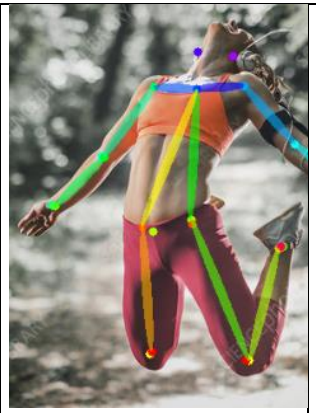
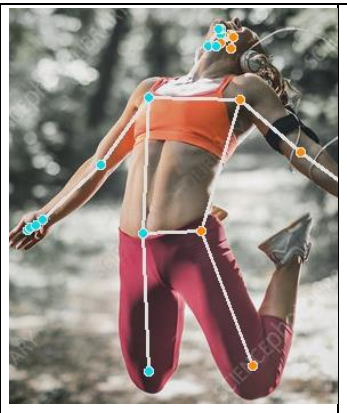
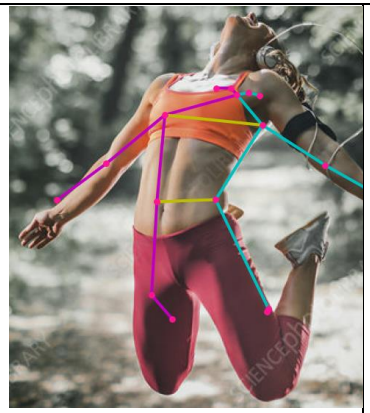
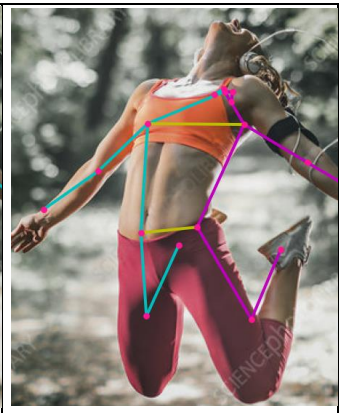
21. Tanugraha P. Understanding OpenPose (with code reference) - Part 1 [Електронний ресурс] / Peter Tanugraha // Medium. – Режим доступу: https://medium.com/analytics-vidhya/understanding-openpose-with-code-reference-part-1-b515ba0bbc73

22. Pose [Електронний ресурс] // mediapipe. – Режим доступу: https://google.github.io/mediapipe/solutions/pose.html

23. PoseEstimation/BlazePose.ipynb at main · daisysermech/PoseEstimation [Електронний ресурс] // GitHub. – Режим доступу: https://github.com/daisysermech/PoseEstimation/blob/main/BlazePose.ipynb

ДОДАТОК А. Результати визначення поз для порівняльного аналізу

Тест	OpenPose	AlphaPose	BlazePose	Lightning	Thunder
Фермер					
Ді-джей					

Стрибок					
Меч