Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
(повна назва)

Кафедра Автоматизації проектування обчислювальної техніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)
(рівень вищої освіти)

Інтелектуальна система попередження аварійних транспортних ситуацій на основі нейромережі
(тема)

Виконав: студент 2 курсу, групи СКСм-22-2
Курченко О. В.
(прізвище, ініціали)
Спеціальність *123 Комп'ютерна інженерія*
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма *Спеціалізовані комп'ютерні системи*
(повна назва освітньої програми)
Керівник роботи доцент Рожнова Т. Г.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Чумаченко С.В.
(підпис) (прізвище, ініціали)

2023 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління

Кафедра Автоматизації проектування обчислювальної техніки

Рівень вищої освіти другий (магістерський)

Спеціальність 123 Комп'ютерна інженерія
(шифр і назва)

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

Освітня програма Спеціалізовані комп'ютерні системи
(повна назва

ЗАТВЕРДЖУЮ:

Зав.кафедри _____
(підпис)

"____" _____ 20__ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Курченко Олегу Вадимовичу
(прізвище, ім'я, по батькові)

1. Тема роботи (проекту) Інтелектуальна система попередження аварійних транспортних ситуацій на основі нейромережі
затверджена наказом по університету від "03" листопада 2023 р. № 1282 Ст.
2. Термін подання студентом роботи до екзаменаційної комісії 26 січня 2024 р.
3. Вихідні дані до роботи (проекту)

Інтелектуальні системи попередження аварій на дорогах

Машинне та глибоке навчання

Сучасні методи дослідження

Мови програму Python/C++

Фреймворки Tensorflow/OpenCV/ OpenNN

4. Перелік питань, що потрібно опрацювати у роботі

Аналіз предметної області

Постановка задачі

Визначення методів швидкої обробки даних

Вибір платформи та інструментів для реалізації програмного інстумента

Тестування програми та аналіз результатів

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) 16 слайдів .pptx

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

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

7. Дата видачі завдання 15 жовтня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання, узгодження і затвердження	15.10.2023 – 17.10.2023	
2	Аналіз проблемної галузі	17.10.2023 – 25.10.2023	
3	Пошук літератури з теми	25.10.2023 – 27.10.2023	
4	Аналіз об'єкту дослідження	27.10.2023 – 01.11.2023	
5	Розробка програмних інструментів	01.11.2023 – 21.11.2023	
6	Проведення експериментів	21.11.2023 – 15.12.2023	
7	Аналіз отриманних результатів	15.12.2023 – 01.01.2024	
8	Оформлення пояснювальної записки	01.01.2023 – 16.01.2024	
9	Перевірка виконаного проекту керівником	16.01.2024 – 20.01.2024	
10	Рецензування, допуск до захисту	20.01.2024 – 26.01.2024	
11	Захист перед екзаменаційною комісією	26.01.2024 – 26.01.2024	

Студент _____
(підпис)

Керівник роботи _____ доцент Рожнова Т. Г.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка містить 61 сторінки, 20 рисунків, 16 джерел за переліком посилань.

ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ, МАТЕМАТИЧНІ МОДЕЛІ, НЕЙРОННІ МЕРЕЖІ, СИСТЕМИ РЕАЛЬНОГО ЧАСУ, PYTHON, ОБРОБКА ВЕЛИКИХ ДАНИХ, ТЕСТОВЕ СЕРЕДОВИЩЕ, МОВА C/C++

Об’єкт дослідження – процес виявлення транспортних засобів на дорозі.

Мета – дослідження та вдосконалення ефективних моделей машинного навчання щодо виявлення транспортних засобів та попередження аварійних ситуацій на дорозі з подальшим їх застосуванням та оптимізацією.

Досліджено собливості роботи інтелектуальних систем попередження аварійних транспортних ситуацій що є важливими для забезпечення безпеки на дорозі. Досліджено методи теорії машинного навчання та обробки великих даних у реальному часі та методи обробки realtime зображення за допомогою штучного інтелекту та проаналізовані основні математичні моделі та алгоритми нейронних мереж: Transformer, CNN модель та RCNN алгоритм, GAN модель, HOG та YOLO алгоритми, проведено їх порівняння.

Наукова новизна – запропонована модель інтелектуальної системи, використовуючи глибоке навчання для аналізу навколишнього середовиш що підвищує точність та ефективність інтелектуальної системи.

Практична значимість – завдяки аналізу запропонаваних моделей машинного навчання, була встановлена ефективна модель машинного навчання по визначенню транспортних засобів на дорозі. Результати дослідження спрямовані на підвищення якості та ефективності процесів розпізнання транспортих засобів на дорозі.

ABSTRACT

Thesis contains contains 61 pages, 20 figures, 16 sources according to the list of links.

INTELLIGENT SYSTEMS, MATHEMATICAL MODELS, NEURAL NETWORKS, REAL-TIME SYSTEMS, PYTHON, BIG DATA PROCESSING, GUI INTERFACE, C/C++ PROGRAMMING LANGUAGE

Research Object: The process of detecting vehicles on the road.

Objective: To investigate and improve effective machine learning models for vehicle detection and accident prevention on the road, followed by their application and optimization. The study explores the peculiarities of intelligent systems for preventing traffic accidents, which are crucial for road safety. Methods of machine learning theory and real-time big data processing, as well as image processing methods using artificial intelligence, have been investigated. The main mathematical models and algorithms of neural networks, including Transformer, CNN model, RCNN algorithm, GAN model, HOG, and YOLO algorithms, have been analyzed and compared.

Scientific novelty: A new machine learning model integrated into an intelligent system is proposed, using deep learning for environmental analysis, which enhances the accuracy and efficiency of the intelligent system.

Practical significance: Through the analysis of the proposed machine learning models, an effective model for determining vehicles on the road was identified. The research results aim to improve the quality and efficiency of vehicle recognition processes on the road. The qualification work consists of analyzing the subject area, developing software for research, and integrating new technologies and methods based on the research results.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AI – Штучний Інтелект (англ. Artificial Intelligence)

ML – Машинне Навчання (англ. Machine Learning)

RL – Навчання з Підкріпленням (англ. Reinforcement Learning)

ANN – Штучні Нейронні Мережі (англ. Artificial Neural Networks)

CNN – Згорткова Нейронна Мережа (англ. Convolutional Neural Networks)

RNN – Рекурентна Нейронна Мережа (англ. Recurrent Neural Networks)

HOG – Гістограма Направлених Градієнтів (англ. Histogram Oriented Gradients)

IS – Інтелектуальні системи (англ. Intelligent Systems)

ВСТУП

Інтелектуальні системи попередження аварійних транспортних ситуацій на основі нейромережі стає новbv сучасним викликом перед суспільством, охороною здоров'я та гігантами автомобільної промисловості. Використання технологій штучного інтелекту, сучасних потужних мікропроцесорів, а також властивостей комп'ютерних систем, надає можливість розробки інтелектуальної системи виявлення та запобігання дорожньо-транспортним пригодам.

Обробка великого масиву даних у реальному часі – це комплексна задача для системи виявлення та запобігання дорожньо-транспортним пригодам і нейронні мережі є ключовим компонентом цієї інтелектуальної системи. Але часто, працюючи з обробкою великого об’єму даних, інтелектуальні системи також стикаються з викликами та обмеженнями. Це може включати складність навчання моделей, вимогливість до обчислювальних ресурсів, необхідність якісних та різноманітних даних, а також питання етики та безпеки.

З одного боку, CNN автоматично виділяють ієрархічні ознаки із вхідних даних, що робить їх ефективними у задачах класифікації, детектування об'єктів та сегментації зображень.

З іншої сторони, DNN здатні вивчати складні та абстрактні уявлення даних. Завдяки своїй глибокій архітектурі вони знаходять застосування в широкому спектрі завдань, включаючи обробку зображень, обробку природної мови та рекомендаційні системи.

Трансформерні нейронні мережі також є привабливими для обробки великих обсягів даних. Трансформерні нейронні мережі повністю покладаються на механізм уваги. Механізм уваги дозволяє трансформерам зосередитися на важливих частинах вхідних даних та враховувати їх у обчисленнях.

Кожен тип нейронної мережі має свої переваги та обмеження, і вибір відповідної моделі нейронної мережі залежить від конкретного завдання та характеристик даних. Використання правильного типу нейронної мережі може значно підвищити ефективність і точність моделі, що робить вибір відповідного типу нейронної мережі важливим кроком у роботі інтелектуальних систем.

Класичні архітектури машинного навчання можуть вирішувати нелінійні задачі. Тому було висунуто завдання для визначення найефективнішої моделі машинного навчання для обробки великого масиву даних у реальному часі у контексті роботи інтелектуальної системи попередження аварійних транспортних ситуацій на дорозі. Для цього було розроблене програмне середовище з метою проведення досліджень щодо поведінок різних архітектур моделей машинного навчання у реальному часі та сформулювання теоретичних висновків для налаштування спеціалізованої нейронної мережі.

1 АНАЛІЗ ПРЕМЕТНОЇ ОБЛАСТІ

1.1 Актуальність дослідження

Актуальність дослідження полягає в необхідності вдосконалення ефективності моделей машинного навчання для точного виявлення транспортних засобів та передбачення аварійних ситуацій на дорозі.

Об’єкт дослідження є процес виявлення транспортних засобів на дорозі.

Метою дослідження є вдосконалення ефективних моделей машинного навчання щодо виявлення транспортних засобів та попередження аварійних ситуацій на дорозі з подальшим їх застосуванням та оптимізацією.

Дослідження проводилось на базі прикладної програми. Процесом дослідження було визначення моделями машинного навчання транспортних засобів та можливістю попередження аварійних транспортних ситуацій на дорозі у режимі реального часу.

Результатами дослідження є метрики ефективності моделей машинного навчання у визначенні транспортних засобів та можливістю попередження аварійних транспортних ситуацій на дорозі у режимі реального часу. Результати спрямовані на підвищення якості та ефективності процесів розпізнання транспортих засобів на дорозі.

Практична цінність результатів дослідження є встановлення ефективних моделей машинного навчання по визначенню транспортних засобів на дорозі.

1.2 Інтелектуальні системи, як явище

Інтелектуальні системи (ІС) – це програмні або апаратні засоби, призначені для виконання складних завдань, що вимагають інтелектуальних здібностей. Вони спрямовані на імітацію та розширення можливостей людського інтелекту.

У сучасному світі ІС (рисунок 1.1) знаходять широке застосування у різних галузях. Вони можуть навчатися на основі даних та алгоритмів, аналізувати інформацію, приймати рішення та взаємодіяти з людьми.

Рисунок 1.1 – Приклад майботньої уніфікованої інтелектуальної системи, впровадженної у всьому світі

Один із поширених типів ІС – експертні системи. Вони використовують знання та досвід експертів у певній галузі для розв'язання складних проблем. Експертні системи можуть надавати рекомендації та приймати рішення на основі логічних правил та факторів.

Ще один важливий тип ІС – системи машинного навчання [1], що навчаються на основі даних та алгоритмів для розпізнавання шаблонів, виявлення залежностей та здійснення прогнозів. Системи машинного навчання широко застосовуються в медицині, фінансах, рекламі та інших галузях, де потрібен аналіз великого обсягу даних.

Нейронні мережі (NN) є ще одним типом ІС, які моделюють роботу людського мозку. Вони складаються з з'єднаних та взаємодіючих штучних нейронів, здатних обробляти інформацію та адаптуватися на основі досвіду. Нейронні мережі застосовуються в розпізнаванні образів, обробці природної

мови, комп'ютерному зорі та інших галузях, де потрібна складна обробка інформації.

Інтелектуальні системи мають значний потенціал для поліпшення нашого життя та розвитку різних галузей. Вони можуть допомагати приймати рішення, оптимізувати процеси, автоматизувати завдання та підвищувати ефективність. Однак із розвитком ІС також виникають питання етики та безпеки, які потребують уваги та регулювання.

У майбутньому застосування інтелектуальних систем охоплюватиме широкий спектр галузей та сфер нашого життя. Ми можемо очікувати використання таких систем у медицині, де вони допомагатимуть у діагностиці захворювань, аналізі медичних зображень та виборі оптимальних методів лікування. У транспортній галузі інтелектуальні системи будуть використовуватися для покращення безпеки та оптимізації руху. Вони допоможуть водіям приймати рішення на основі даних про стан доріг та прогнозів погоди.

У фінансовій сфері інтелектуальні системи можуть застосовуватися для аналізу фінансових даних, прогнозування трендів на ринку та управління інвестиціями. Вони допоможуть у прийнятті рішень щодо інвестування, оптимізації фінансових стратегій та запобіганні шахрайству. У промисловості інтелектуальні системи будуть використовуватися для оптимізації виробничих процесів, контролю якості та планування ресурсів. Вони допоможуть покращити ефективність та надійність виробництва.

У сфері освіти інтелектуальні системи можуть персоналізувати навчання, оцінювати знання студентів та розробляти індивідуальні навчальні плани. Вони допоможуть покращити якість освіти та зробити її більш доступною. У галузі обслуговування клієнтів інтелектуальні системи будуть використовуватися для надання швидкої та якісної підтримки через чат-ботів та віртуальних асистентів.

Завдяки постійному вдосконаленню алгоритмів, доступності великого обсягу даних та розвитку обчислювальної потужності, ІС будуть відігравати

все більш важливу роль у багатьох сферах нашого життя. Вони стануть невід'ємною частиною нашого суспільства, вносячи в нього нові можливості та поставляючи питання, з якими нам доведеться вміти впоратися.

1.3 Інтелектуальні системі попередження аварійних транспортних ситуацій

Інтелектуальні системи попередження аварійних транспортних ситуацій мають значний потенціал для поліпшення безпеки дорожнього руху та запобігання аваріям. Вони використовуються для виявлення потенційно небезпечних ситуацій на дорозі та надання водіям вчасних попереджень та рекомендацій.

Ці системи можуть базуватися на різних технологіях та датчиках, таких як радари, камери, лазерні сканери та інші [2]. Вони здатні виявляти перешкоди на дорозі, небезпечні маневри і неналежну поведінку інших учасників руху. На основі зібраних даних інтелектуальна система аналізує ситуацію та здійснює швидку оцінку ризиків.

У разі виявлення потенційної аварійної ситуації, інтелектуальна система може активувати різноманітні заходи попередження. Наприклад, водію можуть відправити звукові або візуальні сигнали, або навіть автоматично зробити екстрене гальмування або ухил.

Деякі інтелектуальні системи попередження аварійних ситуацій також використовують підключені мережі і обмін даними між транспортними засобами. Це дозволяє створити колективну свідомість на дорозі і отримувати актуальну інформацію про небезпеку з інших автомобілів.

Один із науспішніших прикладів системи, що використовує нейронні мережі для попередження аварійних ситуацій, це система AutoPilot (рисунок 1.2) від компанії Tesla.

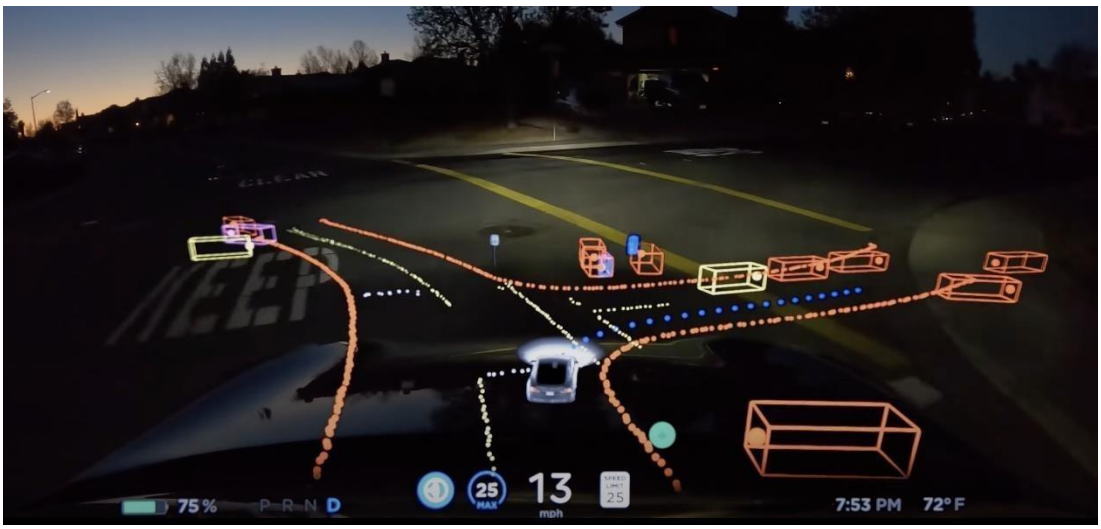

Рисунок 1.2 – Система AutoPilot компанії Tesla

AutoPilot використовує нейронні мережі для обробки інформації, отриманої від різних датчиків та камер автомобіля. Нейронна мережа аналізує вхідні дані, які включають відеопотоки, дані з радарів та інших датчиків, і визначає дорожню ситуацію, включаючи розпізнавання об'єктів, дорожніх знаків та ліній розмітки.

За допомогою навчання на великому обсязі даних, нейронна мережа здатна розпізнавати та класифікувати різні об'єкти на дорозі, такі як автомобілі, пішоходи, велосипедисти та перешкоди. Вона також може аналізувати рух об'єктів та передбачати їх майбутні позиції та поведінку.

На основі цієї інформації нейронна мережа приймає рішення щодо безпечного управління автомобілем. Вона визначає оптимальну швидкість та відстань до інших об'єктів, контролює рух автомобіля, включаючи керування кермом та гальмами, і попереджає водія про можливі небезпеки.

Окрім системи AutoPilot компанії Tesla, існують інші приклади інтелектуальних систем попередження аварійних ситуацій: Mobileye, Mobileye Shield+, Waymo, Nvidia DRIVE, Volvo City Safety та інші.

Ці приклади демонструють, що штучний інтелект [3], як модель, використовується для розробки систем попередження аварійних ситуацій, що забезпечують безпеку на дорозі та знижують ризик аварій. ШІ допомагає автоматизувати процеси аналізу даних та прийняття рішень, що дозволяє

транспортним засобам швидше реагувати на зміни в дорожній ситуації та забезпечувати безпеку для водіїв та пішоходів.

1.4 Важливість розробки та вдосконалення інтелектуальних систем для забезпечення безпеки на дорозі

Розробка та вдосконалення інтелектуальних систем для забезпечення безпеки на дорозі має велику важливість, особливо з огляду на статистику аварійності та її наслідки.

За даними Всесвітньої організації охорони здоров'я (ВООЗ), щорічно відбувається мільйони дорожньо-транспортних пригод, що призводять до великої кількості травмованих і загиблих. Дослідження показують, що ці системи можуть зменшити кількість аварій через неправильну дистанцію, некеровані зміни смуги руху та інші фактори.

Наприклад, за даними Національної адміністрації з безпеки дорожнього руху США (NHTSA), системи попередження зіткнень, такі як ISAAN та ADAS (рисунок 1.3), можуть знизити кількість зіткнень зі збитками на 27%, а системи автоматичного гальмування – на 50%. Дослідження Європейського парламенту також вказують на позитивні результати використання систем допомоги водію у зменшенні аварійності на дорогах.

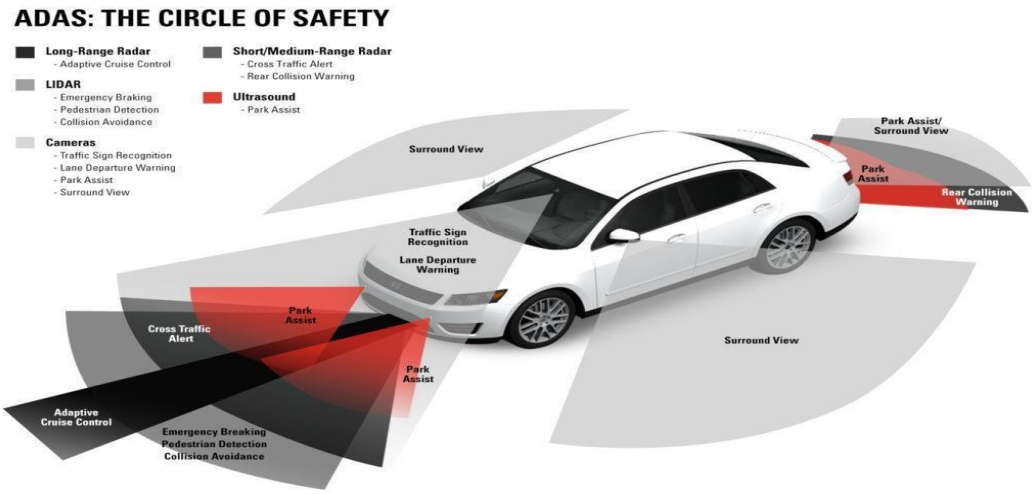

Рисунок 1.3 – Приклад роботи системи ADAS

За даними Європейської комісії, системи розпізнавання пішоходів можуть знизити кількість аварій із пішоходами на 60%. Вони виявляють наявність пішоходів перед автомобілем та надають акустичні або візуальні сигнали водію для уникнення зіткнення.

А за даними Автомобільної асоціації Америки (AAA), системи контролю точки слідкування можуть знизити аварії через неправильну дистанцію на 23%. Вони моніторять відстань до впереді рухаючогося автомобіля та надають водію попередження, якщо він наближається занадто близько.

Однією з головних причин дорожньо-транспортних пригод є людський фактор, помилки водіїв та порушення правил дорожнього руху. Інтелектуальні системи можуть виявляти та прогнозувати небезпечні ситуації на дорозі та надавати водіям вчасні попередження та рекомендації для запобігання аваріям. Наприклад, система розпізнавання дорожніх знаків може сповістити водія про швидкісний обмежувач або заборону обгону, що допоможе уникнути порушень правил дорожнього руху.

Також, інтелектуальні системи можуть допомагати уникнути зіткнень з іншими транспортними засобами або пішоходами. Застосування передових технологій, таких як системи радару та камер у зв’язці зі штучним інтелектом допомагає виявляти об'єкти на дорозі та вчасно реагувати на них. Наприклад, система автоматичного гальмування може автоматично активувати гальма, якщо виявлено перешкоду перед автомобілем, що допомагає уникнути зіткнення або зменшити його наслідки.

Отже, розробка та вдосконалення інтелектуальних систем для забезпечення безпеки на дорозі є необхідним кроком для зменшення аварійності та захисту життя та здоров'я людей. Використання штучного інтелекту в цих системах додає ще більше переваг та можливостей. Ці системи зможуть ефективно виявляти небезпеку та допомагати водіям уникнути аварійних ситуацій, що забезпечує безпечнішу дорожню рух.

1.5 Базова архітектура інтегрованої інтелектуальної системи

Параметричний аналіз дорожньої ситуації на основі оперативного збору даних від автомобіля дає можливість у режимі on-line оптимально оцінювати дорожню ситуацію базуючись на готових моделях поведінки транспортного засобу [4]. Формування взаємодії "автомобіль – система" створює два типи нових відносин (рисунок 1.4):

- впроваджена система з блоком керування автомобілем;
- апаратні датчики автомобіля з впровадженою системою.

Перетворення засобами інтегрованих алгоритмів системи даних з апаратних датчиків дозволяють скласти кінцеву функцію детермінованої оцінки дорожньої ситуації в одиницю часу:

$$f_m(t) = \Sigma_n{}^k{}_{=0} = f_n(e),$$

де f_m – функція оцінки дорожньої ситуації в одиницю часу; f_n – функція вихідного сигналу параметра; e – зважена сума вхідного сигналу.

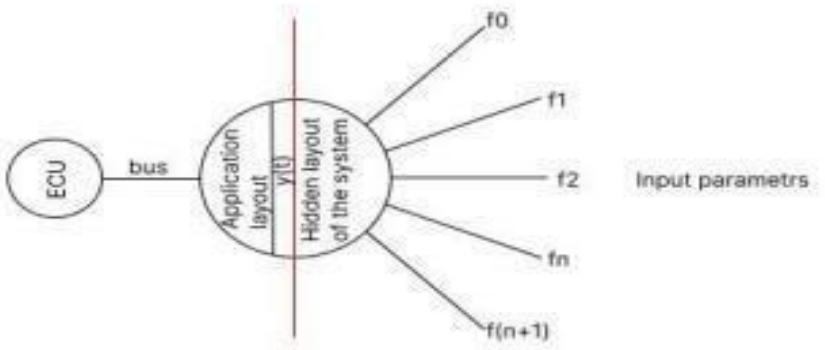

Рисунок 1.4 – Взаємодія системи та транспортного засобу

Архітектура інтелектуальних систем, вбудованих у сучасні автомобілі, представляє собою високоінтегрований комплекс технологічних рішень, спрямованих на забезпечення безпеки, комфорту та автоматизації водіння. У цій архітектурі взаємодіють різні компоненти, включаючи електронні блоки

керування автомобілем, системи безпеки, засоби комфорту та розваг, а також сучасні системи допомоги водію (ADAS).

Електронні блоки керування автомобілем (ECU) включають моторний блок (ECU), який оптимізує роботу двигуна, та блок керування трансмісією (TCU), що відповідає за автоматичне керування трансмісією. Ці компоненти спрямовані на підтримку оптимальної ефективності та екологічності роботи автомобіля.

Блок управління ходовою частиною включає системи, такі як антиблокувальна система гальм (ABS) та система динамічної стабілізації (ESC), які сприяють безпеці та стабільності автомобіля під час руху. Також важливим елементом є адаптивне керування підвіскою, яке спрямоване на оптимізацію якості керування та комфорту пасажирів.

Системи безпеки включають подушки безпеки, які активуються відповідно до сил зіткнення, а також системи запобігання зіткнення, що використовують датчики для виявлення перешкод та можливостей автоматичного гальмування чи уникнення зіткнення.

У сфері комфорту та розваг використовуються клімат-контроль та системи кондиціювання, а також аудіо- та інфотейнмент-системи для забезпечення розваг та інформаційних сервісів для пасажирів.

Системи допомоги водію (ADAS) включають розпізнавання знаків та рухомих об'єктів, що допомагає водієві слідкувати за дорожніми знаками та учасниками руху, а також системи автопілоту, які сприяють автоматизації керування автомобілем.

До інтелектуальних систем також входять різноманітні сенсори та датчики, такі як радари, лідари, камери та ультразвукові датчики, які забезпечують системам автомобіля необхідні дані для виявлення оточуючого середовища, розпізнавання об'єктів та прийняття рішень в реальному часі.

2 МЕТОДИ МАШИННОГО НАВЧАННЯ ТА МОДЕЛІ НЕЙРОННИХ МЕРЕЖ

2.1 Методи машинного навчання

Машинне навчання (ML) є галуззю, що займається розробкою та вдосконаленням методів, які дозволяють комп'ютерам "навчатися" на основі даних з метою поліпшення їх продуктивності у виконанні різноманітних завдань. Алгоритми машинного навчання створюють моделі, використовуючи навчальні дані, що дозволяють здійснювати прогнози та приймати рішення без явного програмування. Машинне навчання [5] застосовується у багатьох галузях, включаючи медицину, електронну пошту, розпізнавання мовлення, сільське господарство та комп'ютерне зорове сприйняття. Це особливо корисно в ситуаціях, де традиційні алгоритми не ефективні або неможливі для вирішення задач. Машинне навчання використовується як узгоджено з обчислювальною статистикою, так і самостійно. Воно також пов'язане з інтелектуальним аналізом даних, який зосереджується на використанні неконтрольованого навчання для дослідження даних. Деякі реалізації машинного навчання використовують дані та штучні нейронні мережі, щоб імітувати роботу людського мозку.

Методи машинного навчання можна поділити на декілька основних категорій (рисунок 2.1).

Навчання з вчителем (Supervised Learning). Метод використовує мітки або позначення для навчання моделі. Модель отримує вхідні дані та відповідні вихідні мітки, і в процесі навчання вона намагається знайти залежності між вхідними та вихідними даними. Прикладами методів навчання з вчителем є лінійна регресія, логістична регресія, метод опорних векторів (SVM) та нейронні мережі.

Навчання без вчителя (Unsupervised Learning). Метод не використовую міток або позначень. Модель намагається знайти приховані закономірності чи структуру в невизначених даних. Прикладами методів навчання без вчителя є кластеризація (наприклад, K-середніх алгоритм), зменшення розмірності (наприклад, метод головних компонент) та правила асоціації.

Навчання з підсиленням (Reinforcement Learning). Метод використовується для навчання моделі при взаємодії з динамічною середовищем. Модель взаємодіє з середовищем, виконуючи дії, і отримує зворотний зв'язок у вигляді нагород або покарань. Вона навчається приймати оптимальні рішення, які максимізують загальну нагороду. Прикладом методу навчання з підсиленням є Q-навчання.

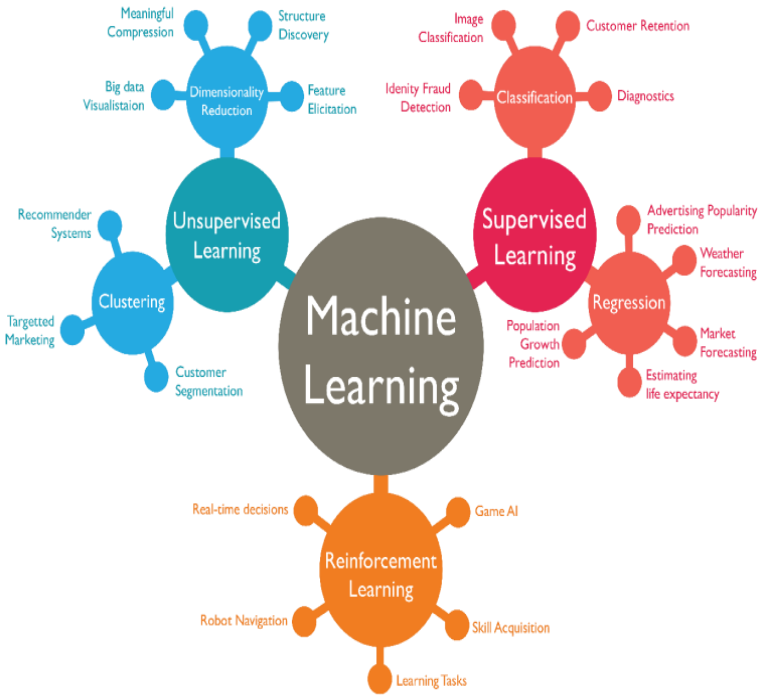

Рисунок 2.1 – Основні категорії машинного навчання

Півнаглядне навчання (Semi-Supervised Learning): Цей метод поєднує елементи навчання з вчителем та без вчителя. Він використовує навчальні дані з позначеними та непозначеними зразками для навчання моделі. Це дозволяє використовувати обмежену кількість позначених даних, використовуючи невизначені дані для знаходження додаткової інформації про структуру даних.

Змішане навчання (Hybrid Learning). Цей метод поєднує різні підходи до навчання, такі як навчання з вчителем, без вчителя та підсилення. Він використовується для вирішення складних завдань, де потрібно враховувати різноманітні аспекти та типи даних.

Машинне навчання є потужним інструментом, але воно також стикається зі складностями. Перша складність полягає в необхідності мати достатню кількість даних для ефективного навчання моделей. Збір, підготовка і анотування даних можуть бути викликом, особливо якщо дані обмежені або складнодоступні.

Друга складність пов'язана з вибором оптимального алгоритму. Існує багато різних алгоритмів машинного навчання, і вибір найкращого для конкретної задачі може бути важким завданням. Кожен алгоритм має свої переваги і обмеження, і важливо вибрати той, що найкраще підходить для потреб задачі.

Третя складність полягає у здатності моделі машинного навчання до ефективного прогнозування на нових, невідомих даних. Часто моделі показують високу точність на навчальних даних, але виявляються менш ефективними на реальних даних. Це вимагає розробки моделей, які можуть генералізувати свої знання на нові дані і працювати ефективно.

Четверта складність пов'язана з обробкою великих обсягів даних. Обробка великих обсягів даних вимагає потужних обчислювальних ресурсів і ефективних алгоритмів. Реалізація швидкої обробки даних в реальному часі може бути складною, особливо якщо дані постійно змінюються.

Остання складність пов'язана з етичними питаннями, які виникають при використанні машинного навчання. Автоматизовані системи прийняття рішень можуть мати великий вплив на суспільство, і важливо забезпечити прозорість, відповідальність і контроль при їх розробці та використанні.

Машинне навчання має великий потенціал і може вирішувати складні проблеми. Проте, розв'язання цих складностей вимагає наукових досліджень, інновацій та постійного вдосконалення. Розробка ефективних алгоритмів, збір

і підготовка якісних даних, здатність до генералізації та вирішення етичних питань є ключовими аспектами розвитку машинного навчання.

2.2 Архітектура нейронних мереж

Нейронні мережі, або конективістські системи (ANN) – це обчислювальні системи, натхнені біологічними нейронними мережами, що складають мозок тварин. Такі системи навчаються задач (поступально покращують свою продуктивність на них), розглядаючи приклади, загалом без спеціального програмування під задачу.

Нейронні мережі ґрунтуються на сукупності з'єднаних вузлів, що називають штучними нейронами (аналогічно до біологічних нейронів у головному мозку тварин). Кожне з'єднання (аналогічне синапсові) між штучними нейронами може передавати сигнал від одного до іншого. Штучний нейрон, що отримує сигнал, може обробляти його, й потім сигналізувати штучним нейронам, приєднаним до нього.

В поширених реалізаціях нейронних мереж (рисунок 2.2) сигнал на з'єднанні між штучними нейронами є дійсним числом, а вихід кожного штучного нейрону обчислюється нелінійною функцією суми його входів. Штучні нейрони та з'єднання зазвичай мають вагу, яка підлаштовується в перебігу навчання. Вага збільшує або зменшує силу сигналу на з'єднанні. Штучні нейрони можуть мати такий поріг, що сигнал надсилається лише якщо сукупний сигнал перетинає цей поріг. Штучні нейрони зазвичай організовано в шари. Різні шари можуть виконувати різні види перетворень своїх входів. Сигнали проходять від першого (входового) до останнього (виходового) шару, можливо, після проходження шарами декілька разів.

Первинною метою підходу нейронні мережі було розв'язання задач таким же способом, як це робив би людський мозок. З часом увага зосередилася на відповідності певним розумовим здібностям, ведучи до відхилень від біології.

Штучні нейронні мережі використовували в ряді різноманітних задач, включно з комп'ютерним баченням, розпізнаванням мовлення, машинним перекладом, соціальномережевим фільтруванням, грою в настільні та відеоігри, та медичним діагностуванням.

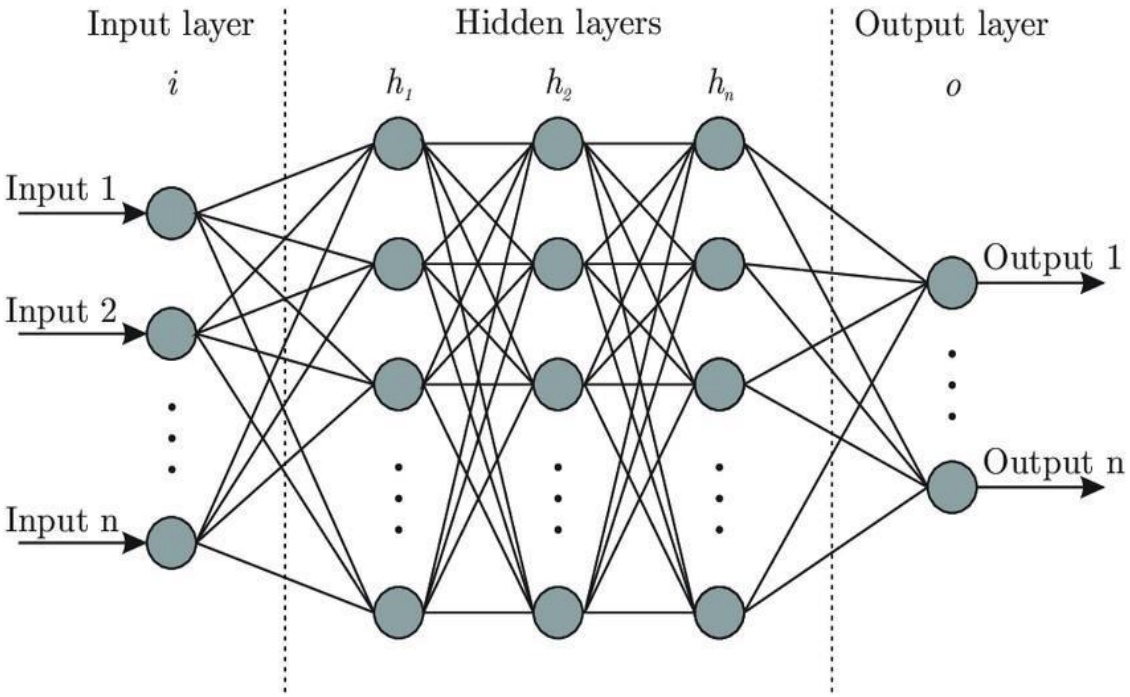

Рисунок 2.2 – Приклад базової реалізації нейронної мережі

Архітектура нейронних мереж – це структура і організація нейронних зв'язків у нейронній мережі. Вона визначає, як нейрони взаємодіють між собою і які шляхи передачі і обробки інформації відбуваються всередині мережі. Архітектура нейронних мереж [6] визначається конкретними завданнями, для яких вона призначена, і може мати різні форми і розміри, в залежності від складності задачі і доступних даних.

Нейронні мережі можуть мати різні архітектурні типи (рисунок 2.3). Одним з найпоширеніших типів є персептрон, який складається з вхідного шару, прихованих шарів і вихідного шару. Вхідний шар отримує вхідні дані, приховані шари здійснюють обчислення, а вихідний шар генерує вихідні результати. Інші типи архітектур включають згорткові нейронні мережі, рекурентні нейронні мережі, автокодери та багатошарові перцептрони.

Крім основних типів архітектур, існують також спеціалізовані архітектури, призначені для вирішення конкретних задач. Наприклад, в машинному зорі використовуються мережі з великою кількістю шарів та зв'язків для розпізнавання об'єктів на зображеннях. У природних мовах використовуються рекурентні нейронні мережі для обробки послідовностей слів.

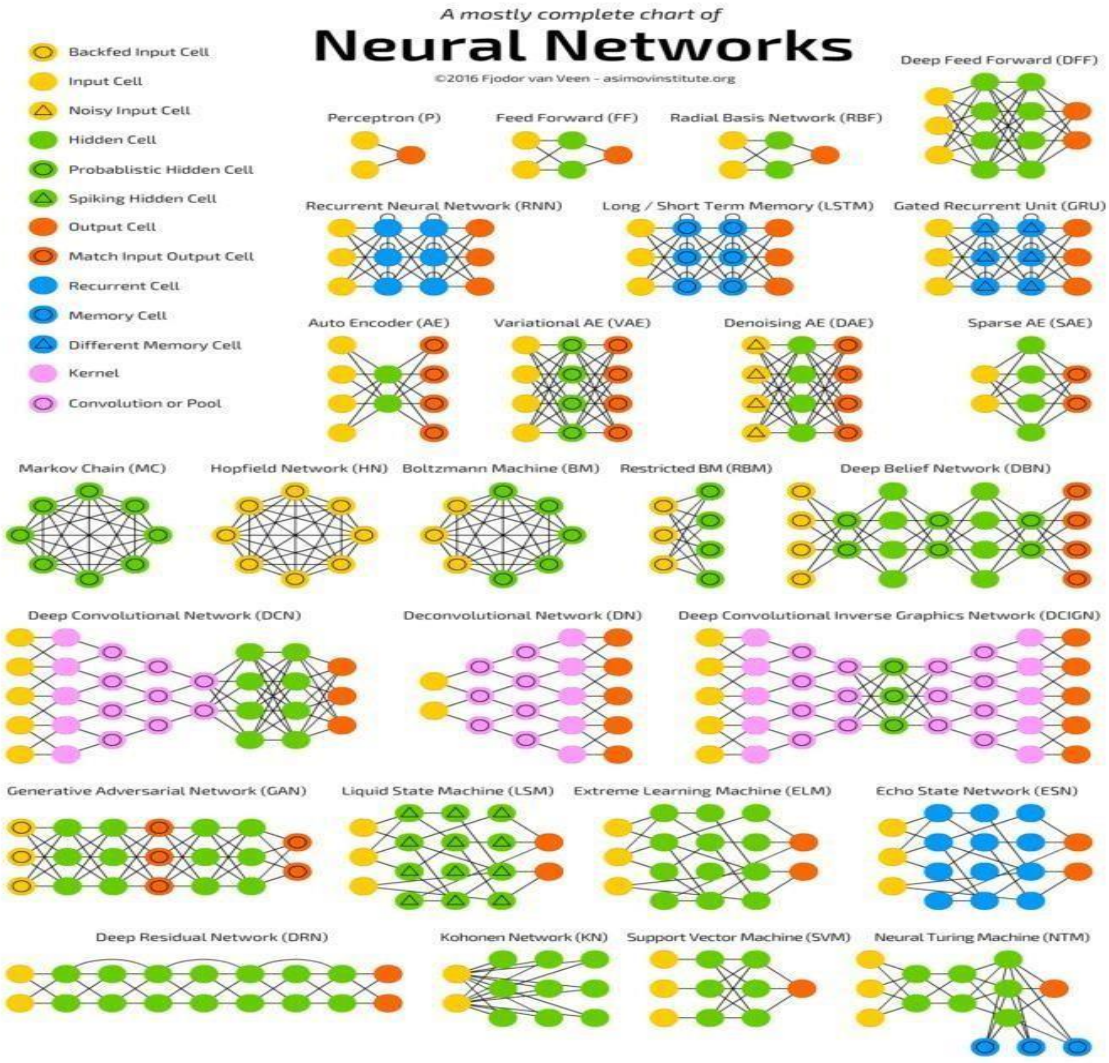

Рисунок 2.3 – Типи нейронних мереж

Основні типи нейронних мереж включають:

– персептрон. Це один з найпростіших типів нейронних мереж, який складається з одного або декількох шарів нейронів, які повністю з'єднані між собою. Використовується для бінарної класифікації або регресії;

– згорткова нейронна мережа (Convolutional Neural Network, CNN). Цей тип мережі ефективно працює з великими обсягами даних, такими як зображення. Вона має спеціальні шари згортки, які дозволяють виявляти локальні особливості на зображенні і розпізнавати об'єкти;

– рекурентна нейронна мережа (Recurrent Neural Network, RNN). Цей тип мережі має зв'язки, які утворюють зворотний зв'язок і дозволяють передавати інформацію від попередніх кроків до наступних. Використовується для обробки послідовних даних, таких як мовлення або текст;

– багатошаровий перцептрон (Multilayer Perceptron, MLP). Це тип нейронної мережі, яка має кілька шарів нейронів, включаючи приховані шари. Використовується для розпізнавання складних залежностей між вхідними та вихідними даними.

Архітектура нейронної мережі визначається в процесі її розробки і може бути підгонена для досягнення бажаних результатів. Вибір архітектури залежить від завдання, обсягу даних, ресурсів обчислювальної системи та інших факторів. Оптимальна архітектура може бути знайдена за допомогою експериментів, налаштування гіперпараметрів та аналізу результатів.

Нейронні мережі можна розглядати з математичного погляду як графові моделі (рисунок 2.4), що складаються зі сполучених вузлів (нейронів) та вагованих зв'язків між ними. Кожен нейрон отримує вхідні дані, виконує обчислення та передає результат наступному нейрону. Обчислення в нейроні зазвичай включають лінійну комбінацію вхідних сигналів з вагами та застосування нелінійної активаційної функції.

Математично, вхідні дані в нейрон можна представити як вектор вхідних значень $x = (x_1, x_2, ..., x_n)$, а ваги зв'язків між нейронами – як матрицю ваг W. Лінійна комбінація вхідних сигналів з вагами обчислюється як добуток матриці ваг на вектор вхідних значень: $z = Wx + b$, де b – вектор зсуву (bias).

Після цього на результат застосовується нелінійна активаційна функція, така як сигмоїда, гіперболічний тангенс, ReLU (Rectified Linear Unit) тощо. Ця

функція надає нелінійності та нелінійного зміщення у вихідному сигналі нейрону.

Нейронні мережі зазвичай організовані у шари, де кожен шар містить набір нейронів, а вихідні значення одного шару є вхідними для наступного. Вхідний шар отримує вхідні дані, останній шар називається вихідним шаром, а проміжні шари – прихованими шарами. Така структура шарової нейронної мережі називається багатошаровим перцептроном.

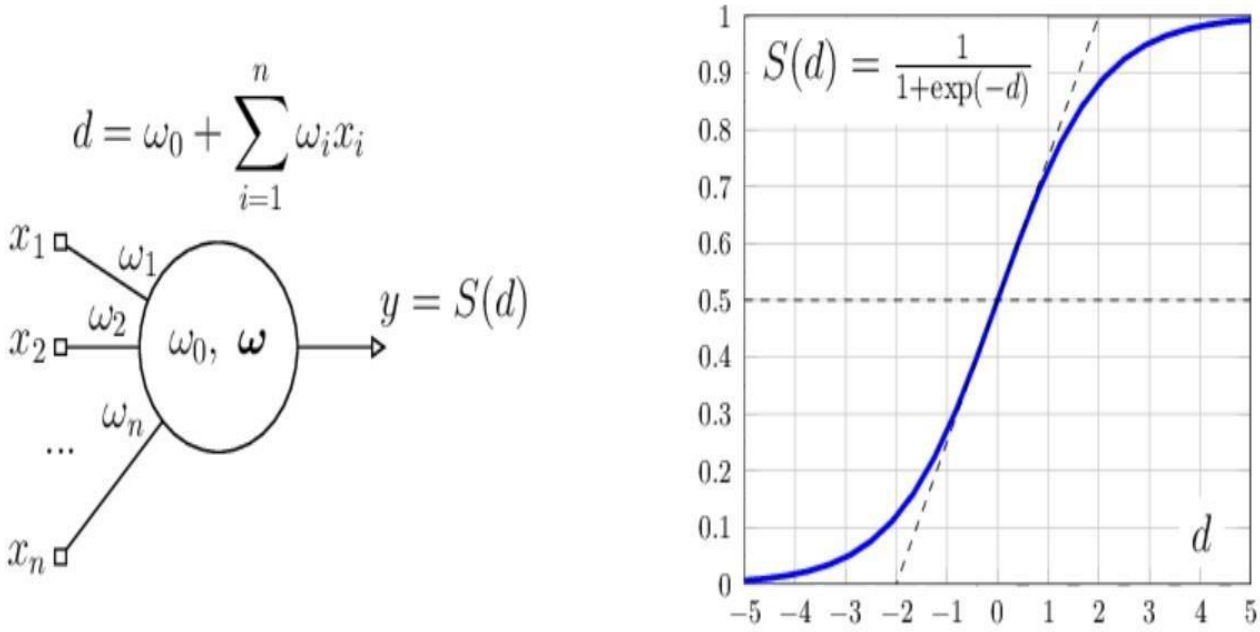

Рисунок 2.4 – Математична модель нейронної мережі

Для навчання нейронних мереж використовуються алгоритми оптимізації, такі як зворотне поширення помилки (backpropagation), які змінюють ваги та зсуви мережі, щоб зменшити помилку між прогнозованими та очікуваними результатами. Цей процес повторюється на багатьох тренувальних прикладах до досягнення достатньої точності.

В загальному розумінні, нейронні мережі є математичними моделями, які застосовуються для вирішення різноманітних завдань машинного навчання, використовуючи лінійні та нелінійні обчислення для обробки вхідних даних та здійснення прогнозів або прийняття рішень.

Нейронні мережі використовують різні алгоритми для тренування та роботи з даними. Основні алгоритми, що використовуються в нейронних мережах, включають:

- зворотне поширення помилки (Backpropagation). Цей алгоритм використовується для тренування нейронних мереж, шляхом зміни ваг та зсувів мережі на основі помилки між прогнозованими та очікуваними результатами. Він передає помилку від вихідного шару до прихованих шарів, оновлюючи ваги зв'язків за допомогою методу градієнтного спуску;
- алгоритм градієнтного спуску (Gradient Descent). Цей алгоритм використовується для мінімізації функції втрати в нейронних мережах шляхом пошуку оптимальних значень ваг та зсувів. Він використовує похідні функції втрати по вагам та зсувам для оновлення їх значень у напрямку спуску градієнту;
- алгоритми оптимізації. У додаток до градієнтного спуску, існують різні алгоритми оптимізації, такі як адам (Adam), RMSprop, адаптивний градієнтний спуск (Adagrad) та інші, які використовуються для ефективного тренування нейронних мереж шляхом адаптації швидкості навчання та управління градієнтом;
- алгоритми розпізнавання образів. Ці алгоритми, такі як згорткові нейронні мережі (Convolutional Neural Networks, CNN), використовуються для розпізнавання та аналізу образів. Вони використовують фільтри для виявлення певних ознак в зображеннях та забезпечують високу точність у завданнях комп'ютерного зору;
- алгоритми згортки та пулінгу. Використовуються у згорткових нейронних мережах для екстракції ознак з вхідних зображень шляхом згортки фільтрів та зведення до більш малих представлень за допомогою операцій пулінгу;
- рекурентні нейронні мережі (Recurrent Neural Networks, RNN). Ці мережі використовуються для роботи з послідовними даними, такими як текст або часові ряди. Вони мають зв'язки, що повертаються, що дозволяє їм

запам'ятовувати попередні стани та використовувати їх для передачі інформації в майбутніх кроках.

Різноманітність алгоритмів дозволяє нейронним мережам розв'язувати різні типи завдань та досягати високої продуктивності в багатьох областях застосування.

2.3 Модель згорткових нейронних мереж

Згорткова нейронна мережа – архітектура штучних нейронних мереж, запропонована Яном Лекуном в 1988 і націлена на ефективне розпізнавання образів, входить до складу технологій глибокого навчання (deep learning) . Використовує деякі особливості зорової кори, в якій були відкриті так звані прості клітини, що реагують на прямі лінії під різними кутами, та складні клітини, реакція яких пов'язана з активацією певного набору простих клітин. Таким чином, ідея згорткових нейронних мереж (рисунок 2.5) полягає в чергуванні згорткових шарів та субдискретизуючих шарів. Структура мережі – односпрямована, важливо багатошарова. Для навчання використовуються стандартні методи, найчастіше метод зворотного розповсюдження помилки. Функція активації нейронів – будь-яка, на вибір дослідника.

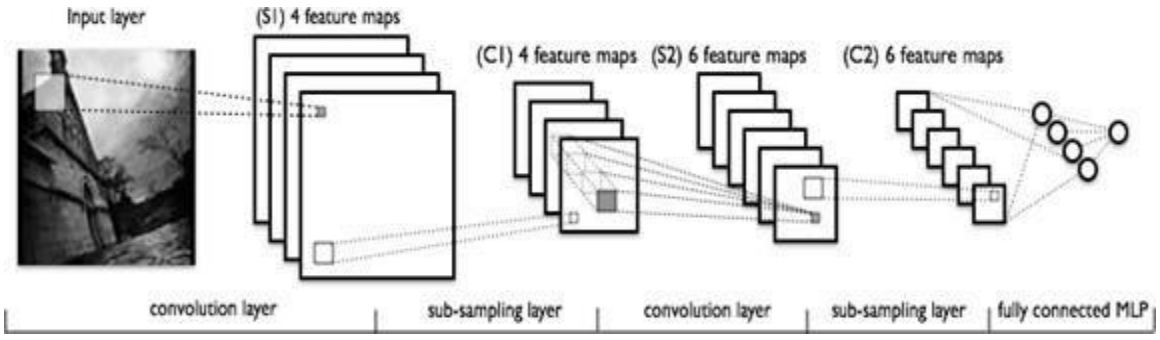

Рисунок 2.5 – Згорткова нейронна мережа

Назву архітектура мережі отримала через наявність операції згортки, суть якої в тому, що кожен фрагмент зображення множиться на матрицю

згортки поелементно, а результат підсумовується та записується в аналогічну позицію вихідного зображення.

Архітектура згорткової нейронної мережі [7] базується на концепції згортки (convolution) та пулінгу (pooling). Основні компоненти сверткової нейронної мережі включають:

– шар згортки (Convolutional Layer). Цей шар використовує фільтри (ядра), які просуваються по вхідному зображенню, здійснюючи згортку. Згортка виконує операцію перемноження значень пікселів зображення з вагами фільтра, формуючи вихідне представлення з важливими ознаками зображення;

– шар пулінгу (Pooling Layer). Цей шар виконує операцію зведення (downsampling) вихідного представлення, зменшуючи його розмір та кількість параметрів. Це допомагає знизити обчислювальну складність та забезпечує ротаційну та масштабну інваріантність ознак;

– шар повнозв'язаних нейронів (Fully Connected Layer). Після шару згортки та шару пулінгу можуть бути додані один або кілька шарів повнозв'язаних нейронів, які здійснюють класифікацію або регресію на основі отриманих ознак;

– функції активації. В мережі також використовуються функції активації, такі як ReLU (Rectified Linear Unit), для введення нелінійності у модель та забезпечення здатності нейронної мережі вирішувати складні задачі.

Однією з ключових особливостей згорткових мереж є їхній здатність до розпізнавання просторових залежностей у вхідних даних, таких як зображення або відео. Згорткові шари в мережі використовують фільтри, які здійснюють локальну згортку зображення. Це дозволяє виявляти різні особливості, такі як краї, форми та текстури, на різних рівнях абстракції. Наприклад, перші шари можуть виявляти прості краї та текстурні ознаки, тоді як наступні шари можуть розпізнавати складніші форми або об'єкти.

Ще одна важлива характеристика згорткових мереж – це їхня здатність до роботи зі змінними просторовими розмірами вхідних даних. Згорткові

мережі можуть працювати з відео, що мають різну роздільну здатність або кількість кадрів на секунду, шляхом застосування згорткових операцій до кожного кадру окремо. Це дозволяє ефективно аналізувати великий обсяг відеоданих та виявляти об'єкти та дії в режимі реального часу.

Окрім того, згорткові мережі можуть бути поєднані з рекурентними мережами (Recurrent Neural Networks, RNN) для аналізу послідовностей відеокадрів і встановлення зв'язків між ними в часі[8]. Це особливо корисно для виявлення руху та тривіальних змін у відеоданих.

2.4 Трансформерні нейронні мережі

Трансформер (Transformer) є типом нейронної мережі, який зарекомендував себе у багатьох задачах обробки природної мови та машинного перекладу. Ця архітектура, запропонована в 2017 році, відмінна від традиційних рекурентних та згорткових нейронних мереж тим, що використовує механізми самоуваги (self-attention) для здійснення операцій над послідовностями даних.

Головною особливістю трансформера є його здатність до моделювання довготривалих залежностей у послідовностях. Завдяки механізму самоуваги, він може приділяти увагу різним частинам вхідних даних на основі їхнього значення та контексту. Це дозволяє трансформеру ефективно аналізувати послідовності різної довжини та встановлювати складні залежності між їхніми елементами.

Архітектура трансформера складається з двох основних компонентів: енкодера і декодер. Енкодер приймає на вхід послідовність даних та генерує їхнє внутрішнє представлення, а декодер використовує це представлення для генерації вихідної послідовності. Кожен компонент має кілька повторюваних шарів, які виконують механізми самоуваги та підсумовування інформації.

Трансформер заснований на ідеї атенції (attention), де кожен елемент послідовності має змогу звертати увагу на інші елементи залежно від їхнього

значення. Цей механізм дозволяє моделі зосередитися на релевантних аспектах даних та забезпечує кращу обробку контексту.

Трансформери, спочатку розроблені для обробки послідовних даних, таких як текст, знайшли застосування й у сфері обробки відео та зображень. Вони виявилися ефективними в задачах, де важлива просторова структура та залежності між пікселями або об'єктами.

Для обробки відео трансформери можуть бути застосовані до кадрів відеопотоку послідовно або паралельно. Кожен кадр може бути розглянутий як послідовність пікселів, а трансформер може моделювати залежності між ними. Використовуючи механізми самоуваги, трансформер може враховувати контекст та взаємодію між різними областями зображення для вирішення завдань, таких як класифікація, детекція об'єктів, сегментація та інші.

2.5 Порівняльний аналіз різних видів нейронних мереж у задачах обробки відео

Згорткові нейронні мережі (ЗНМ), рекурентні нейронні мережі (РНМ) та Трансформери є потужними інструментами у сфері глибинного навчання. Кожен з цих типів мереж має свої особливості і застосовується у різних задачах з обробки даних.

Хоча трансформери є новим типом нейронної мережі, який використовує механізми самоуваги (self-attention) для здійснення операцій над послідовностями даних, архітектура трансформерів не є ідеальним вибором для обробки відео з кількома кадрами з кількома причинами:

– велика обчислювальна складність. Трансформери вимагають значних обчислювальних ресурсів через свою характеристику самоуваги (selfattention), де кожен елемент вхідного сигналу взаємодіє з усіма іншими елементами. Це стає проблемою при обробці великих об'ємів відеоданих, оскільки кількість пар кадрів для взаємодії може бути надто великою, що призводить до високої обчислювальної витратності і значного збільшення часу обробки;

– нездатність до моделювання часових залежностей. Трансформери зазвичай не здатні ефективно захоплювати тимчасові залежності, які мають значення в відеоданих. Вони не мають властивості пам'яті, яка допомагає зберігати і використовувати інформацію про попередні кадри. Це може призвести до втрати контексту та порушення залежностей між кадрами, що впливає на якість обробки відео;

– розмір пам'яті. Трансформери мають потребу в значній кількості пам'яті для збереження усіх відносин між елементами вхідних даних. Це може бути непрактичним для великих відеоданих, які вимагають великої обсягу пам'яті для збереження всіх відносин між кадрами;

– обробка потоку даних. Трансформери не є природнім вибором для обробки відеопотоку, оскільки вони працюють з фіксованим розміром вхідних даних. Відео може мати різну довжину та варіації кадрів на різних етапах обробки. Це може ускладнити використання трансформерів для безперервної обробки відео.

Згорткові нейронні мережі (CNN) та рекурентні нейронні мережі (RNN) є двома популярними типами нейронних мереж, які використовуються для обробки відеоряду:

– якщо основний акцент зроблений на просторову інформацію та вилучення візуальних особливостей з окремих кадрів, згорткові нейронні мережі є сильним варіантом;

– якщо завдання полягає у захопленні тимчасових залежностей та розумінні послідовного характеру відеоданих, рекурентні нейронні мережі, зокрема варіанти, такі як довготривала короткочасна пам'ять (LSTM) або варіативний рекурентний блок (GRU), можуть бути більш підходящими;

– у деяких випадках використовується комбінація згорткових та рекурентних нейронних мереж, відома як згортково-рекурентні нейронні мережі (CRNN), щоб використовувати переваги обох архітектур. CRNN ефективно захоплюють як просторову, так і тимчасову інформацію, що робить їх потужними для різноманітних задач обробки відео.

Висновок в тому, що як згорткові нейронні мережі, так і рекурентні нейронні мережі, а також комбіновані згортково-рекурентні нейронні мережі можуть бути використані при обробці великого масиву даних.

2.6 Каскад Хаара

Каскад Хаара – це ефективна техніка виявлення облич та об'єктів на зображеннях, яка знаходить широке застосування у відеоспостереженні та обробці зображень. Основна перевага цієї техніки полягає в її швидкості розпізнанні об’єктів.

Перший етап використання Каскаду Хаара – навчання. Класифікатор навчається на позитивних та негативних зразках – зображеннях з обличчями чи об'єктами та зображеннях без них. Під час цього процесу класифікатор вивчає особливості, що характеризують певний об'єкт.

Після навчання створюється каскад (рисунок 2.6), що включає послідовність класифікаторів. Кожен класифікатор визначає конкретну особливість на зображенні, і вони використовуються послідовно для виявлення облич чи об'єктів.

Сам процес виявлення відбувається послідовно на різних масштабах та положеннях зображення. Якщо обличчя чи об'єкт виявляється на одному етапі, алгоритм переходить до наступного класифікатора. Якщо ж ні, то процес завершується.

У контексті нейронних мереж, Каскад Хаара може використовуватися як передфільтр для визначення областей інтересу (ROI) перед подальшою обробкою глибокими архітектурами, такими як Convolutional Neural Networks (CNN). Це дозволяє підвищити швидкість виявлення об'єктів та скоротити обсяг обробки глибокими мережами.

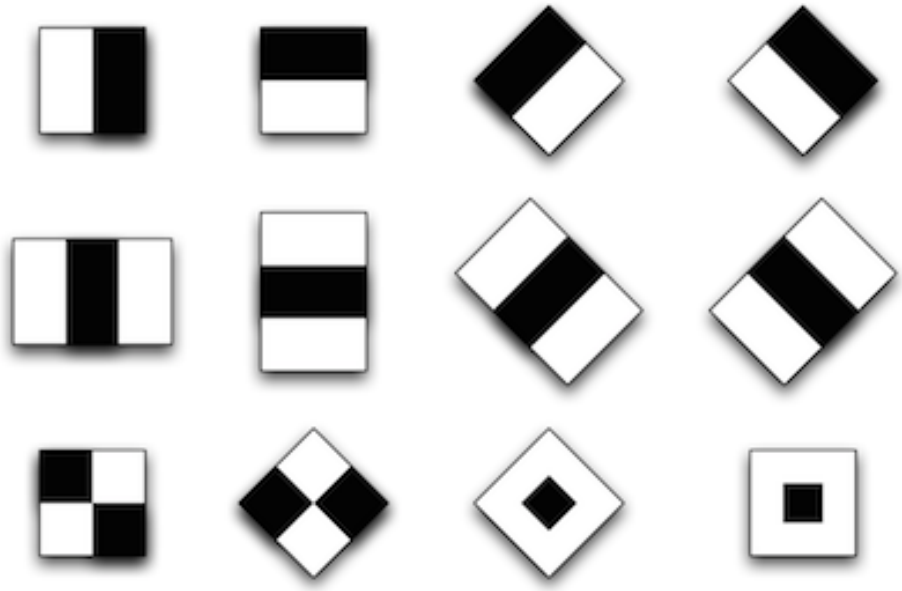

Рисунок 2.6 – примітиви Хаара

Каскад Хаара інтегрується з різними нейронними архітектурами для досягнення кращих результатів у завданнях виявлення об'єктів та облич на зображеннях. Узагальнюючи, ця техніка допомагає покращити продуктивність систем відеоспостереження та комп'ютерного зору.

2.7Архітектура Yolo

Інноваційна архітектура YOLO (You Only Look Once) визначає новий стандарт в галузі об'єктного виявлення на зображеннях та відео. Ця революційна технологія видокремлюється з-поміж існуючих рішень своєю ефективністю, точністю та здатністю виявляти об'єкти в реальному часі, а її переваги роблять її піонером у сфері комп'ютерного зору.

YOLO відрізняється тим, що весь процес виявлення об'єктів відбувається одночасно для всіх областей зображення (рисунок 2.7). Це робить архітектуру надзвичайно ефективною та швидкою, дозволяючи виявляти об'єкти в реальному часі без значних затримок.

YOLO займає унікальну позицію на ринку завдяки своїй універсальності та високій точності виявлення об'єктів різних класів. Інші рішення у сфері

об'єктного виявлення можуть стикатися з проблемами швидкодії, але YOLO забезпечує ідеальний баланс між швидкістю та точністю, підвищуючи результативність у всіх вимірах [9].

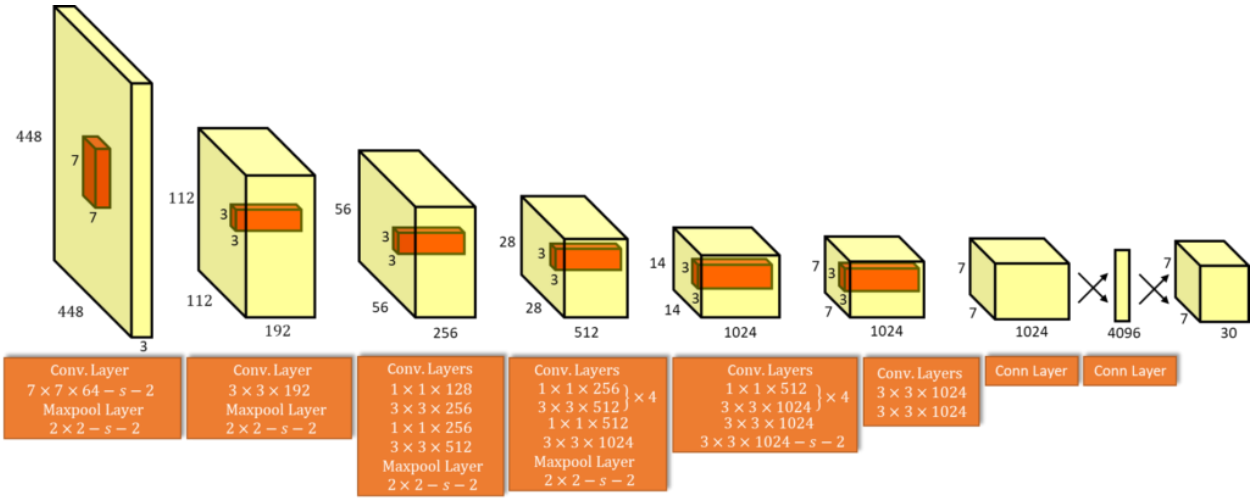

Рисунок 2.7 – Архітектура YOLO (version 1)

Основна концепція YOLO є простою: єдина згорткова нейронна мережа водночас передбачає кілька рамок обмеження і ймовірностей класів для цих рамок. YOLO вивчається на повних зображеннях і безпосередньо оптимізує ефективність виявлення.

3 ПОСТАНОВКА ЗАДАЧІ

3.1 Вимоги до прикладної програми

Графічне середовище повинно представляти Desktop application. Формат тестової програми з виявленням транспортних засобів та попередженням аварійних транспортних ситуацій на дорозі повинно представляти прикладну тестову програму. До прикладної тестової програми по виявленню транспортних засобів та попередженню аварійних транспортних ситуацій на дорозі висунуті наступні вимоги:

- підтримка ОС Windows та Linux;
- простий та зрозумілий UI;
- підтримка формату .mp4;
- обробка відеопотоку у реальному часі;
- підтримка функції Object-Detection та Object-Tracking;
- підтримка функції Collision Avoidance Warning System (Beta);
- відображення результатів Object-Detection та Object-Tracking у реальному часі;

У наступному розділі можна знайти ключові фрагменти програмного коду разом із короткими поясненнями їхньої роботи.

3.2 Засоби розробки програмного середовища

Засоби розробки – це комплекс інструментів, які використовують для створення програмного забезпечення. Цей комплект включає в себе певні мови програмування, бібліотеки, frameworks, компілятори, інтерпретатори, готові бінарні рішення та інші інструменти, які необхідні для ефективного процесу розробки програмного забезпечення.

Розробка даного програмного середовища включає в себе роботу з такими мовами програмування, як C/C++ та Python, фреймворками OpenCV, NumPy, TensorFlow, Keras, PyTorch та SDL.

3.2.1 Мови програмування C/C++

Мова програмування C вважається однією з найвпливовіших та найпопулярніших мов у світі програмування [10]. Вперше створена в 1972 році Денісом Рітчі у Белл Лабораторіях, C стала основою для багатьох інших мов і продовжує грати важливу роль у сфері розробки програмного забезпечення. Навіть зараз мова C має ключові переваги і її значення складає велику частку в сучасному програмуванні.

Використання мови програмування C/C++ у нашому проекті зумовлено наступними причинами:

- продуктивність. Мова C забезпечує низькорівневий доступ до пам'яті та близьке управління ресурсами, що може бути важливим для нашого проекту, де потрібна висока продуктивність;
- бібліотеки. Існуючі бібліотеки та фреймворки, які ми використовуємо, краще доступні та краще підтримуються в мові C;
- низький рівень абстракції. C має відносно низький рівень абстракції, що є корисним для проекту, так як нам потрібен повний контроль над кожною деталлю реалізації;
- оптимізований код. C дозволяє писати оптимізований та ефективний код, що важливо для обчислювально складного проекту.

3.2.2 Мова програмування Python

Мова програмування Python є високорівневою мовою, що визначається своєю простотою та елегантністю. Започаткована Гвідо ван Россумом у 1991

році, Python став популярним інструментом у світі розробки завдяки своїй зручності використання та великій кількості готових бібліотек.

Використання мови програмування Python у нашому проекті зумовлено наступними причинами:

− багата екосистема. Python має широкий вибір бібліотек і фреймворків для машинного навчання та нейромереж, таких як TensorFlow, PyTorch, scikit-learn і інші. Це дозволяє нам використовувати потужні інструменти для розробки та оптимізації машинного навчання;

− інтеграція з іншими мовами. Python легко інтегрується з іншими мовами програмування, що є корисно для оптимізації окремих частин системи, при використанні C/C++ для обчислювально витратних операцій;

− швидкість розробки. Використання Python сприяє швидкій розробці завдяки високому рівню абстракції, динамічній типізації та швидкому прототипуванню.

3.2.3 Simple DirectMedia Layer

Бібліотека Simple DirectMedia Layer (SDL) представляє собою програмне забезпечення, яке забезпечує доступ до різноманітних аспектів графічного та звукового введення-виведення в мові програмування C. Вона була розроблена для створення крос-платформених додатків, які вимагають обробки мультимедійних елементів. Однією з головних переваг SDL є її простота використання та можливість інтеграції з різними платформами без значних труднощів [11].

Використання SDL у графічній частині для Desktop-застосунку вказує на наступні особливості та вимоги до проекту:

− переносимість. SDL є крос-платформеним фреймворком, що дозволяє розробляти програму, яка працює як на ОС Windows, так і на Linux. Це гарантує зручність для розробки та тестування на різних платформах;

− простий та зрозумілий UI: SDL надає можливість створення

інтуїтивно зрозумілого та легко користуваного графічного інтерфейсу, що є важливим для швидкого навчання та прийняття програми користувачами.

3.2.4 Open Computer Vision (OpenCV)

OpenCV (Open Source Computer Vision Library) – це відкрита бібліотека, яка стала необхідним інструментом для розробників у сферах комп'ютерного зору, обробки зображень та машинного навчання. Заснована в 1999 році Гарі Бредслоу, OpenCV стала визнаним лідером завдяки своїм здатностям до виявлення облич, трекінгу об'єктів, розпізнавання образів, обробки відео та багатьох інших застосувань [12].

Використання OpenCV у цьому проекті обгрунтоване декількома ключовими причинами:

- функції виявлення та відстеження об'єктів. OpenCV має потужні функції виявлення та відстеження об'єктів, які можна легко використовувати для ідентифікації транспортних засобів та визначення їхнього руху;
- обробка відео в реальному часі. OpenCV має ефективні функції для обробки відеопотоку в реальному часі, що робить його ідеальним інструментом для системи, яка повинна виявляти транспортні засоби та попереджати про аварійні ситуації в реальному часі;
- підтримка Windows та Linux. Оскільки проект передбачає підтримку операційних систем Windows та Linux, OpenCV стає ідеальним інструментом, оскільки він має платформонезалежний характер.

4 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Розробка інтерфейса середовища

Інтерфейс програмного середовища базується на графічній бібліотеці – Simple DirectMedia Layer (SDL). Для роботи з нею, підключаємо її модулі у нашу збірку (лістинг 4.1):

Лістинг 4.1 – Модулі SDL

```
gcc -masm=intel -o program main.c program.c -lSDL2 -lSDL2_ttf
```

Бібліотека SDL дозволить нам створити базовий Graphical-User Interface (лістинг 4.2). Базова програма складається з простого інтерфейсу та трьох активних кнопок:

– start. Ця кнопка відповідає за запуск нашої програми та активацію коду мовою Python. Скрипт запускає інтерпретатор мови Python та передає йому аргументи для запуску обробки відео нейронною мережею;

– open video. Кнопка відповідає за завантаження відео до нашої програмної середи;

– exit. Миттєвий вихід з програми.

Лістинг 4.2 – Базові модулі програмного середовища

```
#include <SDL2/SDL.h>
#include <SDL2/SDL_image.h>
#include <SDL2/SDL_ttf.h>

int main(int argc, char ** argv)
{
    SDL_Init(SDL_INIT_TIMER | SDL_INIT_AUDIO | SDL_INIT_VIDEO |
SDL_INIT_EVENTS);
    IMG_Init(IMG_INIT_PNG | IMG_INIT_JPG) != (IMG_INIT_JPG |
IMG_INIT_PNG);
```

```
    TTF_Init();
    extern application_t application;

    application.window = SDL_CreateWindow("Car Detecting",
SDL_WINDOWPOS_CENTERED, SDL_WINDOWPOS_CENTERED,
APP_WINDOW_WIDTH, APP_WINDOW_HEIGHT, SDL_WINDOW_SHOWN);
    application.surface =
SDL_GetWindowSurface(application.window);
```

Кінцева програма має наступний вигляд (рисунок 4.1). Даний інтерфейс включає в себе роботу з програмним рендером (SDL_UpdateWindowSurface) та бібліотекую з базовим набором Tool-Kits (лістинг 4.3):

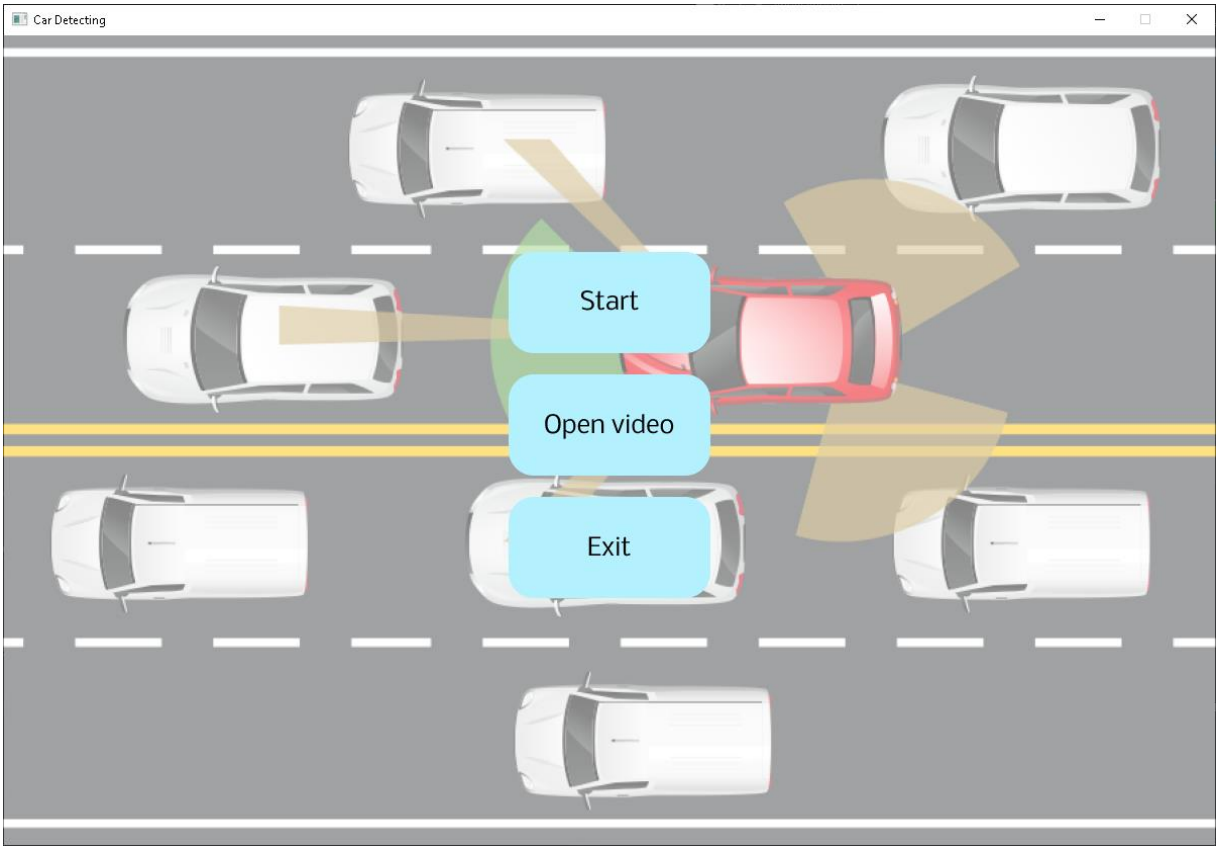

Рисунок 4.1 – Інтерфейс програмного середовища

Лістинг 4.3 – Робота Software Renderer та відображення базових примітивів:

```
static const char * path_to_img_load_menu =
"./resources/img_load_menu.jpg";
img_bg_loading_menu = IMG_Load(path_to_img_load_menu);
```

```
SDL_BlitSurface(img_bg_loading_menu, NULL, application->surface,
NULL);

CreateButton(“Start”, positon_button_start, application-
>surface, (SDL_Color) button_color, text_color);
CreateButton(“Open video”, positon_button_open_video,
application->surface, (SDL_Color) button_color, text_color);
CreateButton(“Exit”, positon_button_exit, application->surface,
(SDL_Color) button_color, text_color);
```

За допомогою функції SDL_PollEvent() ми обробляємо усі події та перевіряємо, чи користувач натиснув на кнопку.

Функція Start() запустить скрипт робити мовою Python лише тоді, коли буде обране та відкрите відео у каталозі за допомогою функції Open_Video(). Функція Open_Video() повертає строку з актуальним шляхом до нашого відео від корневої папки системи та передає цю строку, як аргумент до функції Start(). Функція Exit() робить примусову зупинку роботи всієї програми.

4.2 Запуск тестової частини програми

Після запуску функції Start(const * char path_to_video, nn number_of_NN), програма передає роботу інтерпретатору Python для подальшого запуску натренованої моделі штучної нейронної мережі (лістинг 4.4). У наступній главі буде описана детальна архітектура та робота створенної нейронної мережі.

Лістинг 4.4 – Передача аргументів до інтепретатора Python

```
python3 trained_model.py path_to_video number_of_NN
```

Запуск інтепретатора Python виконує скрипт (number_of_NN = 1) запуску відео на моделі штучної нейронної мережі з розпізнанням транспортних засобів на дорозі викори за допомогою отриманих параметрів для обробки відео каскадами Хаара (лістинг 4.5). На прикладі можна (рисунок 4.2) побачити, що ідентифікований транспортний засіб обводиться синім

квадратом. Червоним квадратом нейронна мережа попереджає про ймовірну заргрозу зіткнення. Усі дані параметри та дані виводяться у реальному часі в консоль.

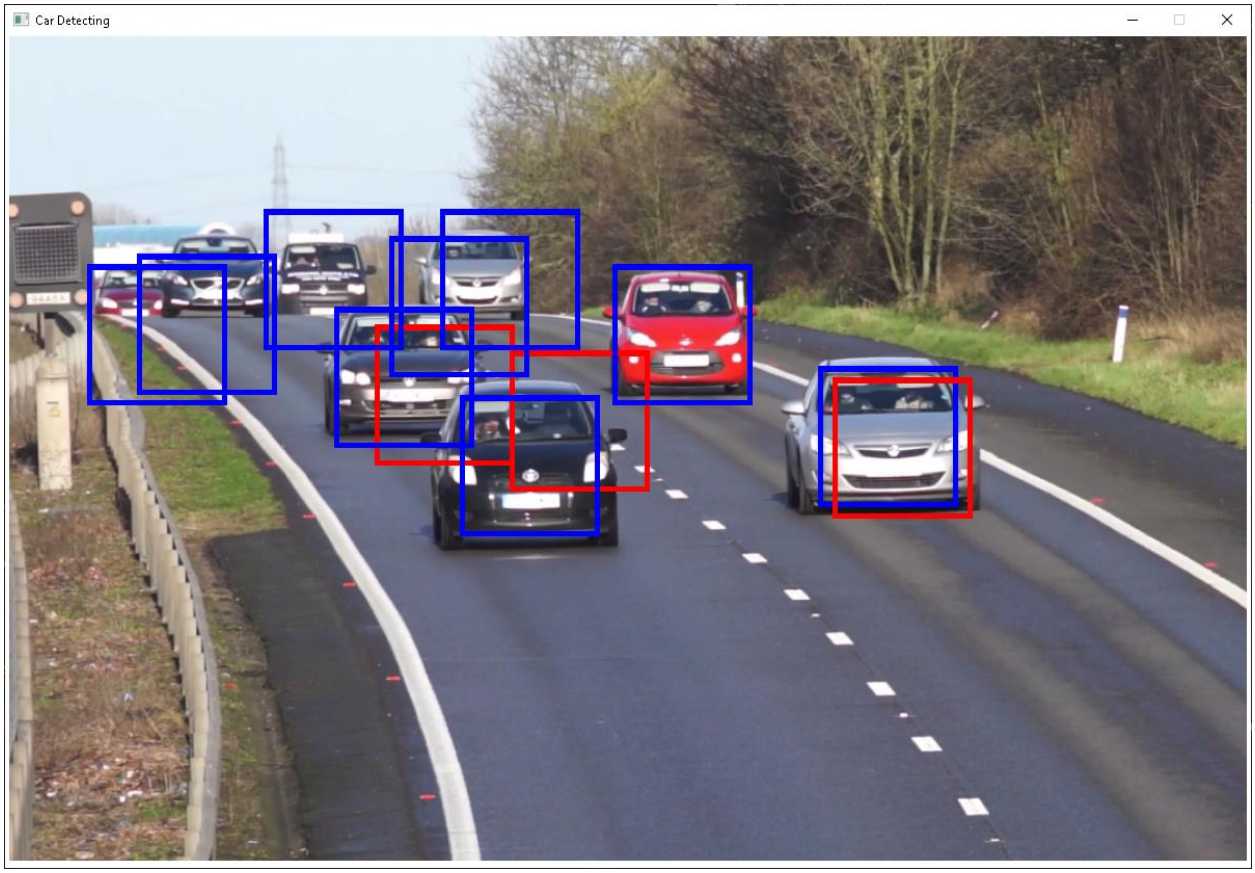

Рисунок 4.2 – Демонстрація роботи нейронної мережі з розпізнання та трекінгу транспортних засобів, .

Лістинг 4.5 – Компоненти обробки відео та розпізнання об’єктів

```
import cv2
import time
import argparse
import sys
arguments = sys.argv
car_classifier = cv2.CascadeClassifier('./car_module.xml')
car_collision_classifier                                    =
cv2.CascadeClassifier('./collision_avoid.xml')
cap = cv2.VideoCapture(arguments[1])
while cap.isOpened():
    time.sleep(.05)
    ret, frame = cap.read()
```

```
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    ident_cars = car_classifier.detectMultiScale(gray, 1.4, 2)
    collisin = car_classifier.detectMultiScale(gray, 1.6, 2)
    for (x,y,w,h) in ident_cars:
        cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 0, 255), 2)
        cv2.imshow('Cars', frame)
     for (x,y,w,h) in collisin:
        cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 0), 2)
        cv2.imshow('Cars', frame)
    if cv2.waitKey(1)== 13:
        break
cap.release()
cv2.destroyAllWindows()
```

5 РОЗРОБКА ТА ТРЕНУВАННЯ МОДЕЛЕЙ

5.1 Підготовка та валідація даних

Обрання та валідація даних є критичним етапом в розробці нейронних мереж. Важливість коректності даних полягає в тому, що неправильні або неточні дані можуть призвести до низької якості моделі, низької продуктивності та до поганих результатів.

Нами були обране готове рішення – кластерізований дата-сет для подальшого навчання нашої моделі (рисунок 5.1).

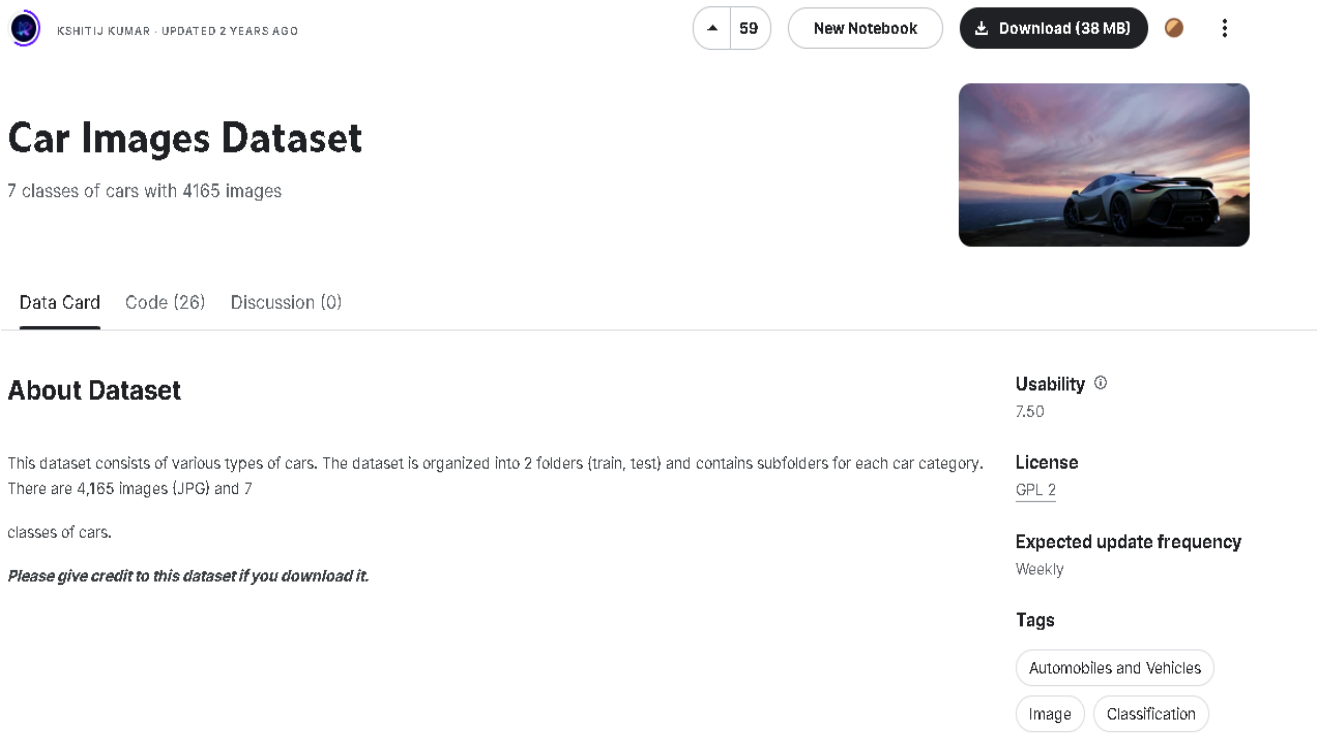

Рисунок 5.1 – Кластерізований дата-сет

Цей набір даних складається з різних класів автомобілів. Набір даних упорядковано у 2 папки (тренування, тест) і містить підпапки для кожної категорії автомобіля. Є 4165 зображень (JPG).

5.2 Архітектура штучної нейронної мережі програми

Вибір правильної архітектури нейронної мережі визначає успіх у завданнях машинного навчання та глибокого навчання. Різні архітектури підходять для різних завдань.

Згорткові нейронні мережі (CNN) виявляються дуже ефективними для завдань ідентифікації автомобілів на зображеннях (рисунок 5.2). CNN враховують локальність визначень на зображенні, що дозволяє їм ефективно розпізнавати об'єкти, які можуть мати різні розташування на зображенні. Також CNN можуть самостійно вивчати та визначати корисні признаки для розпізнавання об'єктів. Це особливо важливо для складних об'єктів, таких як автомобілі, які мають різні форми та деталі. Завдяки використанню згорток та пулінгу, CNN можуть створювати ієрархії признаків, починаючи від простих ліній та кутів і закінчуючи складними структурами.

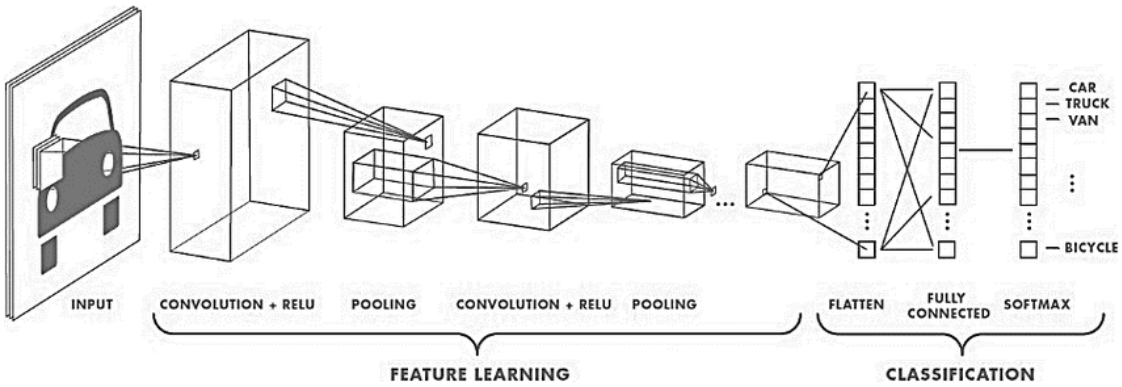

Рисунок 5.2 – Визначення транспорту згортковою нейромережею

Для тренування ми обрали 2 моделі для машинного навчання, використовуючи CNN та R-CNN – CV2.ml та Python Keras [13].

Для тренування CV2.ml (лістинг 5.1), ми заповнюємо наступні параметри:

– samples.Дата сет для тренування нейронної мережі;
– layer_sizes. Кількість прихованих шарів;
– activationFunction. Обрана функція активації нашої нейронної мережі;

– model. Кінцева модель нашої натренованої нейронної мережі.

Лістинг 5.1 – Тренування моделі CV2.ml

```
import cv2
import numpy as np
data = cv2.ml.TrainData_create(samples, cv2.ml.ROW_SAMPLE,
responses)
layer_sizes = np.int32([len(samples[0]), 32, 1])
model = cv2.ml.ANN_MLP_create()
model.setLayerSizes(layer_sizes)
model.setActivationFunction(cv2.ml.ANN_MLP_SIGMOID_SYM)
model.setTrainMethod(cv2.ml.ANN_MLP_BACKPROP)
model.train(data)
model.save("trained_model.xml")
```

Тренована модель зберігається у форматі .xml (рисунок 5.3), яка у собі містить усі метадані, інформацію про шари, параметри та значення порогів для кожного шару. Ці дані потім буде використовувати OpenCV модель для розпізнання транспортних засобів на дорозі у режимі реального часу [14].

Рисунок 5.3 – Нові параметричні дані для знаходження об’єкту

Для тренування Python Keras моделі (лістинг 5.2), ми заповнюємо наступні параметри:

– activation. Обрана функція активації нашої нейронної мережі;
– path. Обрані шляхи до наших дата-сетів;
– dense(units). Наші Dense-шари;
– metrics. Визначення метрики оцінки;
– optimizer. Оптимізатор отриманих результатів.

Лістинг 5.2 – Модель нейронної мережі Python Keras

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image
import matplotlib.pyplot as plt
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
from keras.preprocessing.image import ImageDataGenerator

model = Sequential()
model.add(Conv2D(32, (3, 3), input_shape=(64, 64, 3),
activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(units=128, activation='relu'))
model.add(Dense(units=1, activation='sigmoid'))

model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])

train_datagen = ImageDataGenerator(rescale=1./255,
shear_range=0.2, zoom_range=0.2, horizontal_flip=True)
test_datagen = ImageDataGenerator(rescale=1./255)

training_set = train_datagen.flow_from_directory(path_to_tr_set,
target_size=(64, 64), batch_size=32, class_mode='binary')
test_set = test_datagen.flow_from_directory(path_to_test_set,
target_size=(64, 64), batch_size=32, class_mode='binary')

model.fit(training_set, steps_per_epoch=len(training_set),
epochs=10, validation_data=test_set,
validation_steps=len(test_set))
```

Результат тренування R-CNN показав високу ефективність її застосування при ввиявленні та передбаченні об’єкту на малій кількості навчання (рисунок 5.4).

```
epoch: 10
step: 0, acc: 0.891, loss: 0.263 (data_loss: 0.263, reg_loss: 0.000), lr:
0.0001915341888527102
step: 100, acc: 0.883, loss: 0.257 (data_loss: 0.257, reg_loss: 0.000), lr:
0.00018793459875963167
step: 200, acc: 0.922, loss: 0.227 (data_loss: 0.227, reg_loss: 0.000), lr:
0.00018446781036709093
step: 300, acc: 0.898, loss: 0.282 (data_loss: 0.282, reg_loss: 0.000), lr:
0.00018112660749864155
step: 400, acc: 0.914, loss: 0.299 (data_loss: 0.299, reg_loss: 0.000), lr:
0.00017790428749332856
step: 468, acc: 0.917, loss: 0.192 (data_loss: 0.192, reg_loss: 0.000), lr:
0.00017577781683951485
training, acc: 0.894, loss: 0.291 (data_loss: 0.291, reg_loss: 0.000), lr:
0.00017577781683951485
validation, acc: 0.874, loss: 0.354
```

Рисунок 5.4 – Точність передбачення нейронною мережею

6 ОПИС ДОСЛІДЖЕННЯ

6.1 Параметри та дата-сет дослідження

Інтелектуальні інтегровані системи використовують алгоритми машинного навчання для розпізнавання оточуючого середовища, прийняття рішень та виконання відповідних дій. Для досягнення цих об'єктивів використовують різноманітні методи машинного навчання, такі як глибоке навчання та навчання з підкріпленням, комплексні математичні моделі.

Data-set має ввідноситись до того завдання, де штучні нейронні мережі повинні виконувати їх з найефективнішим показником. Дата-сет – це набір різних даних, які зустрічаються в повсякденному житті, щоб модель могла вчитися на різних вхідних даних. Однак деякі моделі можуть бути недостатньо навчені, перенавчені через набір даних.

Дата-сет нашого дослідження включає в себе разноманітні дані, на яких ми будем досліджувати поведінку нейронних мереж (рисунок 6.1).

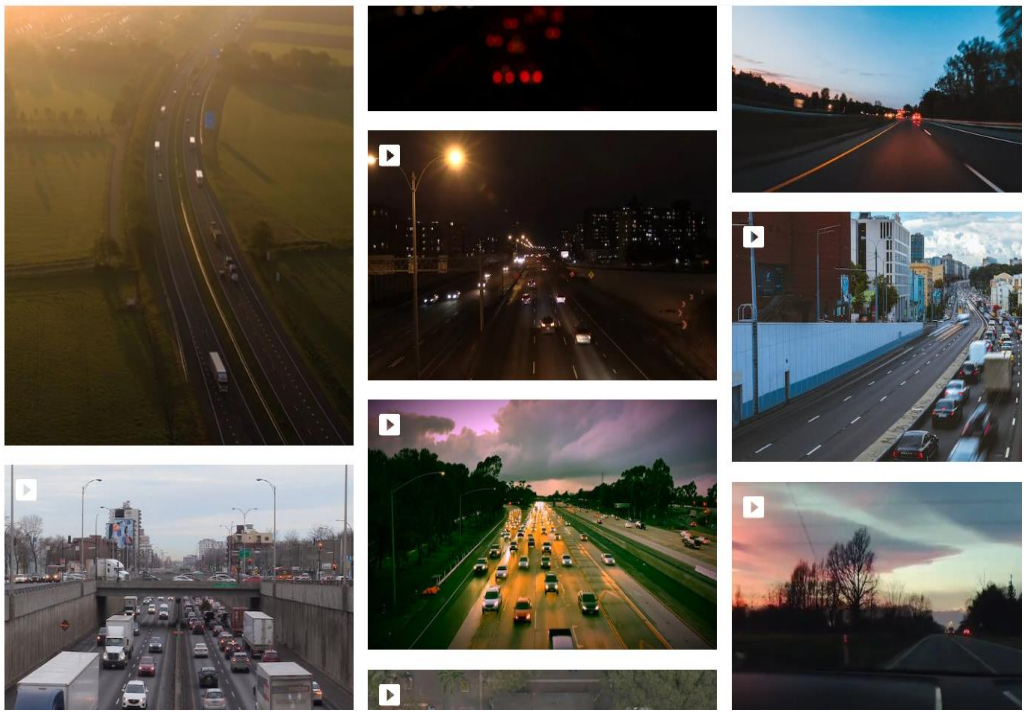

Рисунок 6.1 – Дата-сет нашого дослідження

6.2 Порядок та результати дослідження

Нами було обрано 20 відеорядів, які були максимально наближені до симуляції реальної їзди на транспортному засобі. Дата-сет є актуальним, наближений до реального світу та повністю відповідає базовій постановці завдання на дослідження.

Цей дата-сет ми розбили на дві, High-quality та Low-quality групи. Перші 10 відеорядів попали до групи High-quality, так як вони відповідають наступним параметрам: чітке зображення, висока якість картинки, велика кількість кадрів. До Low-quality групи попали ті відеоряди, на яких картинка була нечітка, якість зображення була меньше за 360p та мали frame-rate у 30 кадрів.

Наступним кроком нам потрібно було визначати ефективність роботи з розпізнанням образів транспортних засобів та їх трекінгу на різних алгоритмах машинного навчання на нашому дата-сеті. Для цього ми обрали два готових комерційних рішення з відкритим кодом, та дві наші навченні моделі. Почерзі ми запустили кожну модель: cv2.ml, Python Keras, Yolo v3 та MobileNetV2 [15]. Отримані результати ми розділили на дві таблиці за точністю (рисунок 6.2) у розпізнанні транспортних засобів для кожної моделі (таблиці 6.1, 6.2).

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Рисунок 6.2 – Формула визначення точності роботи моделі

Таблиця 6.1 – Результати дослідження даних на тестах, група High-quality

Назва моделі	Точність моделі, %
Convolutional NN(cv2.ml)	64,7
R-CNN (Keras)	78,3
Yolov3	93,1
MobileNetV2	87,4

За результатами отриманих даних, можна визначити, що усі чотири моделі мають гарні результати у визначенні та трекінгу транспортних засобів на дорозі з часом. Найкраще з визначенням та трекінгом транспортних засобів на дорозі впоралася Yolov3 (R-CNN архітектура), що базується на швидкому визначенні об'єктів на розділеній площині.

Таблиця 6.2 – Результати дослідження даних на тестах, група Low-quality

Назва моделі	Точність моделі, %
Convolutional NN(cv2.ml)	30
R-CNN (Keras)	70,7
Yolov3	90,3
MobileNetV2	85,1

За результатами дослідження даних на тестах групи Low-quality, можна зробити висновок, що усі моделі, окрім Convolutional NN(cv2.ml), непогано впоралися з визначенням та трекінгу транспортних засобів на дорозі з часом. Низький результат Convolutional NN(cv2.ml) можна пояснити застарілістю моделі, яка базується на визначенні Каскадів Хаара і неможливістю якісної адаптації моделі до визначення об’єктів при змінних параметрах.

Було запропоновано оптимізувати архітектури нейронних мереж для більше ефективнішого розпізнання та трекінгу транспортних засобів що у свою чергу теоретично повинно показати кращі показники роботи інтелектуальної системи впродовж всього часу. Оптимізацію проводили у два етапи: доробивши архітектуру нейронної мережі (Python Keras), ми додали кратну кількість згорткових шарів, а також взяли нові провалідовані дата-сети для навчання з більшим quality-index.

За результатами дослідження даних на оптимізованих моделях, можна побачити, що точність NN(cv2.ml) зросла на 6% у High-quality групи та на 8% у Low-quality групи. Так незначний зріст можна пояснити архітектурним рішенням данної моделі. Точність NN у Python Keras зросла на 10% для High-

quality групи та на 7 % для Low-quality групи. Ріст точності цієї моделі пов’язан з її вдалою архітектурую, яка базуються на передбаченні об’єктів.

Дослідження, проведене на визначення можливості інтелектуальної системи щодо попередження аварійних ситуацій на заданих моделях мереж показало низьку ефективність та великий відсоток помилок (рисунок 6.3).

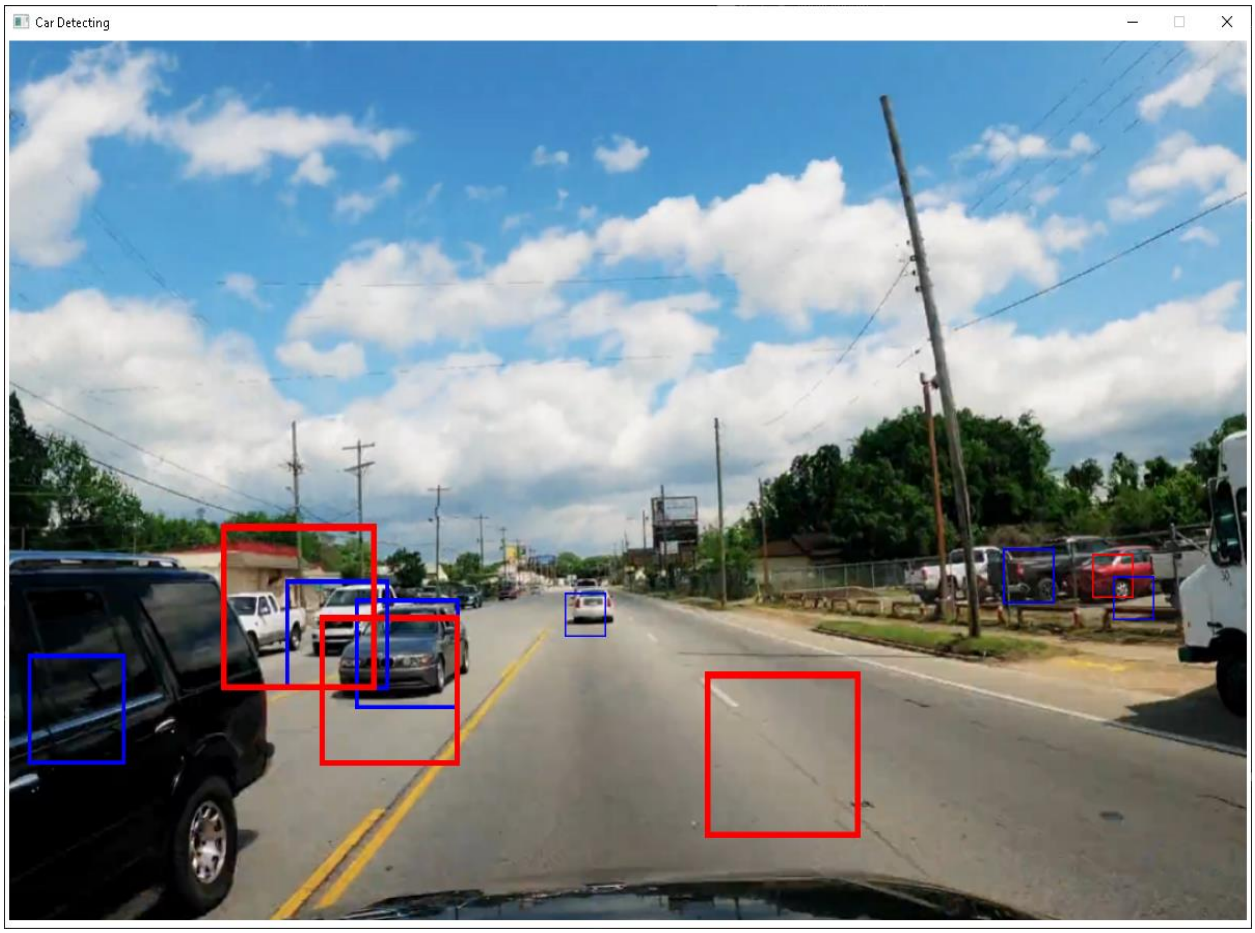

Рисунок 6.2 – Попередження аварійних ситуацій

Отримані результати вказують на те, що cv2.ml (каскади Хаара) не досягла необхідної точності та ефективності в класифікації потенційних транспортних аварійних ситуацій на дорозі. Зважаючи на обмежені можливості cv2.ml, результати тестування системи показали низьку ефективність у попередженні аварійних ситуацій. Однак, ця система, навіть з недоліками, може виступати як міцний фундамент для майбутніх досліджень

та вдосконалень в галузі розпізнавання та попередження аварійних транспортнихситуацій на дорозі.

6.3 Підсумки дослідження

Результати цього дослідження мають наслідки для розвитку та вдосконалення систем розпізнавання транспортних засобів та попередження аварійних ситуацій на дорозі. Наведені нижче висновки та практична цінність дослідження можуть служити основою для подальших заходів та вдосконалень в цій галузі:

– визначення найбільш ефективної архітектури. Результати вказують, що Yolov3 (R-CNN архітектура) найкраще впоралась з визначенням та трекінгом транспортних засобів на дорозі. Це слугує як важлива вказівка для вибору архітектури нейронної мережі для подібних завдань;

– оптимізація архітектури для вдосконалення результатів. Зміни в архітектурі нейронної мережі, такі як збільшення кількості прихованих шарів, показали певний успіх в покращенні ефективності розпізнавання об’єктів та їх трекінгу. Це вказує на важливість оптимізації архітектури для досягнення більш високих результатів;

– важливість якості вихідних даних. Розділення дата-сету на групи High-quality та Low-quality дозволило визначити вплив якості вихідних даних на результати. Це підкреслює необхідність використання високоякісних та репрезентативних даних для навчання та тестування моделей;

– переваги Yolov3 в реальних умовах. Висока швидкість та точність Yolov3 робить її перспективною для використання в реальних умовах. Це може мати практичне значення для систем безпеки на дорозі та розробки автономних транспортних засобів;

– потенціал для подальших оптимізацій. Запропоновані шляхи

оптимізації архітектур нейронних мереж та використання нових, провалідованих дата-сетів вказують на потенціал для подальших вдосконалень та покращень в розпізнаванні та попередженні аварій.

Практична цінність даного дослідження полягає в покращенні систем безпеки на дорозі, що може призвести до зменшення аварій та покращення загальної безпеки у дорожньому русі. Розроблені методи та підходи можуть служити основою для реалізації проривних інтелектуальних систем в автотранспорті з метою вдосконалення їхньої ефективності та надійності.

6.4 Модель Perceptual Decision Making

Perceptual Decision Making в поєднанні з трансформерними нейронними мережами надає інноваційний метод створення інтелектуальних систем для запобігання аварій на дорогах. Цей підхід поєднує передові технології у галузі сприйняття та обробки інформації з унікальною потужністю архітектури трансформатора.

Нейронні мережі, засновані на обробці зображень і сенсорних даних, представляють значний прогрес в області відновлення і класифікації навколишнього середовища. Однак їх функціональність, як правило, обмежує здатність до вирішення конкретних завдань, представлених на тренованих даних. Ці мережі не володіють розумінням і здатністю створювати абстракції або узагальнення, що виходять за рамки свого прямого призначення.

Система повинна не просто надавати ізольовані аналітичні висновки, але мати здатність формувати глобальне розуміння оточуючого середовища на дорозі. Це вимагає, щоб прийняття рішень системою базувалось не лише на даних з окремих сенсорів, а й на складному, високорівневому аналізі, який враховує взаємозв'язки та контекст ситуації в цілому.

Для досягнення цієї мети трансформерні нейронні мережі використовують свою здатність обробки послідовностей даних та виявлення довгострокових залежностей для інтеграції інформації з різних джерел. Це

забезпечує не лише точне сприйняття деталей, але й здатність до узагальнення, що дозволяє системі розробляти глобальні стратегії та адаптуватися до різноманітних сценаріїв на дорозі.

Важливим аспектом цієї концепції є використання навчання з підсиленням для постійного вдосконалення системи. Алгоритми навчання з підсиленням не тільки забезпечують адаптацію до змінюючихся умов, але і сприяють формуванню глобального розуміння шляхом інтеграції нових даних та досвіду. Такий підхід робить систему більш гнучкою, дозволяючи їй ефективно справлятися з різними сценаріями і підвищувати рівень безпеки на дорогах [16].

Результатом впровадження Perceptual Decision Making з використанням трансформерних нейронних мереж, збагачених глобальним розумінням ситуації є перспективним напрямком у розробці інтелектуальних систем для автомобільної безпеки. Це поєднання передових технологій та наукових методів призводить до створення високоефективних систем, здатних не тільки сприймати оточуюче середовище, але і глобально розуміти складні дорожні сценарії.

ВИСНОВКИ

У кваліфікаційній роботі були досліджені сучасні інтелектуальні системи попередження аварійних транспортних ситуацій на дорозі. Розглянута важливість, доцільність та актуальність розробки систем попередження аварійних ситуацій з метою підвищення загальної безпеки на дорогах та зниженню частоти та тяжкості аварій.

Була поставлена задача на дослідження ефективності моделей машинного навчання щодо виявлення транспортних засобів та попередження аварійних ситуацій на дорозі з подальшим їх застосуванням та оптимізацією.

Була досліджена концепція штучного інтелекту, теорія, парадигми та основні методи машинного навчання. Розглянуті основні види штучних нейронних мереж (ANN), математичні моделі NN, а також їх програмне та апаратні рішення.

Проаналізовані та порівняні такі архітектури нейронних мереж, як Transformer, CNN, RNN модель та R-CNN алгоритм, Faster R-CNN модель, GAN модель, HOG та YOLO алгоритми. Наведені практичні реалізації певних моделей нейронних мереж.

Аналіз показав, що різниця між згортковими нейронними мережами та регіонально-згортковими нейронними мережами є незначною. RNN модель є більш гнучкою де зв’язки між елементами утворюють спрямований цикл, що дозволяє демонструвати динамічну поведінку у часі. Transformer показав гнучкість моделі при обробці послідовних даних, у той час, як YOLO показав ефективність моделі для реального часу виявлення об'єктів на великих зображеннях та відео, які добре впорядкована з точки зору обчислювальної ефективності.

Було розроблене тестове середовище для реалізації моделі машинного навчання з розпізнанням транспортних засобів. На базі цієї програми було

проведене досліження щодо оптимізації архітектури моделей шляхом збільшенням прихованих шарів та вдосконаленням вхідних даних.

Результати дослідження показали, що системи, яка мають певні архітектурні недоліки, можуть виступати у ролі базового фундаменту для майбутніх досліджень та вдосконалень в галузі розпізнавання аварійних ситуацій на дорозі. Напрямки їх подальшого вдосконалення можуть включати: оптимізація архітектури моделі машинного навчання, збільшення обсягу та якості навчального набору, автоматизоване навчання, регуляризація та уникнення переобучення.

За підсумками дослідження було запропоновано інтеграцію нової розширеної системи навчання із підкріпленням (Reinforcement Learning) у розробляєму інтелектуальну систему попередження аварійних транспортних ситуацій на дорозі. Був запрононован концепт інтегрованої системи управління транспортним засобом, який базується на соціальному аналізі взаємодії учасників руху – Perceptual Decision Making Neural Network Architecture.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Schalkoff R. J. Intelligent Systems : Principles, Paradigms, and Pragmatics. R. J. Schalkoff. – Virginia : Jones & Bartlett Learning, LLC, 2011.

2. Karim R. L. Competing in the Age of AI. Strategy and Leadership When Algorithms and Networks Run the World. R. L. Karim, I. Marco, – Boston: Harvard Business Review Press, 2020.

3. Anderson K. Designing Autonomous AI. A Guide for Machine Teaching / 1st Edition. K. Anderson – Virginia: O'Reilly Media, 2022.

4. Курченко О. В. Інтелектуальна система попередження аварійних транспортних ситуацій на основі нейронної мережі. Матеріали 27-го Міжнар. Молодіж. форуму, м. Харків, 10 травня 2023 р. – Харків, 2023.

5. Mitchel, M. Artificial Intelligence: A Guide for Thinking Humans. M. Mitchel. – New York: Farrar, Straus and Giroux, 2019.

6. Russell S. J. Artificial Intelligence. S. J. Russell, P. Norvig. – New York : Pearson, 2020.

7. Bishop C. M. Pattern Recognition and Machine Learning. C. M. Bishop. – Berkeley : Springer, 2016.

8. Celikyilmaz A. Modeling Uncertainty with Fuzzy Logic. A. Celikyilmaz. – Berlin, Heidelberg : Springer Berlin Heidelberg, 2009.

9. Bostrom N. Superintelligence: Paths, Dangers, Strategies / N. Bostrom. – Oxford: Oxford University Press, 2014.

10. Kaicker S. The Complete ANSI C. S. Kaicker. – NY : BPB Publications, 2003

11. Mitchel S. SDL Game Development. S. Mitchel. – Birmingham : Packt, 2013.

12. Bradski G. Learning OpenCV . G. Bradski, A. Kaehler. – Virginia : O'Reilly Media, Incorporated, 2008.

13. OpenCV [Електронний ресурс]. – https://docs.opencv.org/4.x/da/d9d/tutorial_dnn_yolo.html.

14. OpenCV Haar Cascade [Електронний ресурс]. – Режим доступу: https://pyimagesearch.com/2021/04/12/opencv-haar-cascades/.

15. Mobilenetv2 [Електронний ресурс]. – Режим доступу: https://www.mathworks.com/help/deeplearning/ref/mobilenetv2.html.

16. Курченко О. В., Рожнова Т. Г. Модель Perceptual Decision Making. IV Міжнародна науково-практична конференція «Contemporary challenges of society and ways to overcome them», м. Таллінн, Естонія, 2024.