МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики
(повне найменування назва факультету)

дискретного аналізу та інтелектуальних систем
(повна назва кафедри)

Дипломна робота

Використання алгоритмів машинного навчання для розпізнавання дорожніх знаків

Виконав: студент групи ПМі-45
спеціальності
122 «Комп'ютерні науки»
(шифр і назва спеціальності)

Мельник С.В.

(підпис) (прізвище та ініціали)

Керівник Коркуна Н.М.
(підпис) (прізвище та ініціали)

Рецензент
(підпис) (прізвище та ініціали)

2023

Зміст

Вступ

З швидким зростанням автотранспорту і рухомих засобів у світі, безпека на дорозі стала однією з найважливіших проблем. Один з основних аспектів безпечного руху на дорозі - правильне розпізнавання та розуміння дорожніх знаків. Це особливо важливо для систем автономного водіння, які потребують точного визначення правил руху та передачі цієї інформації водієві або автоматичній системі керування.

У цьому контексті алгоритми машинного навчання здобули широке застосування в задачах розпізнавання дорожніх знаків. Вони дозволяють автоматично визначати, аналізувати та класифікувати різноманітні знаки на основі навчання на великому обсязі даних. Такі алгоритми машинного навчання надають можливість використовувати розпізнавання дорожніх знаків у режимі реального часу, що сприяє покращенню безпеки на дорозі.

Однак, досягнення точності та надійності розпізнавання дорожніх знаків за допомогою алгоритмів машинного навчання залишається викликом. Проблеми, з якими стикаються дослідники, включають зміну освітлення, забруднення знаків, зміну погодних умов та різноманітність геометричних форм та кольорів знаків. Крім того, потрібно розв'язати завдання високошвидкісного розпізнавання знаків, щоб забезпечити ефективність та безпеку в системах автономного водіння.

Мета

Метою моє роботи є створення нейронної мережі, яку теоретично можна було б використовувати в безпілотному автомобілі. Але враховуючи, що у мене немає такого автомобіля, я реалізую веб-сайт, основна функція якого буде розпізнавання дорожніх знаків.

Розділ 1. Теоретичні відомості

1.1 Штучний інтелект

Штучний інтелект — це копіювання людського інтелекту в пристроях, розроблених для міркування та отримання знань подібно до людей. Метою цієї величезної галузі інформатики є розробка інтелектуальних машин, здатних виконувати операції, які зазвичай вимагають людського інтелекту.

Машинне навчання, обробка природної мови, комп'ютерний зір, робототехніка, експертні системи та інші сфери входять до широкої категорії штучного інтелекту. Ці методи дають комп'ютерам можливість спостерігати, думати, навчатися та приймати рішення.

Важливою частиною штучного інтелекту є машинне навчання, яке включає в себе алгоритми, які дозволяють машинам навчатися на основі даних і робити прогнози або виконувати дії на основі цих даних. Системи штучного інтелекту можуть автоматично підвищувати свою продуктивність за допомогою машинного навчання, а не вручну навчатися кожному окремому завданню.

Сфера комп'ютерного зору пов'язана з наданням машинам здатності сприймати та інтерпретувати візуальні дані з зображень або фільмів. Системи штучного інтелекту можуть «бачити» й оцінювати візуальний контент завдяки таким завданням, як розпізнавання об'єктів, класифікація зображень, виявлення об'єктів і синтез зображень.

1.2 Машинне навчання

Машинне навчання — це галузь штучного інтелекту, яка зосереджена на розробці алгоритмів і моделей, які дозволяють комп'ютерам навчатися та приймати прогнози чи рішення без явного програмування. Сфера охоплює широкий спектр методів і підходів, які дозволяють машинам аналізувати та інтерпретувати дані, розпізнавати шаблони та покращувати свою продуктивність з часом.

Основна концепція машинного навчання полягає в тому, щоб комп’ютери могли навчатися на прикладах або досвіді, подібно до того, як навчаються люди. Замість того, щоб явно програмувати комп’ютер для виконання конкретного завдання, алгоритми машинного навчання можуть аналізувати великі обсяги даних, визначати закономірності та тенденції, а також робити прогнози або вживати заходів на основі цієї інформації.

Існує кілька типів алгоритмів машинного навчання, кожен із яких має свої особливості та застосування:

- Контрольоване навчання: цей тип навчання передбачає навчання моделі на позначених даних, де відомий бажаний результат. Алгоритм навчається зіставляти вхідні дані з правильним виходом шляхом узагальнення з позначених прикладів. Поширені методи навчання під наглядом включають лінійну регресію, дерева рішень, опорні векторні машини і нейронні мережі.
- Неконтрольоване навчання: у неконтрольованому навчанні алгоритм працює з даними без міток і спрямований на виявлення прихованих шаблонів або структур у даних. Модель вчиться визначати подібності, відмінності або кластери в даних без будь-яких попередньо визначених результатів. Алгоритми кластеризації та методи зменшення розмірності, такі як аналіз головних компонентів, є прикладами неконтрольованого навчання.
- Навчання з підкріпленням: цей тип навчання передбачає навчання агента взаємодії з навколишнім середовищем і навчання оптимальним діям методом проб і помилок. Агент отримує зворотній зв’язок у вигляді винагород або штрафів на основі своїх дій, що дозволяє йому вчитися на наслідках своїх рішень. Навчання з підкріпленням зазвичай використовується в таких програмах, як робототехніка, ігри та автономні системи.

Машинне навчання знайшло застосування в різних областях, зокрема:

- Розпізнавання зображень і мовлення: алгоритми машинного навчання можна навчити розпізнавати та класифікувати зображення, виявляти об’єкти та транскрибувати мовлення. Це призвело до прогресу в таких сферах, як комп’ютерний зір, автономні транспортні засоби та голосові помічники.
- Обробка природної мови (NLP): NLP зосереджується на тому, щоб дозволити комп’ютерам розуміти, інтерпретувати та створювати людську мову. Методи машинного навчання, такі як рекурентні нейронні мережі (RNN) і трансформаторні моделі, значно покращили переклад мови, аналіз настрою і функціональності чат-ботів.
- Охорона здоров’я та медицина: машинне навчання використовується для аналізу медичних даних, діагностики захворювань, прогнозування результатів пацієнтів і рекомендацій планів лікування. Це може допомогти в ранньому виявленні захворювань, персоналізованій медицині та відкритті ліків.
- Фінансовий аналіз. Алгоритми машинного навчання використовуються для аналізу фінансових даних, прогнозування цін на акції, виявлення шахрайських транзакцій і автоматизації торгових стратегій.
- Системи рекомендацій: багато онлайн-платформ, таких як веб-сайти електронної комерції та потокові сервіси, використовують алгоритми машинного навчання, щоб надавати персоналізовані рекомендації користувачам на основі їхніх уподобань і поведінки.

Важливо зазначити, що алгоритми машинного навчання вимагають високоякісних даних для навчання й оцінювання, а також ретельного розгляду етичних міркувань і міркувань справедливості, щоб уникнути упереджень і дискримінаційних результатів.

1.3 Глибоке навчання

Глибоке навчання – це техніка машинного навчання, яка вчить комп'ютери робити те, що є природним для людини: вчитися на прикладі. Глибоке навчання є ключовою технологією для безпілотних автомобілів, що дозволяє їм розпізнавати знак зупинки або відрізняти пішохода від ліхтарного стовпа. Це ключ до голосового керування споживчими пристроями, такими як телефони, планшети, телевізори та гучномовець. Останнім часом глибокому навчанню приділяється багато уваги, і це не дарма. Йдеться про досягнення результатів, які раніше були неможливими.

У глибокому навчанні комп'ютерна модель вчиться виконувати завдання класифікації безпосередньо з зображень, тексту чи аудіо. Моделі глибокого навчання можуть досягти найсучаснішої точності, іноді перевищуючи продуктивність людського рівня. Моделі навчаються з використанням великого набору мічених даних і архітектури нейронних мереж, які містять кілька рівнів.

Глибоке навчання поділяється на підмножини:

Розпізнавання зображень: галузь штучного інтелекту (ШІ), яка дозволяє комп'ютерам і системам отримувати значущу інформацію з цифрових зображень, відео та інших візуальних даних і виконувати дії або давати рекомендації на основі цієї інформації. Якщо ШІ дозволяє комп'ютерам мислити, комп'ютерний зір дозволяє їм бачити, спостерігати та розуміти. Розпізнавання зображень працює так само, як людський зір, за винятком того, що люди мають перевагу. Людський зір має ту перевагу, що все життя в контексті навчить вас розрізняти об'єкти, як далеко вони знаходяться, чи рухаються вони та чи щось не так на зображенні. Розпізнавання зображень навчає машини виконувати ці функції, але воно має робити це за набагато менший час, використовуючи камери, дані та алгоритми, а не сітківку ока чи нерви очей і зорової кори. Оскільки система, навчена перевіряти продукти або стежити за виробничими активами, може аналізувати тисячі продуктів або

процесів за хвилину, помічаючи непомітні дефекти чи проблеми, вона може швидко перевищити людські можливості.

Обробка природної мови (NLP) відноситься до галузі інформатики, точніше до галузі штучного інтелекту або штучного інтелекту, яка займається тим, щоб дозволити комп'ютерам розуміти текст і вимовлені слова так само, як люди. NLP поєднує в собі комп'ютерну лінгвістику – засноване на правилах моделювання людської мови - зі статистичними моделями, машинним навчанням і глибоким навчанням. Разом ці технології дозволяють комп'ютерам обробляти людську мову у формі тексту або голосових даних і «розуміти» її повне значення, включаючи наміри та почуття того, хто говорить або пише. NLP контролює комп'ютерні програми, які перекладають текст з однієї мови на іншу, реагують на голосові команди та швидко підсумовують великі обсяги тексту — навіть у реальному часі. Крім того, NLP використовується для голосових систем GPS, цифрових помічників, програмного забезпечення для диктування мови в текст, чат-ботів обслуговування клієнтів та інших споживчих зручностей. Але NLP також відіграє все більш важливу роль у корпоративних рішеннях, які допомагають оптимізувати бізнес-операції, підвищити продуктивність працівників і спростити критично важливі бізнес-процеси.

1.4 Мова програмування Python

Python — це універсальна, широко використовувана мова програмування високого рівня, відома своєю простотою та зручністю читання. Python підкреслює читабельність коду та використовує відступи замість круглих дужок або ключових слів для структурування блоків, що полегшує розуміння та написання.

Ось деякі ключові функції та аспекти Python:

- Зручність читання: Синтаксис Python розроблений таким чином, щоб бути

ясним і лаконічним, що робить його легким для читання та розуміння. Використання відступів як структурного елемента сприяє узгодженості та читабельності коду.

- Універсальність: Python підтримує різні парадигми програмування, включаючи процедурне, об’єктно-орієнтоване та функціональне програмування.
- Динамічно типізований: Python динамічно типізується, тобто типи змінних визначаються під час виконання. Вам не потрібно явно оголошувати типи змінних, що дозволяє більш гнучке та виразне кодування.
- Інтерпретований: Python є інтерпретованою мовою, що означає, що він не потребує явної компіляції. Натомість код Python виконується рядок за рядком інтерпретатором, що робить процес розробки більш інтерактивним і сприяє швидкому створенню прототипів.
- Міжплатформна сумісність: Python є кросплатформною мовою. Це означає, що код, написаний на Python, може працювати в різних операційних системах, таких як Windows, macOS і Linux.

Python широко використовується в різних областях і програмах, зокрема:

- Веб-розробка: фреймворки Python, такі як Django та Flask, дозволяють швидко розробляти веб-додатки.
- Аналіз даних і наука: Python разом із такими бібліотеками, як Pandas, NumPy і SciPy, широко використовується для маніпулювання даними, аналізу та візуалізації в таких сферах, як наука про дані та аналітика.
- Машинне навчання та штучний інтелект: простота Python і доступність таких бібліотек, як TensorFlow, PyTorch і scikit-learn, зробили його популярним вибором для машинного навчання та досліджень і розробок ШІ.

- Наукові обчислення: Python разом із такими бібліотеками, як SciPy і Matplotlib, забезпечує надійну платформу для наукових обчислень і моделювання.

Популярність і універсальність Python зробили його однією з найбільш поширених мов програмування в різних галузях і областях. Його простота використання, великий вибір бібліотек роблять його чудовим вибором як для початківців, так і для досвідчених розробників.

Розділ 2. Аналіз та алгоритмічне забезпечення систем з розпізнавання дорожніх знаків.

2.1 Аналіз систем розпізнавання дорожніх знаків

Дорожні знаки відіграють важливу роль у безпеці дорожнього руху та надають важливу інформацію водіям. Однак ручне визначення знаків може бути дорогим процесом і схильним до людських помилок. Використання алгоритмів машинного навчання дозволяє автоматизувати цей процес і підвищити точність розпізнавання.

Дорожні знаки можуть бути різної форми, різних кольорів і мати багато різних символів і піктограм. Деякі знаки можуть використовуватися в усьому світі, тоді як інші можуть бути неприйнятними для решти країн. Все це вимагає розробки алгоритмів, які можуть розпізнавати велику кількість ознак і відрізняти їх характеристики.

Щоб застосувати алгоритм машинного навчання для розпізнавання дорожніх знаків, нам необхідний великий набір даних для навчання моделі. Ці дані можуть бути зібрані з різних джерел. Це також можуть бути фотографії дорожніх знаків, відео про дорожній рух і спеціальні набори даних.

Основні методи та алгоритми, що використовуються для розпізнавання дорожніх знаків, включають:

- Згорткова нейронна мережа (Convolutional Neural Networks, CNN). Це

глибока нейронна мережа для обробки та аналізу зображень, включаючи розпізнавання дорожніх знаків.

- Використання інструментів покращення зображення. Щоб покращити якість фотографій, можна використовувати низку методів, зокрема фільтрування, нормалізацію, видалення фону тощо. Це покращує ефективність алгоритмів розпізнавання.
- Поєднання кількох моделей і алгоритмів. Ми можемо використовувати ансамблеві підходи, які поєднують результати різних моделей або алгоритмів, щоб отримати підвищену точність і надійність.

Також ось кілька проблем пов’язаних з розпізнаванням дорожніх знаків:

- Недостатня кількість даних для навчання моделі
- Вплив освітлення та змін середовища
- Швидкість та час відгуку
- Розпізнавання на великій відстані
- Різноманітність дорожніх знаків

2.2 Алгоритмічне забезпечення

2.2.1 Згорткові нейронні мережі

Згорткові нейронні мережі (Convolutional Neural Networks, CNN) є типом нейронних мереж, спеціально розроблених для обробки зображень і використовуються в різних завданнях комп'ютерного зору. Також їх можна використовувати для розпізнавання дорожніх знаків. У 1988 році Ян ЛеКун побудував першу згорткову нейронну мережу яка називається LeNet. LeNet використовувалася для завдань розпізнавання символів, таких як читання поштових індексів і цифр.

У CNN кожне зображення представлено у вигляді масиву значень пікселів. Ось приклад представлення зображення числа 8:

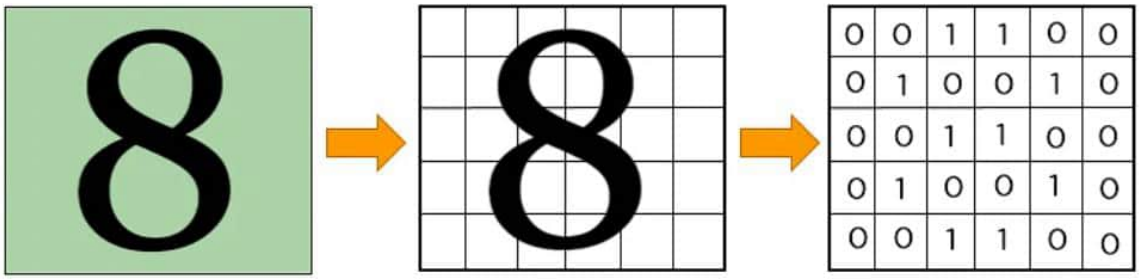


Рис 2.1 Представлення числа 8

Шари в CNN:

1. Шар згортки
2. Шар ReLU
3. Шар об'єднання
4. Повнозв'язаний шар

Шар згортки

З цього починається процес виявлення цінних елементів із зображення. Кілька фільтрів працюють разом, щоб виконати дію згортки в шарі згортки. Кожне зображення можна розглядати як матрицю значень пікселів. Візьмемо до уваги наступне зображення 5x5, де значення кожного пікселя дорівнює 0 або 1. Також присутня матриця фільтра розміром 3x3.

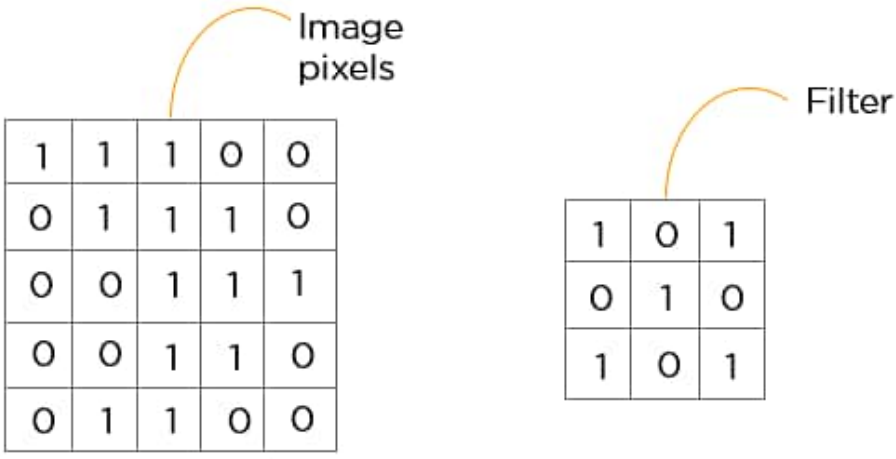


Рис 2.2 Матриця зображення та фільтра

Щоб отримати згорнуту матрицю ознак, потрібно провести матрицею фільтра по зображенню та обчислити скалярний добуток.

Функція ReLU

ReLU означає випрямлену лінійну одиницю. Після того як карти функцій видобуто, наступним кроком є їх переміщення за допомогою функції ReLU. ReLU виконує операцію поелементно, встановлюючи всі негативні пікселі на 0. Результатом є виправлена карта функцій, яка надає мережі нелінійність. Вихідне зображення сканується з кількома згортками та шарами, які використовували ReLU, для визначення місцезнаходження елементів.

Шар об'єднання

Об'єднання — це операція зменшення вибірки, яка зменшує розмірність карти функцій. Виправлена карта об'єктів тепер проходить через шар об'єднання для створення об'єднаної карти об'єктів. Найпоширенішим типом об'єднання є максимальне об'єднання(max pooling), де максимальне значення вибирається в кожній області та передається на наступний рівень.

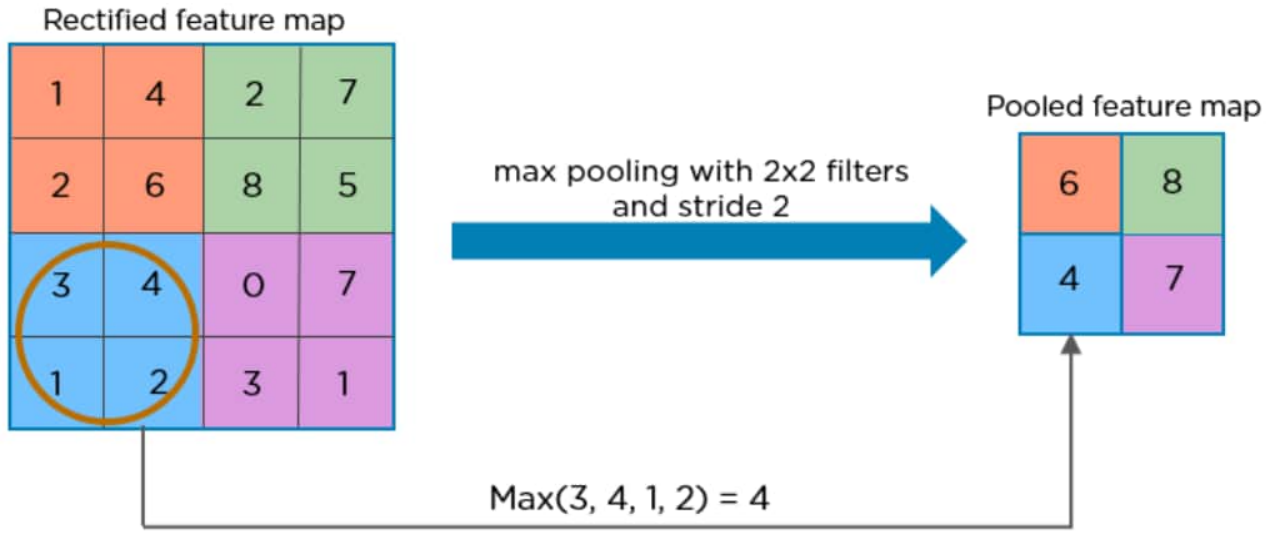


Рис 2.3 Максимальне об'єднання

Шар об'єднання використовує різноманітні фільтри для визначення різних частин зображення, таких як краї, кути.

Наступний крок у цьому процесі називається зведенням. Зведення використовується для перетворення всіх результуючих 2-вимірних масивів із об'єднаних карт об'єктів в один довгий безперервний лінійний вектор.

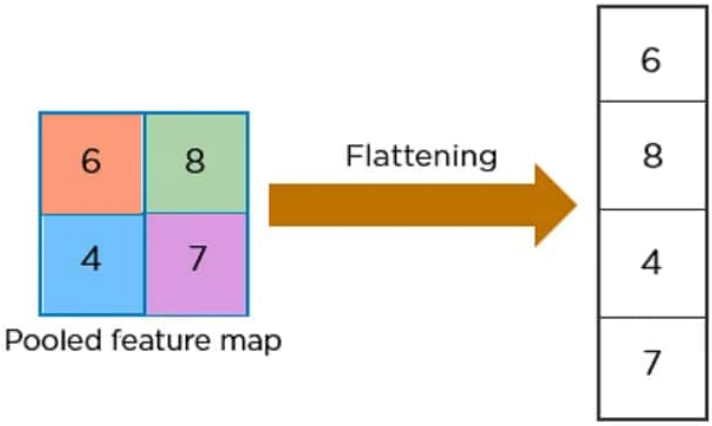


Рис 2.4 Крок зведення

Сплощена матриця подається як вхідна інформація до повнозв'язаного шару для класифікації зображення.

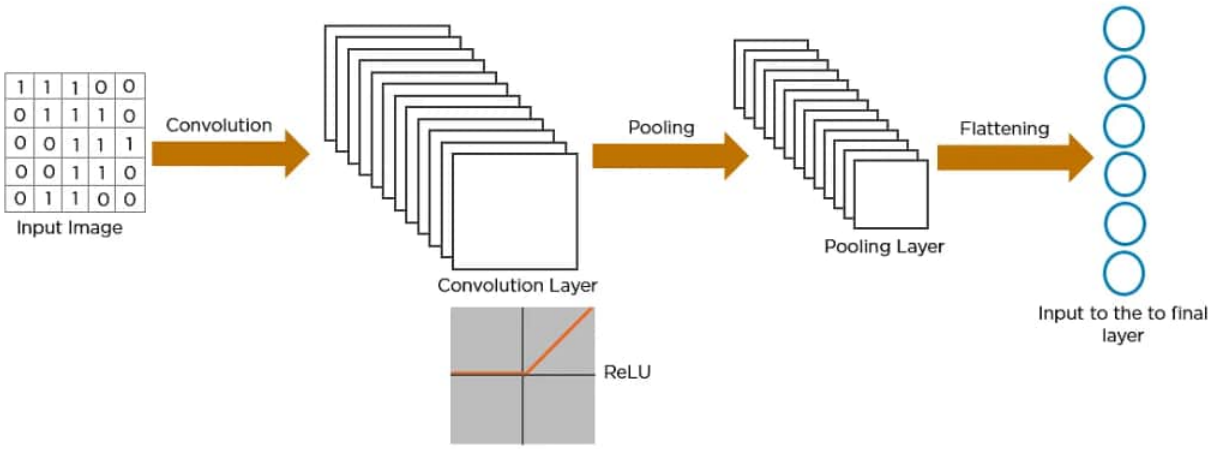


Рис 2.5 Перехід до повнозв'язного шару

Повнозв'язний шар

Використовується для ідентифікації та класифікації об'єктів на основі інформації, отриманої з попередніх шарів. Цей рівень з'єднує кожен нейрон одного шару з кожним нейроном наступного шару та забезпечує приблизні висновки.

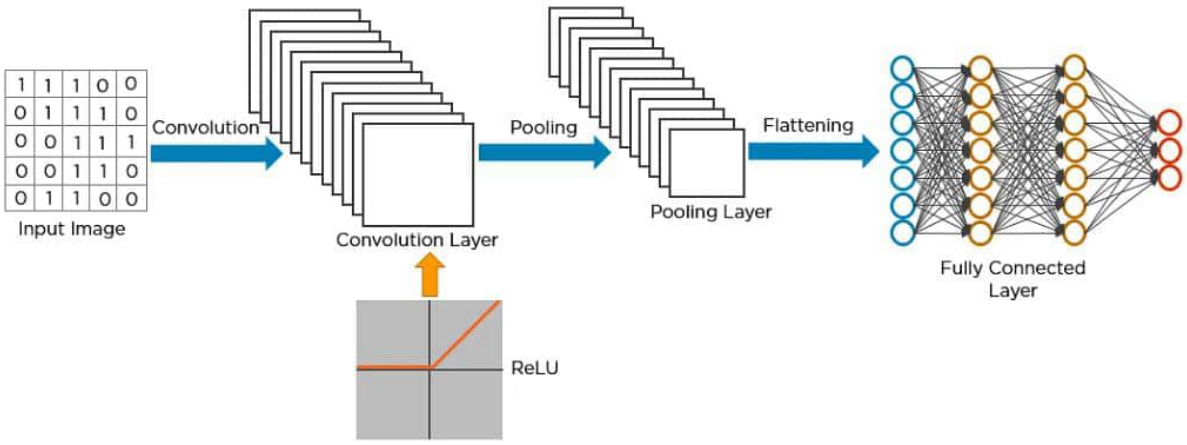


Рис 2.6 Повна візуалізація алгоритму

2.2.2 Рекурентні нейронні мережі

Рекурентні нейронні мережі (Recurrent Neural Networks, RNN) – це ще один тип нейронної мережі яку можна використовувати для розпізнавання дорожніх знаків. RNN є досить ефективними при роботі з даними які є залежні від часу, оскільки вони можуть виявляти залежності та патерни в часі. RNN є дуже корисним в ситуаціях, де часова інформація є досить важливою. До прикладу, у випадках розпізнавання дорожніх знаків на основі відео або послідовних кадрів з камери у автомобілі.

Якщо RNN навчати на досить великому та багатому наборі даних, алгоритм може ефективно розпізнавати та класифікувати різні дорожні знаки з досить високою точністю. Нижче показано, як можна перетворити нейронну мережу прямого зв’язку на рекуренту нейронну мережу:

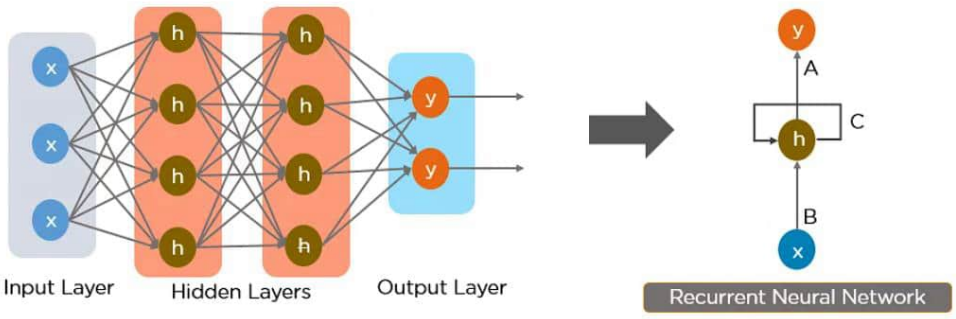


Рис 2.7 Проста RNN

Вузли в різних шарах нейронної мережі стискаються, щоб утворити один шар рекурентних нейронних мереж. A, B і C — параметри мережі.

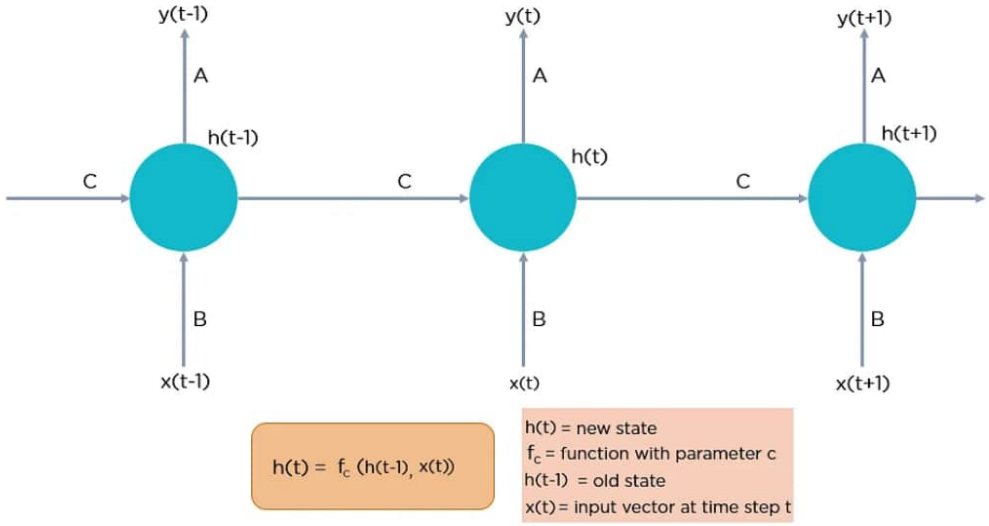


Рис 2.8 Повністю підключена RNN

Тут «x» — вхідний шар, «h» — прихований шар, а «y» — вихідний шар. A, B і C — параметри мережі, які використовуються для покращення виходу моделі. У будь-який момент часу t поточний вхідний сигнал є комбінацією вхідного сигналу в x(t) і x(t-1). Вихідні дані в будь-який момент часу повертаються в мережу для покращення результатів.

У рекурентних нейронних мережах інформація проходить циклічно до середнього прихованого шару.

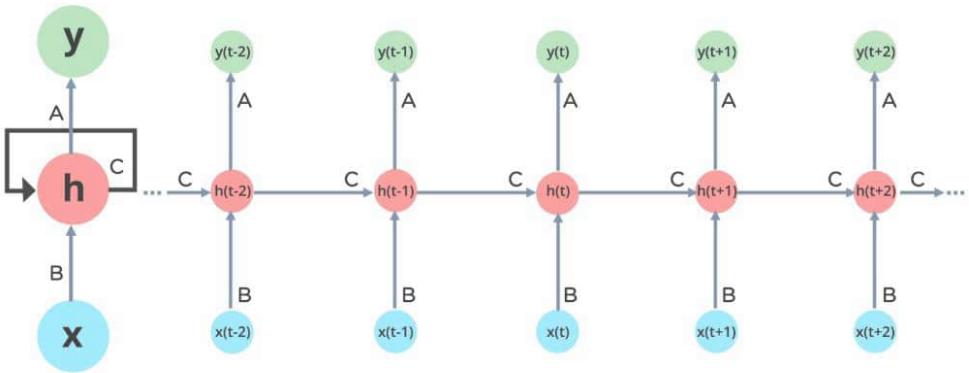


Рис 2.9 Робота RNN

Вхідний рівень «x» приймає вхідні дані для нейронної мережі, обробляє їх і передає на середній рівень.

Середній шар «h» може складатися з кількох прихованих шарів, кожен зі своїми функціями активації, вагами та упередженнями. Якщо у вас є нейронна мережа, де попередній шар не впливає на різні параметри різних прихованих шарів, тобто нейронна мережа не має пам'яті, тоді ви можете використовувати рекурентну нейронну мережу.

Рекурентна нейронна мережа стандартизує різні функції активації, ваги та зміщення, щоб кожен прихований шар мав однакові параметри. Потім, замість створення кількох прихованих шарів, він створить один і повторить його стільки разів, скільки потрібно.

Розділ 3. Програмна реалізація

3.1 Використані технології

У ході розробки LeNet-CNN моделі, я використовував мову програмування Python. Через широкий вибір бібліотек, які облегшують роботу з моделями, ця мова є однією з найкращих мов для роботи з машинним навчанням. Ось кілька бібліотек, які використовувались для програмної реалізації:

Pillow – це одна з найпопулярніших та зручних бібліотек для роботи із зображеннями. Вона включає в себе різні функції, які дозволяють накладати різні фільтри на зображення, змінювати кут нахилу.

Pandas – ця бібліотека використовується для роботи з даними. Вона дозволяє досліджувати, аналізувати та маніпулювати даними.

OpenCV – ця бібліотека дозволяє нам працювати з машинним навчанням. Вона надає нам багато алгоритмів та інструментів, які використовуються машинним навчанням. Також ця бібліотека має кілька модулів:

- Opencv_ml – пропонує методи та моделі машинного навчання

- OpenMP і TBB – для розпаралелення
- Opencv_imgproc – допомагає з обробкою зображень
- Opencv_video – використовується для аналізу відео
- Opencv_objdetect – для розпізнавання образів з зображень

NumPy – це пакет для обробки масивів загального призначення. Він надає високопродуктивний багатовимірний об’єкт масиву та інструменти для роботи з цими масивами. Це основний пакет для наукових обчислень на Python.

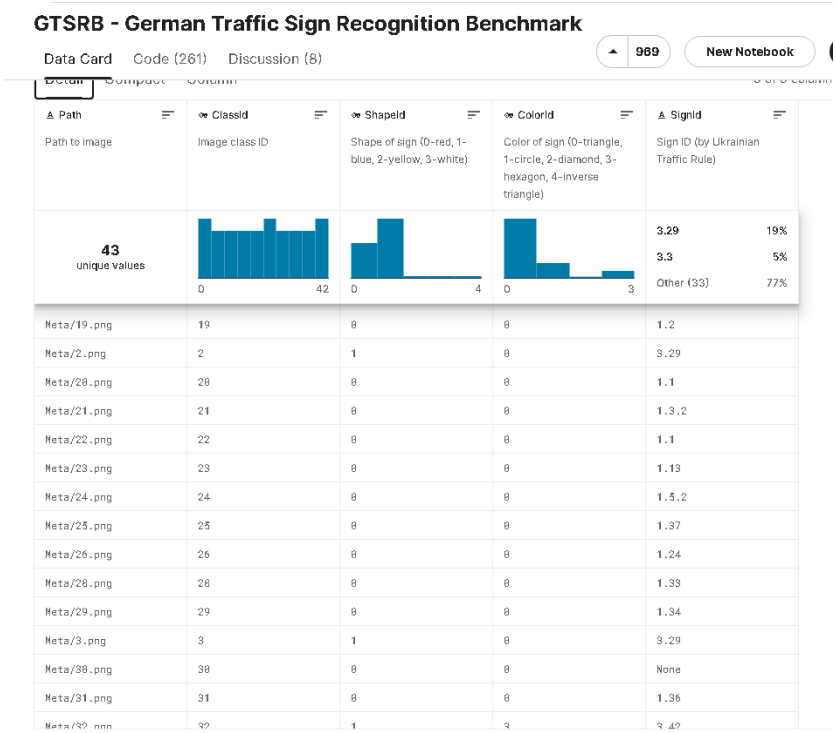
Для побудови моделі глибинного навчання я використав такі бібліотеки як TensorFlow та Keras. TensorFlow – ця бібліотека розроблена компанією Google. Вона дозволяє нам розробляти та тренувати нейронні мережі. Використовуючи tf.keras, ми можемо розробляти та використовувати моделі глибокого навчання.

Для створення веб-сайту, я також обрав мову Python, а саме, фреймворк Flask. Я обрав саме цей фреймворк, через те, що він є модульним, ефективним та зручним засобом для розробки веб-застосунків. Для надання сайту зовнішнього вигляду, я використав мову розмітки HTML, та мову CSS, яка необхідна для надання стилів.

3.2 Опис програмної реалізації

3.2.1 Огляд даних

Для реалізації моєї програми, я використав датасет GTSRB(German Traffic Sign Recognition Benchmark) із сайту Kaggle. Обрав його через розмір – 50 тис. образів і розподіл їх на класи. Також, під час проведення аналізу датасету, замінив тестові зображення для покращення роботи алгоритму з фотографіями поганої якості.



GTSRB - German Traffic Sign Recognition Benchmark

Data Card Code (261) Discussion (8) 989 New Notebook

Path Path to image	ClassId Image class ID	ShapeId Shape of sign (0-red, 1-blue, 2-yellow, 3-white)	ColorId Color of sign (0-triangle, 1-circle, 2-diamond, 3-hexagon, 4-inverse triangle)	SignId Sign ID (by Ukrainian Traffic Rule)
43 unique values	0 42	0 4	0 3	3.29 19% 3.3 5% Other (33) 77%
Meta/19.png	19	0	0	1.2
Meta/2.png	2	1	0	3.29
Meta/20.png	20	0	0	1.1
Meta/21.png	21	0	0	1.3.2
Meta/22.png	22	0	0	1.1
Meta/23.png	23	0	0	1.13
Meta/24.png	24	0	0	1.5.2
Meta/25.png	25	0	0	1.37
Meta/26.png	26	0	0	1.24
Meta/28.png	28	0	0	1.33
Meta/29.png	29	0	0	1.34
Meta/3.png	3	1	0	3.29
Meta/30.png	30	0	0	None
Meta/31.png	31	0	0	1.36
Meta/32.png	32	1	3	3.42

Рис 3.1 Вигляд класифікації алгоритму

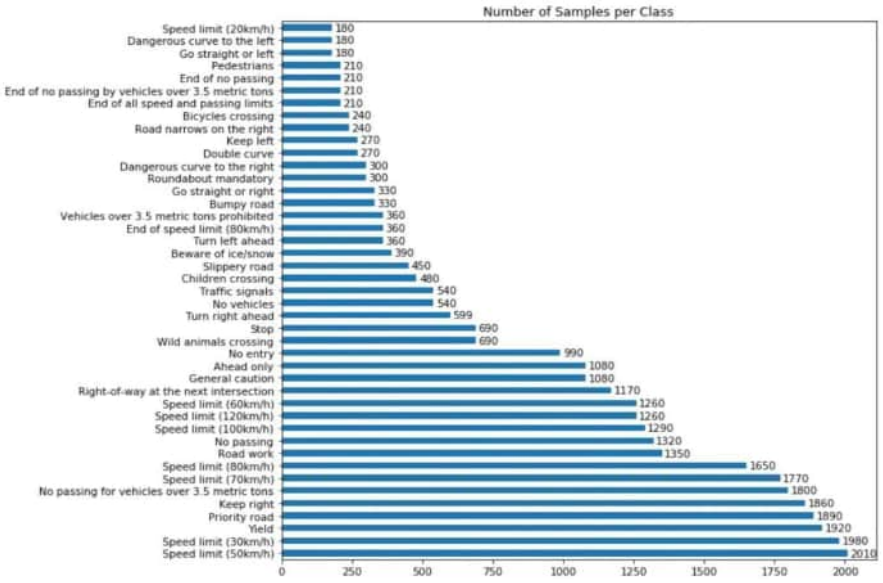


Рис 3.2 Розподіл зображень по класах

Цей датасет включає 43 класи дорожніх знаків, розділених на навчальні та тестові зображення. Всі зображення мають різний фон, різні ракурси зйомки та умови освітлення.

Рис 3.3 Приклади тестових зображень

Рис 3.4 Приклади навчальних зображень

3.2.2 Попередня обробка зображень

Провівши огляд даних, я вирішив змінити яскравість і контрастність деяких зображень, тобто спотворити зображення. Це реалізовано бібліотекою Pillow, мови програмування Python. Все це виконується для того, щоб модель краще розпізнавала зображення з поганою якістю.

Рис 3.5 Приклади спотворених зображень

Для кожного із зображень є анотація у якій вказано шлях до зображення, розміри зображення, клас зображення та 2 пари координат – координати обмежувального прямокутника (Рис.3.5) у якому знаходиться знак на зображенні.

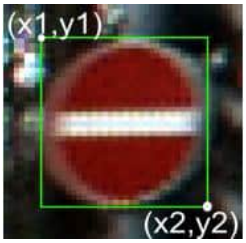


Рис 3.5 Координати обмежувального прямокутника

Також під час обробки даних, код читає зображення з кількох каталогів класів, обрізає зображення по координатах обмежувального прямокутника і змінює їх розмір до 30x30 пікселів, для кращої точності, після цього перетворює їх на масиви NumPy, і зберігає оброблені зображення пов’язуючи їхні дані з відповідними категоріями. Крім цього обробляються винятки, які можуть виникнути під час обробки зображень і повідомляє про помилки, щоб була можливість їх усунути.

3.2.3 Розробка методу детекції дорожнього знаку

Перед тим як реалізовувати сам алгоритм розпізнавання образів, потрібно дати програмі змогу відрізняти дорожні знаки на фоні інших непотрібних предметів. Для цього було використано покращений метод Віола-Джонса, він відрізняється від оригінального тим, що він дозволяє працювати з кольоровими зображеннями. Цей метод є реалізованим в бібліотеці OpenCV інструментом opencv_traincascade. Та для його використання потрібно мати вибірку зображень, в якій об’єкт, який повинен розпізнаватись повинен бути кольоровий, а весь залишковий фон у чорно-білому кольорі. Після створення такої вибірки було запущено метод і реалізовано метод детекції дорожніх знаків на зображеннях.

Рис 3.6 Приклад зображення для використання метода Віола-Джонсона

Рис 3.7 Приклад вдалої детекції

3.2.4 Огляд створеного алгоритму

Алгоритм можна розбити на такі пункти:

- Попередня обробка даних
- Створення масивів з отриманих даних.

Для створення масивів використовувалась бібліотека NumPy,

- Розбиття даних на навчальні і тестові вибірки

При розбитті данних значно допомогла бібліотека Scikit-image, а саме її інструмент train_test_split

- Перетворення даних до унітарного коду

Перетворення виконувалось з функцією .to_categorical, яка відноситься до keras

- Створення LeNet-CNN моделі та її компіляція завдяки алгоритму оптимізації Adam

Для створення моделі було використано модель “Sequential”, вона дозволяє послідовно додавати шари до мережі.

Перший і другий шари “Conv2D” з 32 фільтрами, розміром ядра 5x5 і функцією активації “relu”.

Наступний шар “MaxPool2D” з пудлінговим розміром 2x2, використовувався для зменшення розмірності зображення.

Далі шар “Dropout” з рейтом 0.25 для регуляризації і для уникнення перенавчання.

Після цього ще два шари “Conv2D” з 64 фільтрами і ядром 3x3.

Ще раз зменшуємо розмірність зображення через шар “MaxPool2D” і додаємо “Dropout” з таким ж рейтом і перетворюємо попередній шар у вектор завдяки шару “Flatten”.

Далі додаємо два шари “Dense”, перший з 256 нейронами, другий з 43 та функцією активації “softmax”, і між ними використовуємо шар “Dropout” з рейтом 0.5.

- Тренування моделі, визначення точності та втрат даних

Компіляція моделі відбувається з використанням функції втрати “categorical_crossentropy” , оптимізатора “Adam” та метрики “accuracy”.

Навчання нейронної мережі проводилось на 20 епохах, а також з функцією “fit”.

- Тестування на тестовій вибірці даних

Тестування починається з завантаження та читання тестових даних за допомогою “pandas”, вони перероблюються і завдяки навченій моделі і функції “predict_classes” виконується розпізнавання і зберігається результат.

У нейронній мережі використовувалось чергування згорткових шарів та шарів субдискретизації, на виході отримувалися повнозв’язні шари. На всіх шарах в ролі функції активації використовувалась ReLU.

3.3 Огляд структури проекту

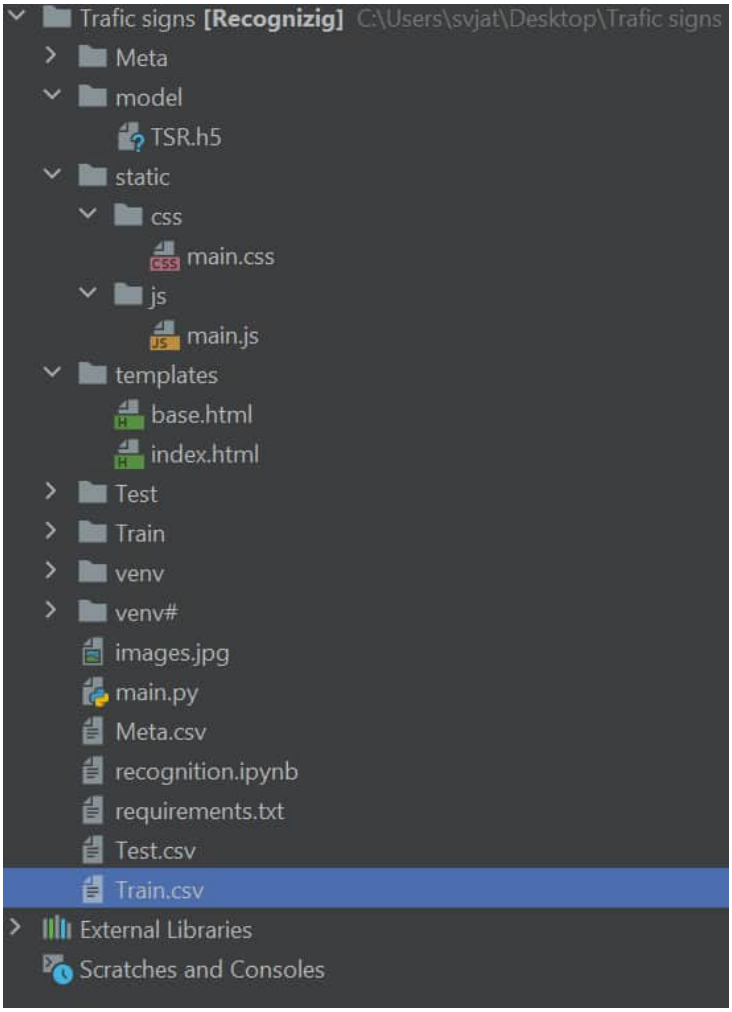


Рис 3.8 Структура проекта

- Папка Meta містить в собі 43 зображення дорожніх знаків, по одному на кожну категорію, яку ми будемо розпізнавати.
- У папці model знаходиться наша натренована нейронна мережа.
- Директорії Static та templates відповідають за інтерфейс для користувача. У static є файли на мовах Javascript та CSS, у templates – HTML.
- Папка Test містить велику кількість оброблених фото, на яких і тестувалась нейрона мережа.
- У папці train знаходяться посортовані по категоріях зображення, на яких тренувалась мережа.

- Директорія training є місцем куди зберігаються дані і категорії під час попередньої обробки, а під час навчання використовуються.
- Файл main.py відповідає за роботу веб-сайту.
- Файл .ipynb формату використовується для навчання та створення моделі.

3.4 Клієнтська частина

Мій веб-сайт розроблявся так, щоб користувачу було максимально просто використовувати функцію розпізнавання дорожніх знаків. Тому користувачу не потрібно реєструватись, створювати свій профіль, чи переходити на інші сторінки. Все реалізовано на одній основній сторінці, де знаходиться кнопка Upload. Нажимаючи на неї, користувачу пропонує завантажити фотографію, з якої він хоче розпізнати дорожній знак.

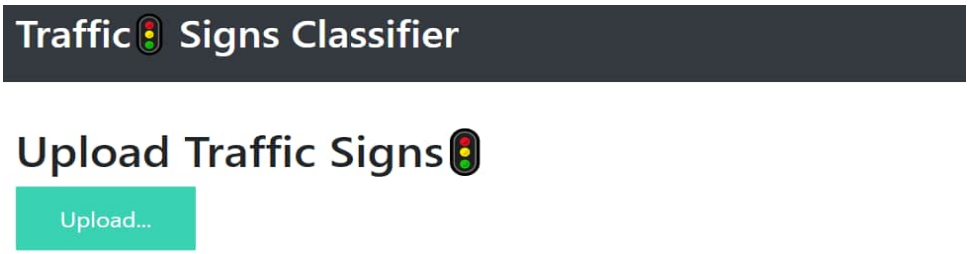


Рис 3.9 Інтерфейс веб-сайту

Після того, як користувач загрузив фотографію, появляється кнопка Predict Traffic Signs, нажимаючи на яку і запускається функція розпізнавання дорожнього знаку.

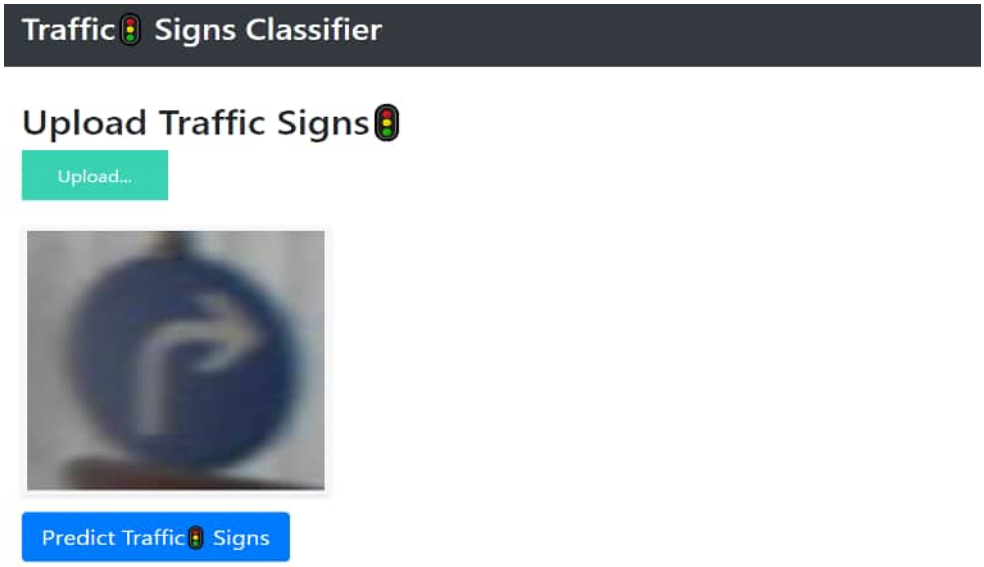


Рис 3.10 Поява фотографії і кнопки для розпізнавання дорожнього знаку

Рис 3.11 Приклади розпізнаного зображення

Висновок

У цій роботі досліджується розпізнавання дорожніх знаків за допомогою методів машинного навчання. Для досягнення поставленої мети проаналізовано існуючі методи та алгоритми, які використовуються в цій галузі.

У роботі було визначено набір знаків для розпізнавання та було зібрано достатньо даних для навчання та тестування моделі. Використовувалися різні методи попередньої обробки даних, такі як зміна розміру зображення, зміна контрастності та нормалізація зображень.

Отримані результати показують, що використання методів машинного навчання є ефективним і перспективним підходом для розпізнавання дорожніх знаків. Це може бути дуже важливо для систем безпеки дорожнього руху та безпілотних автомобілів. Використання цих методів може допомогти підвищити безпеку на дорозі та зменшити ймовірність аварій.

Однак варто враховувати, що успіх моделі може залежати від якості та різноманітності навчальних даних. Інші дослідження можуть включати розширення набору дорожніх знаків для більш точного розпізнавання, а також вдосконалення алгоритмів попередньої обробки даних і підвищення швидкості обробки зображень.

Загалом результати цієї роботи демонструють сильний потенціал методів машинного навчання для розпізнавання дорожніх знаків і підтверджують їх важливість для підвищення безпеки дорожнього руху.

Список використаних джерел

1. Deisenroth M.P. Mathematics for Machine Learning/M. P. Deisenroth, Published by Cambridge University Press, 2020.
2. Нікольський Ю.В. Системи штучного інтелекту. Навчальний посібник / Ю.В. Нікольський – Львів, видавництво «Магнолія –2006», 2013.
3. Simplilearn | Online Courses – Bootcamp & Certification Platform // Simplilearn. Available from: https://www.simplilearn.com/
4. Kaggle: Your Home for Data Science // Kaggle. Available from: https://www.kaggle.com
5. Lutz M. Learning Python / Mark Lutz., 2013. – (5th Edition).
6. MacDonald M. Creating a Website: The Missing Manual / Matthew MacDonald. – 1005 Gravenstein Highway North, Sebastopol,: O'Reilly Media, Inc., 2011.