Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та мехатроніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

Другий (магістерський)
(рівень вищої освіти)

Розроблення автоматизованого метода розпізнавання транспортних технологічних об'єктів за допомогою нейронної мережі для робототехнічної платформи
(тема)

Виконав:
студент 2 курсу, групи КІТПВм-20-1
Левченко Євген Олександрович
(прізвище, ініціали)

Спеціальності 151 Автоматизація та комп'ютерно інтегровані технології
(код і повна назва спеціальності)

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

Освітня програма Комп'ютерно-інтегровані Технологічні процеси і виробництва
(повна назва освітньої програми)

Керівник Филипенко І. О.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри КІТАМ _____ Невлюдов І. Ш.
(підпис) (прізвище, ініціали)

2021р.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ
Факультет АКТ
Кафедра КІТАМ
Рівень вищої освіти другий (магістерський)
Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
Тип програми Освітньо-професійна
Освітня програма КІТПВМ, Комп'ютерно-інтегровані технологічні процеси і виробництва
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. Кафедри КІТАМ_____
(підпис)
«___»_____2021р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Левченку Євгену Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення автоматизованого метода розпізнавання транспортних технологічних об'єктів за допомогою нейронної мережі для робототехнічної платформи
Затверджена наказом по університету від 08.11.2021 № 1697Ст
2. Термін подання студентом роботи до екзаменаційної комісії 06.12.2021
3.Вихідні дані до роботи
4.Перелік питань,що потрібно опрацювати в роботі
4.1 Вступ;
4.2 Аналіз сучасного стану проблеми розпізнавання технологічних об'єктів за допомогою нейронної мережі;
4.3 Обгрунтування обраної нейронної мережі та вибір інструментів для програмної реалізації;
4.4 Розробка програмного забезпечення;
4.5 Висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій Демонстраційний матеріал, представлений у форматі презентації PowerPoint (*.ppt). – 12 с. Формату А4.

6. Консультанти розділів роботи

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

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Приміт-ка
1	Аналіз технічного завдання	08.11.21	виконано
2	Аналіз сучасного стану проблеми	10.11.21	виконано
	розпізнавання технологічних		
	об'єктів за допомогою нейронної		
	мережі		
3	Розробка структурної схеми модулю	13.11.21	виконано
	розпізнавання транспортних		
	технологічних об'єктів для		
	роботизованої платформи		
4	Розробка методу розпізнавання	15.11.21	виконано
	технологічних транспортних об'єктів		
5	Розробка алгоритмічно-програмного	19.11.21	виконано
	забезпечення автоматизованої		
	системи розпізнавання транспортних		
	технологічних об'єктів		
6	Експериментальні дослідження	21.11.21	виконано
	автоматизованого методу розпізнавання		
	транспортних технологічних об'єктів		
7	Оформлення пояснювальної записки	26.11.21	виконано
8	Подання атестаційної роботи в ЕК		виконано

Дата видачі завдання 08.11.2021

Студент _____ Левченко Є.О.
(підпис) (прізвище, ініціали)

Керівник роботи _____ проф. каф. КІТАМ Филипенко І.О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 92 с., 4 табл., 38 рис., 1 дод., 64 джерел.

НЕЙРОННА МЕРЕЖА, КОМП’ЮТЕРНИЙ ЗІР, ШТУЧНИЙ ІНТЕЛЕКТ, ВИРОБНИЦТВО.

Об’єкт дослідження – роботизоване виробництво.

Предмет досліджень – методи розпізнавання транспортних технологічних об'єктів.

Мета кваліфікаційної роботи – підвищення ефективності виробництва шляхом впровадженням розробленого автоматизованого метода розпізнавання транспортних технологічних об’єктів за допомогою нейронної мережі.

Методи дослідження – теоретичний аналіз архітектурних рішень нейронних мереж, структурного складу мереж, переваг та недоліків структур штучних нейронних мереж та розробка автоматизованого методу розпізнавання із застосуванням ПЗ бібліотек.

Практична цінність полягає у подальшому використанні розроблених структур нейронних мереж для задач з розпізнавання та класифікації транспортних технологічних об’єктів.

Упровадження розробленого програмного засобу можливе в комп’ютерно-інтегрованих автоматизованих системах на виробництві або у процесі виготовлення транспортних технологічних об’єктів, так і в начальних лабораторіях у освітньому процесі на лабораторних практикумах та практичних роботах в межах спеціальності 151 Автоматизація та комп’ютерно-інтегровані технології та можуть бути цікавим для фахівців, що пов’язані з промисловою автоматизацією.

ABSTRACT

Explanatory note: 92 pages, 4 tables, 38 figures, 1 appendix, 64 sources.

NEURAL NETWORK, COMPUTER VISION, ARTIFICIAL INTELLIGENCE, MANUFACTURING.

The object of research is robotic production.

The subject of research – methods of recognition of transport technological objects.

The purpose of the qualification work is to increase the efficiency of production by implementing the developed automated method of recognition of transport technological objects using a neural network.

Research methods – theoretical analysis of architectural solutions of neural networks, structural composition of networks, advantages and disadvantages of structures of artificial neural networks and development of an automated method of recognition using library software.

The practical value lies in the further use of the developed neural network structures for the tasks of recognition and classification of transport technological objects.

Implementation of the developed software is possible in computer-integrated automated systems in production or in the process of manufacturing transport technological objects, and in primary laboratories in the educational process in laboratory workshops and practical work within the specialty 151 Automation and computer-integrated technology and may be of interest to professionals involved in industrial automation.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ГНГ – гістограма направлених градієнтів;

ГС – градієнтний спуск;

ДСТУ – Держстандарт України;

КЗ – комп'ютерний зір;

МН – машинне навчання;

НМ – нейронна мережа;

ООП – об’єктно-орієнтоване програмування;

ПЗ – програмне забезпечення;

ШІ – штучний інтелект;

GUI – graphical user interface;

ВСТУП

Практичне застосування машинного навчання стимулює бізнес-результати, які можуть суттєво вплинути на прибуток компанії. Нові технології в цій галузі швидко розвиваються і розширюють застосування машинного навчання майже безмежними можливостями. Галузі, які залежать від величезної кількості даних і потребують системи для їх ефективного та точного аналізу, прийняли машинне навчання як найкращий спосіб побудови моделей, стратегії та планування.

На сьогоднішній день багато галузей розробляють більш надійні моделі, здатні аналізувати більші та складніші дані, забезпечуючи швидші та точніші результати у великих масштабах. Інструменти машинного навчання дозволяють організаціям швидше визначати прибуткові можливості та потенційні ризики.

Об’єкт дослідження – роботизоване виробництво.

Предмет досліджень – методи розпізнавання транспортних технологічних об'єктів.

Актуальність роботи полягає в дослідженні сучасних методів проектування нейронних мереж для задач з розпізнавання об’єктів, з’ясовуванні підходів до побудови нейронних мереж та розробці власного рішення для розпізнавання та класифікації об’єктів.

Методи дослідження – теоретичний аналіз архітектурних рішень нейронних мереж, структурного складу мереж, переваг та недоліків структур штучних нейронних мереж та розробка автоматизованого методу розпізнавання із застосуванням програмного забезпечення бібліотек.

Мета кваліфікаційної роботи – підвищення ефективності виробництва шляхом впровадженням розробленого автоматизованого метода розпізнавання транспортних технологічних об’єктів за допомогою нейронної мережі.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– провести аналіз сучасного стану проблеми розпізнавання технологічних об’єктів за допомогою нейронної мережі;

– провести розробку структурної схеми системи розпізнавання технологічних об’єктів;

– провести розробку структурної схеми та об'єктної моделі програмного засобу;

– провести обґрунтування обраної нейронної мережі та вибір інструментів для програмної реалізації;

– провести експериментальні дослідження на розробленому програмному засобі.

Атестаційна робота виконана згідно ДСТУ 3008 – 15 [1], та керуючись навчальним посібником з дипломного проекту [2] та методичними вказівками [3]. Під час навчання на другому (магістерському) рівні матеріали, що були використані для написання атестаційної роботи було апробовано на міжнародних конференціях [4 – 5] та результати висвітлено в наукових статтях [6 – 11].

1 АНАЛІЗ СУЧАСНОГО СТАНУ ПРОБЛЕМИ РОЗПІЗНАВАННЯ ТЕХНОЛОГІЧНИХ ОБ'ЄКТІВ ЗА ДОПОМОГОЮ НЕЙРОННОЇ МЕРЕЖІ

1.2 Аналіз завдання розпізнавання транспортних технологічних об'єктів

Сучасне виробництво зосереджено на покращенні показників промисловості за рахунок інноваційного застосування технологій, процесів та методів для проектування, та виробництва продукції. Фундаментальними напрямками кожного виробництва на сьогоднішній час є покращення базових показників: надійність, якість, темп виготовлення кінцевого виробу. Завдяки введенню систем розпізнавання в виробничий процес дані показники можна покращити. В результаті цього виробництво зможе виробляти більш конкурентні вироби та буде мати менші витрати в процесі виробництва.

Прослідковуючи проблеми, які є на виробництві, можна сказати, що однією з них є проблема розпізнавання технологічних об'єктів. Розпізнавання об'єктів дає можливість вирішувати широкий спектр проблем на виробництві. Тобто, завдяки використанню системи виявлення об'єктів, можна класифікувати типи технологічних об'єктів, а також знаходити їх на зображенні, підраховувати їх, а потім відстежувати ці об'єкти в реальному часі. Використовуючи розпізнавання об'єктів на базі машинного навчання, можна вибрати певні параметри та відповідну статистику кількості зображених об'єктів. Це не тільки робить технологічні процеси більш гнучкими, а й знижує кількість відхилень під час категоризації.

У виробництві технологічних об'єктів всі технологічні процеси

повністю автоматизовані. Використання робототехніки в цій галузі надзвичайно корисне. Впровадження технології комп’ютерного зору для правильного визначення розташування та диференціації об’єктів відповідно до їх переміщення відкриє можливості більш ефективного виробництва та підвищення продуктивності. Виявлення об'єктивів на основі машинного навчання дозволяє реалізувати цю можливість.

Виявлення об’єктів – це такий процес , який дозволяє ідентифікувати та знаходити об’єкти на зображенні чи відео. Завдяки такому виду ідентифікації та локалізації виявлення об’єктів його можна використовувати для підрахунку об’єктів на виробничій лінії або автоматизованій ділянці для відстеження їх точного розташування, при цьому точно позначаючи їх. Виявлення об’єктів дозволяє відразу класифікувати типи технологічних об’єктів, а також знаходити їх екземпляри на зображенні.

Взагалі, виявлення об’єктів можна розділити на два підходи, а саме використання технології виявлення об’єктів на основі машинного навчання або на основі глибокого навчання.

Найбільш ефективним є метод заснований на машинному навчанні, який використовує методи комп’ютерного зору для перегляду різних ознак зображення, таких як кольорова гістограма або краї, для визначення груп пікселів, до яких можуть належати об’єкти. Ці ознаки потім подаються в регресійну модель, яка передбачає розташування об’єкта разом із його міткою.

Виявлення об’єктів нерозривно пов’язане з методами комп’ютерного зору, такими як розпізнавання зображень і сегментація зображень, оскільки воно дає змогу зрозуміти та аналізувати об’єкт на зображеннях або відео. Але є важливі відмінності. Розпізнавання зображень виводить лише мітку класу для ідентифікованого об’єкта, а сегментація зображення створює розуміння елементів на рівні пікселів. Унікальна здатність знаходити об’єкти на зображенні або відео відрізняє виявлення об’єктів від інших завдань. Це дозволяє підраховувати, а потім відстежувати ці об’єкти.

Варіантом вирішення представлених вище проблем є впровадження систем розпізнавання технологічних об’єктів нарізних етапах виробництва.

Системи розпізнавання можуть бути встановлені стаціонарно, в тих місцях, де технологічні операції потребують постійного суворого контролю, або є вірогідність отримання браку та постійного контролю. Або ж, такі системи можуть бути – мобільними і встановленими, наприклад, на робототехнічній платформи.

Встановлення камери на робототехнічну платформу, може бути статичним або мобільним.

Пересуваючись в приміщеннях або на "зовні" роботехнічна платформа з камерою може слідкувати, розпізнавати та підраховувати технологічні об’єкти на певній ділянці, за заданим маршрутом, або в указаних, навіть, точках виробництва, або може бути використана для "швидкого реагування", тобто отримання певної інформації і керуватися оператором з пульту у "живому режимі".

Отримавши зображення з певної ділянки виробничого процесу , використовуючи можливості комп’ютерного зору дають змогу обробляти вхідний потік кадрів, де кожний кадр буде представляти собою матрицю кольорів, так як кожен піксель отримає цифрове значення. В результаті це дасть змогу обробити матрицю, яка буде представлена в виді двомірного масиву. Кожний кадр буде представляти масив цілих чисел значень кольорів пікселів довжиною. Довжина масиву буде співпадати з дозволом вхідного кадру, який буде помножений на кількість основних кольорів в адитивній моделі кольорів значень.

Адитивна модель описує спосіб кодування кольору для відтворення кольорів за допомогою трьох кольорів, які прийнято називати основними. Внаслідок чого, вхідний масив цілих чисел значень кольорів пікселів буде набором вхідних данних для навчання нейронної мережі. Мережа буде навчатись класифікувати об’єкти не тільки за кольором, але і за формою за допомогою згорткових шарів, які використовують класифікатори

особливостей об’єкта, обробляючи карту особливостей кожного нового вхідного об’єкта, яка представляє собою під-масив, який входить до двомірного масиву, який представляє з себе один кадр.

Можна зробити висновок, що мережа, яка має згорткові шари та багатошаровий персиптрон у свої моделі буде найефективнішим вибором так як ця проблема нелінійна. В результаті чого в якості функції має бути використана формула нелінійної функції активації - сигмоїда. Вона дозволить прискорити навчання мережі, бо вона має діапазон від 0 до 1, бо вхідний масив не може мати від'ємних значень. Це дозволить активувати більшу кількість штучних нейронів. У якості функції похибки, будемо використовувати середню квадратичну помилку.

1.2 Аналіз принципів функціонування нейронних мереж

Дослідження, розробка, реалізація методів розпізнавання об’єктів є провідним напрямом у розвитку сучасного програмного забезпечення. Розпізнавання образів в останні роки знаходить все більше застосування завдяки нейронним мережам.

НМ є високоякісною сучасною технологією, в основі якої лежить робота людського мозку.

Нейронні мережі з’явилися в теорії управління одним з перших прикладів переходу від управління найпростішими лінійними стаціонарними системами до управління складними нелінійними, нестаціонарними, багатовимірними, багато зв’язними системами.

Переваги простої нейронної мережі в тому, що вона досить проста в обслуговуванні та їх краще використовувати для обробки даних, які містять шум.

НМ є універсальним інструментом для обробки даних.

Труднощі, з якими стикаються системи, що спираються на жорстко зафіксовані знання, підказують, що системам штучного інтелекту потрібна

здатність здобувати власні знання, витягуючи шаблони з необроблених даних. Ця здатність відома як МН. Введення машинного навчання дозволило комп’ютерам вирішувати проблеми, пов'язані з пізнанням реального світу та приймати рішення, які здаються суб'єктивними.

Багато завдань з навчання штучного інтелекту можна вирішити, розробивши правильні налаштування для отримання цього завдання, тобто надавши функції простим алгоритмом машинного навчання. Наприклад, корисною ознакою для ідентифікації оратора за допомогою звукового сигналу є оцінка голосового тракту мовця. Ця особливість дає чітку підказку щодо того, чи є оратор чоловіком, жінкою чи дитиною. Однак для багатьох завдань важко дізнатись, з якими саме особливостями слід працювати. Наприклад, потрібно написати програму для виявлення автомобілів на фотографіях. Відомо, що у автомобілів є колеса, тому з’явиться бажання використовувати присутність колеса як особливість. На жаль, важко описати, як саме виглядає колесо з точки зору значень пікселів. Колесо має просту геометричну форму, але його зображення може бути ускладнене тінями, що падають на колесо, сонячним світлом, металевими частинами колеса, крилом автомобіля чи предметом на передньому плані, затемнюючи частину колеса тощо.

Одним з варіантів вирішення цієї проблеми є використання машинного навчання для виявлення не тільки відображення від подання до виводу, але й самого представлення. Цей підхід відомий як навчання ознакам, або навчання представлень.

Вивчене представлення часто призводить до набагато кращої продуктивності, ніж це можна отримати за допомогою вручну розроблених ознак. Також це дає можливість системам штучного інтелекту швидко адаптуватися до нових завдань, з мінімальним втручанням людини.

Розробляючи функції або алгоритми для вивчення функцій, метою є, як правило, відокремлення факторів варіації, що пояснюють спостережувані дані. У цьому контексті використовується слово «фактори» для позначення

окремих джерел впливу. Фактори, як правило, не поєднуються множенням. Такі фактори часто не є ознакою, яка безпосередньо спостерігається. Натомість вони можуть існувати як неспостережні сили, які впливають на спостережувані величини. Вони також можуть існувати як конструкції у свідомості людини, які надають корисні спрощення пояснень або випливають із причин спостережуваних даних. Їх можна вважати поняттями або абстракціями, які допомагають зрозуміти мінливість даних. Під час аналізу мовного запису фактори зміни включають в себе мовлення, стать, акцент та слова, які говорять. Аналізуючи анімацію автомобіля, фактори коливання включають положення машини, її колір, кут та яскравість сонця. Основна проблема у багатьох реальних системах штучного інтелекту – це багато факторів змін, які впливають на кожну частину даних, яку можна спостерігати. Окремі пікселі на зображенні червоного автомобіля можуть бути дуже близькими до чорного кольору вночі. Форма силуету автомобіля залежить від кута огляду. Більшість застосувань вимагають роз'єднання чинників варіації та відкидання таких, які є не такими важливими. Звичайно, витягнути такі високі, абстрактні риси із початкових даних дуже важко.

Багато з цих факторів варіації, такі як акцент мовця, можна визначити лише за допомогою складного, майже людського розуміння даних. Алгоритми глибокого навчання можна розглядати як складну та математично складну еволюцію алгоритмів машинного навчання. Останнім часом ця галузь привертає багато уваги, і недарма: останні події привели до результатів, які раніше не вважалося можливими. Воно описує алгоритми, які аналізують дані з логічною структурою, подібною до того, як людина робить висновки. Зауважте, що це може статися як шляхом навчання під наглядом, так і без нагляду. Щоб досягти цього, програми глибокого навчання використовують багатошарову структуру алгоритмів, яку називають штучною нейронною мережею. Дизайн такої штучнії нейроннії мережі натхненний біологічною нейронною мережею людського мозку, що веде до процесу навчання, який набагато ефективніший, ніж стандартні

моделі машинного навчання [12].

Глибоке навчання вирішує дану центральну проблему в навчанні виявленню ознак шляхом введення уявлень, які виражаються в інших, більш простих уявленнях. Сьогодні глибоке навчання використовується в багатьох сферах. У автоматизованому водінні, наприклад, глибоке навчання використовується для виявлення об’єктів, таких як знаки СТОП або пішоходів. Військові використовують глибоке навчання для ідентифікації об’єктів із супутників, напр. виявляти безпечні чи небезпечні зони для своїх військ. Звичайно, індустрія побутової електроніки також сповнена глибокого навчання. Допоміжні пристрої, такі як Amazon Alexa, наприклад, покладаються на алгоритми глибокого навчання, щоб реагувати на голос і знати уподобання. Глибоке навчання спирається на багатошарову структуру алгоритмів, звану штучною нейронною мережею. Глибоке навчання потребує величезних даних, але для належного функціонування вимагає незначного втручання людини.

Глибоке навчання дозволяє комп’ютеру будувати складні концепції з більш простих концепцій. Багатошаровий перцептрон – це просто математична функція, яка відображає деякий набір вхідних значень до вихідних значень (рис. 1.1).

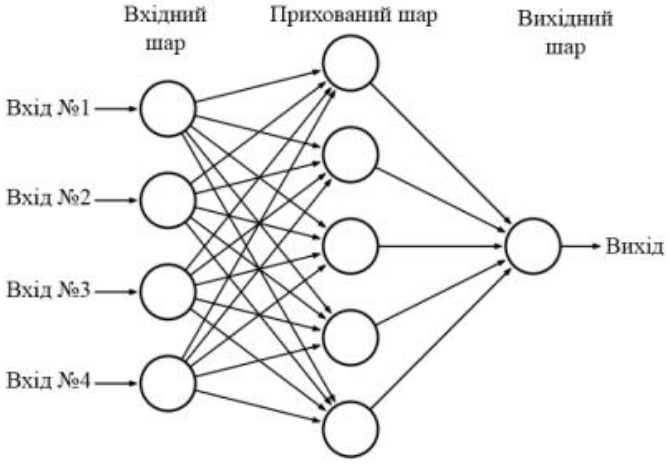


Рисунок 1.1 – Багатошаровий перцептрон [13]

Вхідний шар – перший – це вхідний шар. Цей рівень прийме дані та передасть їх решті мережі. Крайній лівий шар називається вхідним шаром, крайній правий шар вихідного. Середні шари називаються прихованими, оскільки їх значення не можна спостерігати в навчальному наборі. Простіше кажучи, приховані шари – це обчислені значення, які використовуються мережею для «магії». Чим більше прихованих шарів у мережі між вхідним і вихідним шарами, тим вона глибша. Загалом, будь-яка штучні нейронні мережі з двома або більше прихованими шарами називається глибокою нейронною мережею [14].

Прихований шар – другий тип шару називається прихованим шаром. Приховані шари для нейронної мережі – це один або кілька. У наведеному вище випадку число дорівнює 1. Приховані шари – це ті, які насправді відповідають за чудову продуктивність і складність нейронних мереж. Вони виконують кілька функцій одночасно, наприклад, перетворення даних, автоматичне створення функцій тощо.

Вихідний шар – останній тип шару – вихідний шар. Вихідний шар містить результат або вихід проблеми. Необроблені зображення передаються на вхідний шар, а отримуємо вихід у вихідному шарі. Нейрони в моделях глибокого навчання є вузлами, через які протікають дані та обчислення [15].

Шар складається з невеликих окремих одиниць, які називаються нейронами. Нейрон в нейронній мережі можна краще зрозуміти за допомогою біологічних нейронів. Штучний нейрон схожий на біологічний нейрон. Він отримує вхідні дані від інших нейронів, виконує певну обробку та виробляє вихід. Нейрони отримують один або кілька вхідних сигналів. Ці вхідні сигнали можуть надходити або з набору необроблених даних, або від нейронів, розташованих на попередньому рівні нейронної мережі. Потім виконуються деякі розрахунки. В результаті нейрони посилають деякі вихідні сигнали нейронам глибше в нейронній мережі через синапс.

Нейрони в моделі глибокого навчання здатні мати синапси, які

з’єднуються з більш ніж одним нейроном на попередньому шарі. Кожен синапс має відповідну вагу, що впливає на важливість попереднього нейрона в загальній нейронній мережі.

Вагові показники є дуже важливою темою в області глибокого навчання, оскільки коригування ваг моделі є основним способом навчання моделей глибокого навчання.

Як тільки нейрон отримує вхідні дані від нейронів попереднього рівня моделі, він складає кожен сигнал, помножений на його відповідну вагу, і передає їх функції активації.

Функція активації обчислює вихідне значення для нейрона. Це вихідне значення потім передається до наступного шару нейронної мережі через інший синапс. Функції активації є основною концепцією глибокого навчання. Саме вони дозволяють нейронам нейронної мережі спілкуватися один з одним через свої синапси.

Вагові коефіцієнти та зміщення (зазвичай іменовані як w і b) є параметрами, доступними для вивчення деяких моделей машинного навчання, включаючи нейронні мережі [16].

МН сьогодні не схоже на МН минулого. Воно було створене з розпізнавання образів і теорії, що комп’ютери можуть навчатися, не будучи запрограмованими для виконання конкретних завдань; дослідники, які цікавляться штучним інтелектом, хотіли перевірити, чи можуть комп’ютери вчитися на даних. Ітеративний аспект машинного навчання важливий, оскільки, коли моделі піддаються впливу нових даних, вони здатні самостійно адаптуватися. Вони вчаться на попередніх обчисленнях для отримання надійних, повторюваних рішень і результатів. Це наука, яка не нова, але набула нових обертів. За допомогою машинного навчання підприємства можуть автоматизувати рутинні завдання. Це також допомагає в автоматизації та швидкому створенні моделей для аналізу даних. Різні галузі залежать від величезної кількості даних для оптимізації своєї діяльності та прийняття розумних рішень. МН допомагає створювати

моделі, які можуть обробляти та аналізувати великі обсяги складних даних для отримання точних результатів. Ці моделі точні й масштабовані та функціонують із меншим часом виконання. Створюючи такі точні моделі машинного навчання, підприємства можуть використовувати прибуткові можливості та уникнути невідомих ризиків [17].

Розпізнавання зображень, генерація тексту та багато інших варіантів використання знаходять застосування в реальному світі. Це збільшує можливості експертів з машинного навчання показати себе як затребуваних професіоналів.

Нейрони є основними одиницями нейронної мережі. У штучній нейронній мережі кожен нейрон шару з’єднаний з деякими або всіма нейронами наступного шару. Коли входи передаються між нейронами, ваги застосовуються до вхідних даних разом із зміщенням [18].

Вага контролює сигнал і силу зв'язку між двома нейронами. Іншими словами, вага вирішує, який вплив матиме вхід на вихід.

Зміщення, які є постійними, є додатковим входом у наступний рівень, який завжди матиме значення 1. На одиниці зміщення не впливає на попередній рівень, але вони мають вихідні з'єднання зі своїми власними ваги. Одиниця зміщення гарантує, що навіть коли всі входи дорівнюють нулю, все одно буде активація в нейроні

Функція формується зі складання багатьох більш простих функцій. Можна подумати про використання різних математичних функцій для надання нового уявлення про вхідні дані. Ідея навчання правильно поданими даними – це перевага глибокого навчання. Інша перевага глибокого навчання полягає в тому, що вона дозволяє комп'ютеру ставати багатоступеневою комп'ютерною програмою. Кожен шар представлення може розглядатися як стан пам’яті комп'ютера після виконання іншого набору інструкцій паралельно. Мережі з більшою глибиною можуть виконувати більше інструкцій послідовно. Послідовні вказівки – це дуже потужний інструмент, оскільки пізніші інструкції можуть посилатися на

результати попередніх інструкцій [19].

Відповідно до цього погляду на глибоке навчання, не вся інформація в активізації шару обов’язково використовується для пояснення вхідних даних.

У нинішні часи штучний інтелект на основі нейронних мереж широко використовується при аналізі прихованих закономірностей для виявлення в історичних даних таких шаблонів, якими можна керуватися в майбутньому.

Нейронні мережі застосовувалися для вирішення практичних завдань усіх типів.

Однією з переваг нейронних мереж є те, що вони дозволяють вирішувати завдання, які виявляються занадто складними для звичайних технологій. Маються на увазі завдання, які не мають алгоритмічного рішення або, для яких алгоритмічне рішення є дуже складним, щоб його можна було визначити аналітично. Власне кажучи, нейронні мережі добре підходять для вирішення завдань, які успішно вирішують люди, але не можуть пояснити, як вони це роблять [20].

Також однією із переваг нейронної мережі є її простота у використанні. Для навчання нейронної мережі використовують приклади. Ці приклади користувач сам обирає в залежності від очікуваного результату. Хоча користувач виконує якусь роботу у налагодженні нейронної мережі, витрати при повному класичному розрахунку набагато більші чім від використанні нейронної мережі.

Також до основних переваг нейронних мереж як логічного базису алгоритмів слід віднести:

– інваріантність (незмінність, незалежність) методів синтезу нейронних мереж;

– можливість вибору структури нейронних мереж в значному діапазоні параметрів в залежності від складності та специфіки розв'язуваної задачі з метою досягнення необхідної якості рішення.

1.3 Аналіз методів розпізнавання об’єктів

Основне завдання розпізнавання зрозуміти, чи стосуються данні на зображенні шуканого об’єкту. Необхідно не просто виділити об’єкт, але й точно вказати в системі зображення його положення та розмір.

При розпізнаванні виникає ряд труднощів, які обумовлені такими чинниками:

– перешкоди та шум;

– складний текстурований фон, на якому відбувається виявлення об’єкту;

– ефекти загороджування одних об’єктів іншими об’єктами з невизначною формою;

– спотворюючі оптичні ефекти у вигляді різних розфокусувань, дисторсій об’єктивів, ракурсних спотворень і інші;

– різноманітність і змінність самих об’єктів. що призводить до змінної структури зображень;

– ефекти різкої зміни освітлення, відблиски, тіні;

– наявність дефектів та тимчасових змін форми об’єктів;

– зміна середовища розповсюдження світла між сенсорами та об’єктами спостереження (пил, дим, опади);

– несинхронність реєстрації та обробки даних, яка пов’язана з обмеженням швидкодії комп’ютерних засобів зберігання та аналізу зображень. А також перебої в комп’ютерних програмах обробки.

Для вирішення задачі розпізнавання об’єктів на зображенні на різних етапах застосовують різні методи і прийоми. На рис. 1.2 представлені основні методи, що існують і застосовуються на сьогоднішній день.

Етап підготовки зображення до розпізнавання це поліпшення його візуальної якості: усунення розмитості, підвищення контрасту,

підкреслення границь, фільтрація тощо.

Задача розпізнавання комплексна і ділиться на етапи:

– попередня обробка;

– сегментація;

– фільтрація;

– розпізнавання об’єктів.

Рисунок 1.2 – Основні методи розпізнавання об’єктів [21]

Всі ці етапи використовуються при обробці зображень від початкового етапу до розпізнавання зображення.

Для підвищення ефективності та якості розпізнавання об’єктів на зображенні передує попередня обробка зображень. Методи попередньої обробки залежать від мети розпізнавання і їх досить багато.

Основна мета попередньої обробки це зниження перешкод на зображенні, а також придушення зовнішніх шумів [22].

Сегментація – це поділ зображення на сегменти. Основна мета сегментації це спрощення зображення, щоб його легше було аналізу. Нині семантична сегментація є однією з ключових проблем у сфері комп’ютерного зору. Розглядаючи загальну картину, семантична сегментація є одним із завдань високого рівня, що відкриває шлях до повного розуміння сцени. Важливість розуміння сцени як основної проблеми комп'ютерного зору підкреслюється тим фактом, що все більша кількість додатків живиться від виведення знань із зображень. Деякі з цих додатків включають самокеровані транспортні засоби, взаємодію людини та комп’ютера, віртуальну реальність тощо. Семантична сегментація є природним кроком у прогресі від грубого висновку до точного: початок може бути розташовано в класифікації, яка складається з передбачення для всього вхідного даних. Наступним кроком є локалізація / виявлення, які забезпечують не лише класи, а й додаткова інформація щодо просторового розташування цих класів. Нарешті, семантична сегментація досягає тонкого висновку, роблячи щільні передбачення, виводячи мітки для кожного пікселя, так що кожен піксель позначається класом його охоплюючого об'єкта або регіону. З популярністю глибокого навчання в останні роки багато проблем семантичної сегментації вирішуються за допомогою глибоких архітектур, найчастіше згорткових нейронних мереж, які перевершують інші підходи. з великим відривом щодо точності та ефективності. Сегментація основується на двох принципах: розривності та подібності [23].

При принципі розривності основний підхід базується на визначенні контурів.

При принципі подібності орієнтуються на визначення порогового рівня та нарощування областей.

Результат сегментації є множина, яка покриває все зображання або

множина виділена із зображення.

Морфологічні методи – використовуються для роботи з чорнобілими зображеннями. Ці методи частіше використовують для подання форми об’єкту.

Порогова сегментація – особливість цього методу полягає в тому що вибирається порогове значення. Розділяють метод з оптимальним порогом і метод з адаптивним порогом.

У методах глобальної бінарізації порогова поверхня є постійним значенням порогової яскравості.

Методи локальної бінарізації змінюють значення порогової яскравості від точки до точки зображення, розрахунок проводиться в зоні локальних ознак в оточенні пікселя.

Основна мета бінарізації це зменшення кількості інформації.

Метод Бернсена самий швидкий серед інших методів, але при використанні цього методу для обробки однорідних зон яскравості з’являються помилкові чорні плями, що приводить до додаткової обробки – постпроцесинг.

Метод Еквеля підходить для використання обробки чітких і контрастних зображень. При обробці зображень з низькою контрастністю можуть виникати розриви та помилкові чорні області.

При обробці зображень яким властива нерівномірна яскравість використовують метод Яновіц і Брукштейна, але серед методів локальної бінарізації він найменш продуктивний.

Метод нарощування областей – це поділ зображення на області, які мають подібні властивості.

Метод нарощування областей використовують при обробці зображень на яких присутні шуми. Недоліком цього методу є те, що він виділяє загальні фрагменти, не показуючи змін яскравості всередині області.

В основі методу злиття з аналогічними областями лежить вибір центрів областей, до яких по черзі приєднуються сусідні точки, що

відповідають певним критеріям. Процес зупиняється тоді, коли не одна точка не приєднується до області. Критерії на підставі яких точка приєднується або не приєднуються до області дуже різні: близкість до центру області, близкість за деякою статистикою, близькість до сусідньої точки [24].

Одним із основних методів розщеплення є метод водорозділу. Суть методу водорозділу полягає в тому, що спочатку будується зона контрасту зображення, а потім водорозділи (області високої контрастності). Отримані перегородки є лінії водорозділу.

Комбінування двох методів складається з розщеплення і злиття. Спочатку зображення розбивається на області. Процес розщеплення областей відбувається до тих пір, доки не отримано однорідності сегментів в розщепленні. Після цього іде процес злиття схожих сусідніх сегментів, поки не буде отримано розщеплення зображення на однорідні області максимального розміру [25].

Метод кластерного аналізу використовують для розщеплення зображення к-кластерів.

Текстура є яскравою ознакою сегментації. Один із способів текстурної сегментації є підрахунок міри зернистості текстури у всіх точках зображення і подальшою обробкою.

Вхідне зображення обробляють з метою перетворення його в зображення по яскравості.

Параметри текстури виміряють у вікні, значення отримані в межах кордону між текстурними областями будуть усередненими. Тому важко точно визначити кордон між текстурними областями.

Задача розпізнавання текстур є перспективною і потрібною. Існує велика кількість методів для її розв’язання. Вони відрізняються кількістю елементів текстур, представленням елементів розміщення відносно сусідів.

Потрібно обирати такі методи та підходи, які стануть складовою у розпізнаванні на зображенні [26].

Одним із методів контурної сегментації – є метод виділення границь.

Цей метод набув поширення в силу простоти свого використання. Основою для побудови контурних методів є граничні детектори, які контролюють роботу граничних пікселів зображення по контрасту використовуючи маски.

Для виявлення границь, використовують маски, що створюються на перетворенні Фур'є. Їх використання складне і більш трудомістке.

До переваг контурних методів можна віднести простоту вирішення, стійкість до незначних перепадів яскравості і контрастності зображень. До недоліків складно усунення помилкових границь при низькій контрастності, а також складність при усуненні розривів границі при підвищеному порозі.

Більшість зображень піддається впливу різного роду шумів. Наявність шумів на зображення може причинити неточності та спотворення, що приведе до ускладнення процесу розпізнавання зображення [27].

Фільтрація зображення проводиться з метою усунення різних шумів на зображенні та отриманні бажаного результату.

Виділяють частотну і просторову фільтрацію.

Частотні методи базуються на методі Фур'є. У випадках, якщо функція є періодичною уявлення називають рядами Фур'є. Неперіодичну функцію, виражену у вигляді інтеграла від тригонометричних функцій, помножених на вагову функцію називають перетворенням Фур'є.

Просторова фільтрація застосовується для покращення зображень і задач відновлення зображень.

Покращення зображення – це надання зображенню форми, яка є придатна для подальшої обробки. Це видалення із зображення перешкод та шуму, підкреслення границь, задачі зміни контрасту.

Відновлення зображення – це отримання зображення, яке є близьким до ідеалу.

Розпізнавання об'єктів – це кінцевий етап обробки зображень. Для вирішення задач розпізнавання використовують такі методи [28]:

– кореляційні методи – засновані на пошуку максимальної кореляції точок шаблону і областей зображення. Ці методи використовують при спостереженні за роботою машин, виявлення і розпізнаванні зображень в системі навігації, охоронній системі та системі розпізнання і т. д.;

– ознакові методи – це методи, які використовують виділення і зіставлення характерних ознак шуканого об’єкта з областю зображення, яка має ці ознаки. Основні проблеми при використанні цього методу – це вибір ознак;

– синтаксичний метод – заснований на отриманні структурно-лінгвістичних ознак. Правило з’єднання цих ознак, однакові для еталону і вхідного зображення;

– методи нормалізації – мета цих методів полягає в автоматичному обчисленні невідомих параметрів перетворень, якими є вхідні зображення і приведе їх до вигляду еталона;

– перетворення здійснюється за допомогою операторів нормалізації, а обчислення параметрів – функціоналами.

1.4 Можливості штучного інтелекту для розвитку промислової автоматизації

З часів промислової революції людство досягло величезних успіхів у виробництві. З часом все частіше бачимо, як рутинна ручна робота замінюється автоматизацією за допомогою передової інженерії, комп’ютерів, робототехніки, а тепер і IoT. Останні досягнення в галузі штучного інтелекту або, якщо бути точнішим, глибокого навчання допоможуть привабливо прискорити цю тенденцію до автоматизації. Це пояснюється тим, що штучний інтелект додає один дуже важливий компонент, якого на заводах не вистачало до сьогодні – «здатність машин бачити». За допомогою роботів з комп’ютерним зором тепер можна досліджувати багато нових та незвіданих напрямків автоматизації.

Виявлення об’єктів – це комп’ютерна технологія, пов’язана з комп’ютерним зором та обробкою зображень, яка займається виявленням екземплярів семантичних об’єктів певного класу (наприклад, людей, будівель чи автомобілів) у цифрових зображеннях та відео. В останні роки методи глибокого навчання досягають найсучасніших результатів для виявлення об’єктів, наприклад, на стандартних наборах даних еталонів та у змаганнях з комп’ютерного зору. Примітним є сімейство нейронних мереж, які досягають майже найсучасніших результатів за допомогою єдиної наскрізної моделі, яка може виконувати виявлення об’єктів у режимі реального часу.

Комп’ютерне бачення – це область, яка займається розширенням можливостей комп’ютерів «бачити» такі речі, як люди. Виявлення об’єктів – це основне завдання візуального сприйняття та одна з ключових сфер застосування комп’ютерного зору. По суті, це стосується пошуку та розташування конкретних об’єктів у зображенні. Для виявлення загальних об’єктів (таких як автомобіль, людина, стіл, дерево) доступні відкриті та попередньо навчені моделі, такі як YOLO.

YOLO – це алгоритм, який виявляє і розпізнає різні об’єкти на зображенні в режимі реального часу. Виявлення об’єктів у YOLO виконується як проблема регресії і забезпечує імовірність класу виявлених зображень. Цей алгоритм використовує згорткові нейронні мережі (CNN) для виявлення об’єктів у режимі реального часу. Як випливає з назви, алгоритм вимагає лише одного поширення вперед через нейронну мережу для виявлення об’єктів. Це означає, що передбачення у всьому зображенні здійснюється за допомогою одного алгоритму. CNN використовується для прогнозування ймовірностей різних класів та обмежувальних блоків одночасно [29].

Алгоритм можна використовувати в автономних автомобілях для виявлення об’єктів навколо автомобілів, таких як транспортні засоби, люди та сигнали паркування. Виявлення об'єктів в автономних автомобілях

проводиться, щоб уникнути зіткнення, оскільки жоден водій не керує автомобілем. Також, він використовується для виявлення різних типів тварин у лісах. Даний тип виявлення використовується диспетчерами дикої природи та журналістами для ідентифікації тварин у відеозаписах (як записаних, так і в режимі реального часу) та зображеннях. Деякі з тварин, яких можна виявити, включають жирафів, слонів та ведмедів. В системах безпеки для забезпечення безпеки в зоні. Припустимо, що людям заборонено проходити через певну територію з міркувань безпеки. Якщо хтось пройде через зону заборони, алгоритм YOLO виявить його/її, що вимагатиме від персоналу безпеки подальших дій (рис. 1.3).

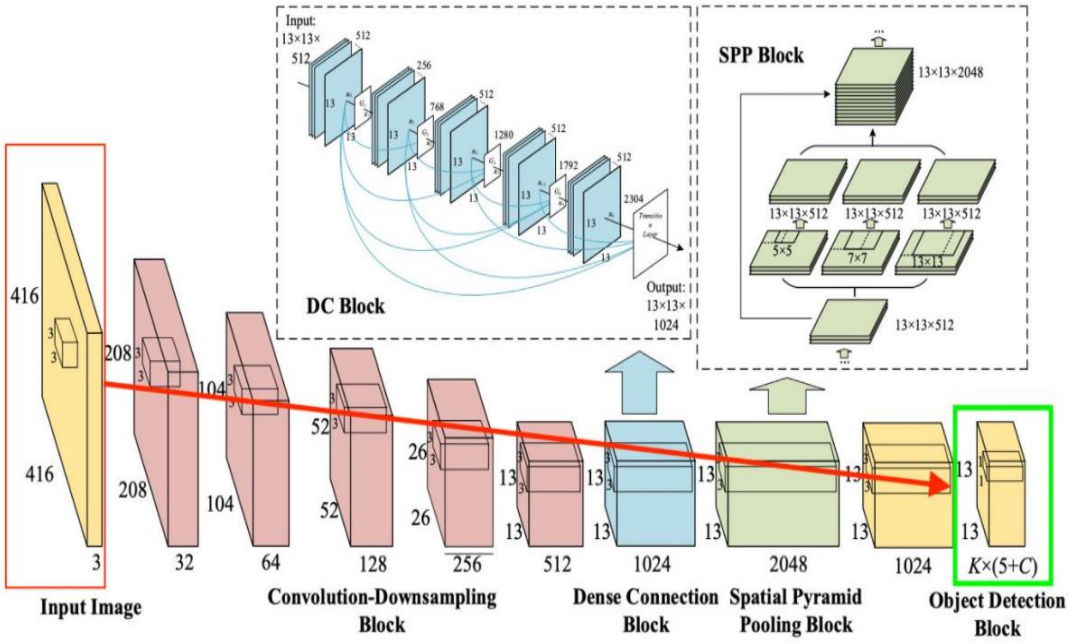


Рисунок 1.3 – Модель роботи YOLO [30]

Дана уніфікована модель має ряд переваг перед традиційними методами виявлення об’єктів. По-перше, вона надзвичайно швидка. Оскільки ми визначаємо виявлення як проблему регресії, складний конвеєр не потрібен. Ми просто запускаємо свою нейронну мережу на новому зображенні під час випробування, щоб передбачити виявлення. Наша базова мережа працює зі швидкістю 45 кадрів в секунду без пакетної обробки на

графічному процесорі Titan X, а швидка версія працює зі швидкістю понад 150 кадрів в секунду. Це означає, що ми можемо обробляти потокове відео в режимі реального часу із затримкою менше 25 мілісекунд. Модель бачить всі зображення під час навчання та тестування, тому вона неявно кодує контекстну інформацію про класи, а також їх зовнішній вигляд.

Управління запасами може бути дуже складним, оскільки елементи важко відстежити в режимі реального часу, що небудь щодня додається, видаляється та переміщується. Неякісне управління запасами може завдати шкоди компанії як з точки зору капіталу, так і часу. Система ШІ може виконувати автоматичний підрахунок та локалізацію об’єктів, що дозволить підвищити точність інвентаризації. Автоматизація штучного інтелекту усуває людські помилки з рівняння, точно підраховуючи запаси та вихідні запаси. В автоматизованому режимі підприємства замовлятимуть потрібну кількість продукції за найкращою ціною, гарантуючи, що гроші не витрачаються на неточні або сторонні замовлення.

Частина контролю якості виробничого циклу продовжує залишатися складним завданням через свою залежність від візуального розуміння на людському рівні та адаптації до постійно мінливих умов та продуктів.

За допомогою ШІ більшість із цих ускладнень можна впоратися. ШІ може автоматично відрізняти хороші деталі від несправних деталей на конвеєрі з неймовірною швидкістю, що дає достатньо часу для вжиття коригувальних заходів. Це дуже корисне рішення для динамічних середовищ, де середовища продуктів постійно змінюються, а час цінний для бізнесу [31].

За допомогою відстеження об’єктів на базі штучного інтелекту об’єкти класифікуються відповідно до параметра, обраного виробником, і відображається статистика кількості об’єктів.

Це значно зменшує відхилення у категоризації та робить гнучку конвеєрну лінію. Наприклад, у сільськогосподарських галузях сортування відіграє вирішальну роль на конвеєрі. Компанії вкрай необхідно виявити та

викинути пошкоджені фрукти/овочі, які можуть вплинути на готовий продукт. Виявлення об'єктів на базі ШІ може допомогти перетворити цей виснажливий та ручний процес в ефективний та автоматизований [32].

Приклад розпізнавання мотоциклів, велосипедів та людей представлений на рисунку 1.4.

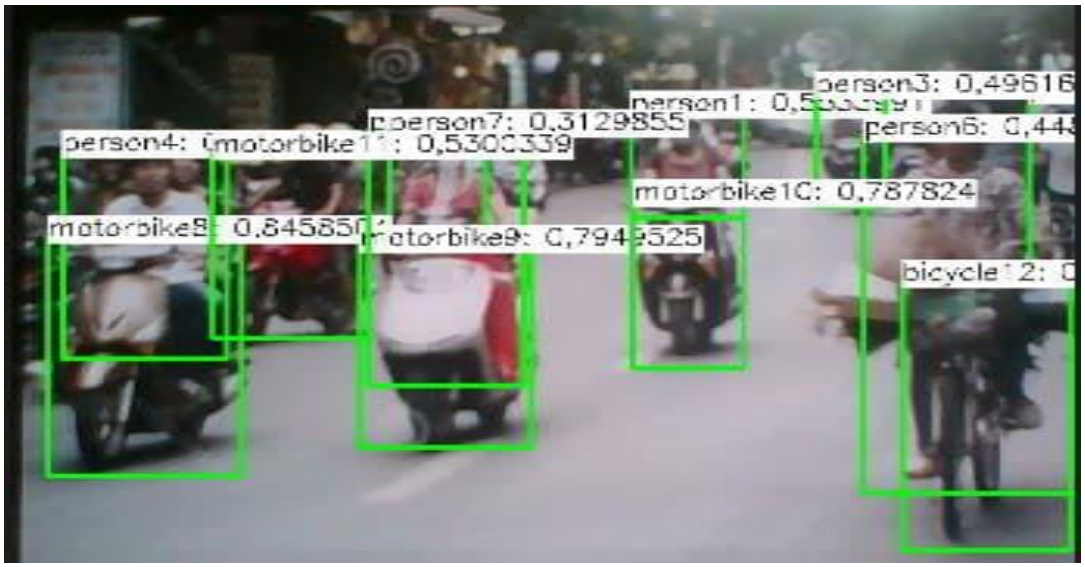


Рисунок 1.4 – Приклад розпізнавання мотоциклів, велосипедів та людей

Приклад розпізнавання машин представлений на рис. 1.5.

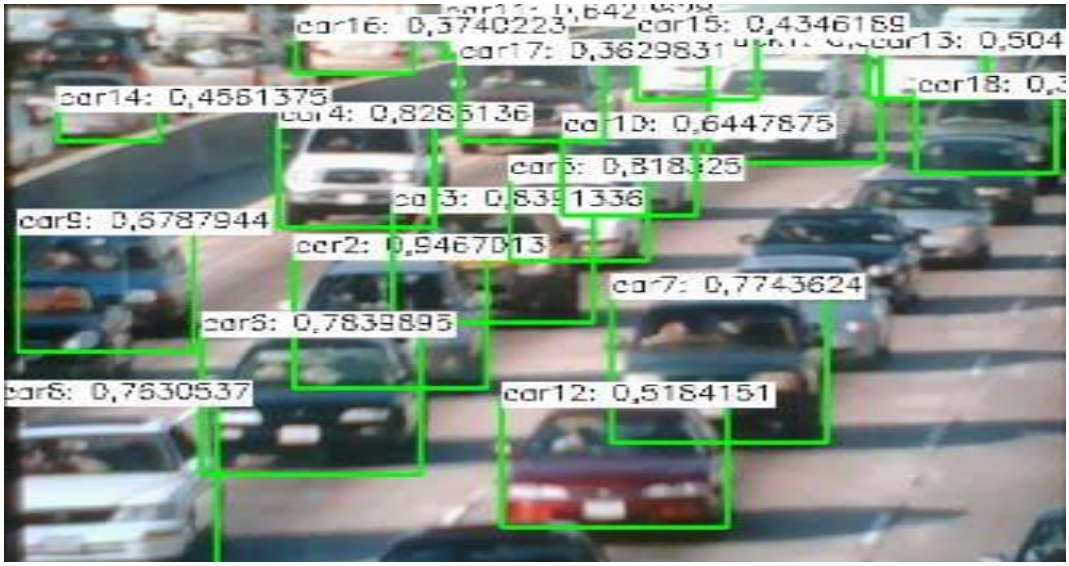


Рисунок 1.5 – Приклад розпізнавання машин

Існує декілька проблем, які необхідно враховувати під час визначення індивідуальних об'єктів для ношеного варіанту використання у виробничій установі. Об’єкти бувають різних форм, розмірів, орієнтації, кольорів, а фабричне середовище в реальному світі має додатковий шум, що виникає внаслідок зміни точки зору, освітлення, оклюзій та тіней. З боку алгоритму, потрібно переконатися, що бажана точність досягається без необхідності упорядковувати занадто багато навчальних прикладів.

1.5 Висновки до 1 розділу

Існує велика кількість методів обробки зображень. Що дає змогу поліпшити якість аналізу і прискорити його. Але не існує універсального методу обробки.

Тому доцільно використовувати методи під конкретні види зображень, що підвищить ефективність при обробці зображень.

При виборі нейронної моделі слід враховувати: структурне уявлення мережі, підбір вхідних даних, можливість використання нейронної мережі для обраної системи, здатність мережі до навчання, можливість пошуку оптимального рішення задачі.

Аналіз характеристик нейронних мереж дозволив обрати найбільш оптимальну для поставленої мети модель, а саме багатошаровий персептрон, який є підвидом односпрямованих багатошарових мереж.

2 РОЗРОБКА СТРУКТУРНОЇ СХЕМИ МОДУЛЮ РОЗПІЗНАВАННЯ ТРАНСПОРТНИХ ТЕХНОЛОГІЧНИХ ОБ’ЄКТІВ ДЛЯ РОБОТИЗОВАНОЇ ПЛАТФОРМИ

2.1 Розробка структурної схеми системи розпізнавання технологічних об’єктів

При побудові роботизованої платформи не обійтися без структурної схеми рис 2.1. Структурна схема роботизованої платформи в даній атестаційній роботі також має модуль розпізнавання об’єктів.

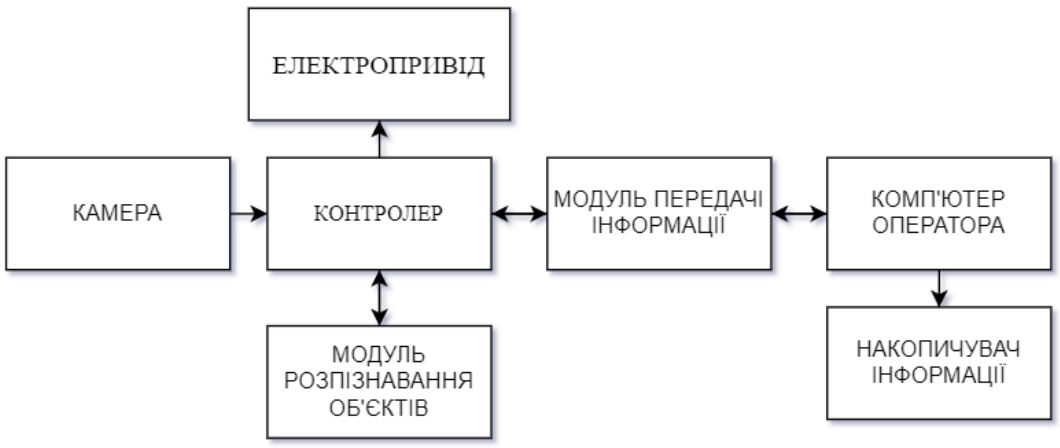


Рисунок 2.1 – Структурна схема роботизованої платформи

Розроблена структурна схема системи розпізнавання дає змогу здійснити підбір вже конкретних елементів з певними характеристиками. Передача інформації може здійснюватися бездротовим або дротовим методом, модулем передачі інформації може бути Wifi модуль якщо платформа мобільна, якщо платформа є статичною то передача інформації може здійснюватися за допомогою дротового методу.

Схема автоматизованого модулю розпізнавання має комплексну

структуру та включає в себе наступні рівні такі як: визначені функціональне, інформаційне, математичне, алгоритмічне, програмне та технічне забезпечення, схема представлена на рис. 2.2

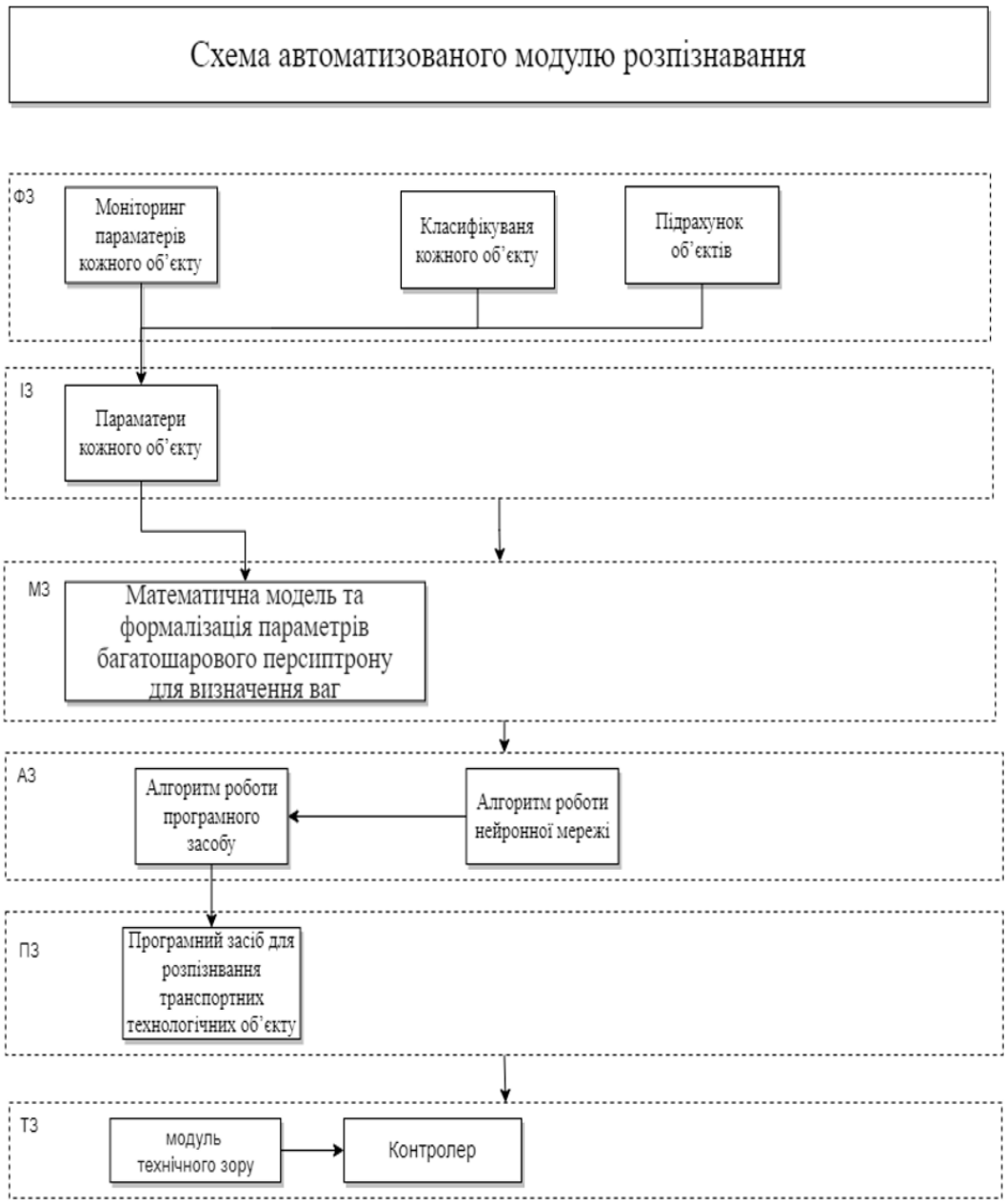


Рисунок 2.2 – Схема автоматизованого модулю розпізнавання

Модуль розпізнавання є гнучким, бо в залежності від використаної мережі, її конфігурації, ваг та файлу з класами об'єктів, метод може змінювати свою поведінку та розпізнавати інші типи об'єктів, якщо це потрібно.

Виходячи зі схеми автоматизованого модулю розпізнавання, були

сформовані наступні вимоги-завдання до функціональних характеристик ПЗ, які необхідно урахувати ри розробці на наступних етапах:

– розробити модель роботи програмного засобу;

– розробити модель навчання нейронної мережі;

– розробити математичну модель та формалізувати параметри багатошарового персиптрону для визначення ваг;

– розробити алгоритмічно-програмне забезпечення;

2.2 Висновки до розділу 2

В ході виконання другого розділу було розроблено структурну схему роботизованої платформи, що містить модуль розпізнавання технологічних об’єктів, яка дозволяє зрозуміти принцип його роботи. Також було розроблено схему автоматизованого модулю розпізнавання, на базі якої було сформовано вимоги до функціональних можливостей розроблюваного програмного забезпечення.

3 РОЗРОБКА МЕТОДУ РОЗПІЗНАВАННЯ ТЕХНОЛОГІЧНИХ ТРАНСПОРТНИХ ОБ'ЄКТІВ

3.1 Математичне обґрунтування вибору нейронної мережі для розпізнавання транспортних технологічних об'єктів

Типові біологічні нейрони – це окремі клітини, кожна з яких складається з основної частини клітини разом із безліччю вусиків, що відходять від цього тіла (рис 3.1). Тіло, або сома, містить механізми для підтримки основних функцій клітини та переробки енергії (наприклад, ядро, що містить ДНК, і органели для побудови білків та переробки цукру та кисню). Існують два типи вусиків: дендрити, які отримують інформацію від інших нейронів і доставляють її до тіла клітини, і аксони, які передають інформацію від тіла клітини до інших нейронів. Передача інформації від нейрона, що передає, до нейрона, що приймає, приблизно складається з трьох етапів. По-перше, нейрон, що передає, генерує просторово та тимчасово обмежений електричний сплеск або спайк, який рухається вздовж аксона нейрона і аксональних гілок від тіла клітини до кінцевих кінців аксона. Термінал аксона передавального нейрона «з'єднаний» з дендритом приймального нейрона синапсом. Спайк змушує синапс нейрона, що передає, вивільняти хімічні речовини або нейромедіатори, які долають коротку відстань між двома нейронами за допомогою дифузії [33].

Електричний синапс: щілина з'єднання між двома протилежними нейронами, що забезпечує швидшу передачу нерва. Форма синапсу між двома приєднаними нейронами, в якій передача нервового імпульсу відбувається швидко і відбувається шляхом проходження іонів від одного нейрона до іншого через щілинний з'єднання. В електричному синапсі нервові імпульси передаються шляхом проходження іонів від одного нейрона до іншого через канали щілинного з'єднання. У порівнянні з

хімічним синапсом, електричний синапс швидше проводить нервові імпульси. Він використовується завжди, коли швидка реакція та координація часу мають вирішальне значення, як-от у рефлексах втечі, сітківці хребетних і серці. Крім того, передача нервового імпульсу в електричних синапсах також двонаправлена, хоча в деяких вона односпрямована [34].

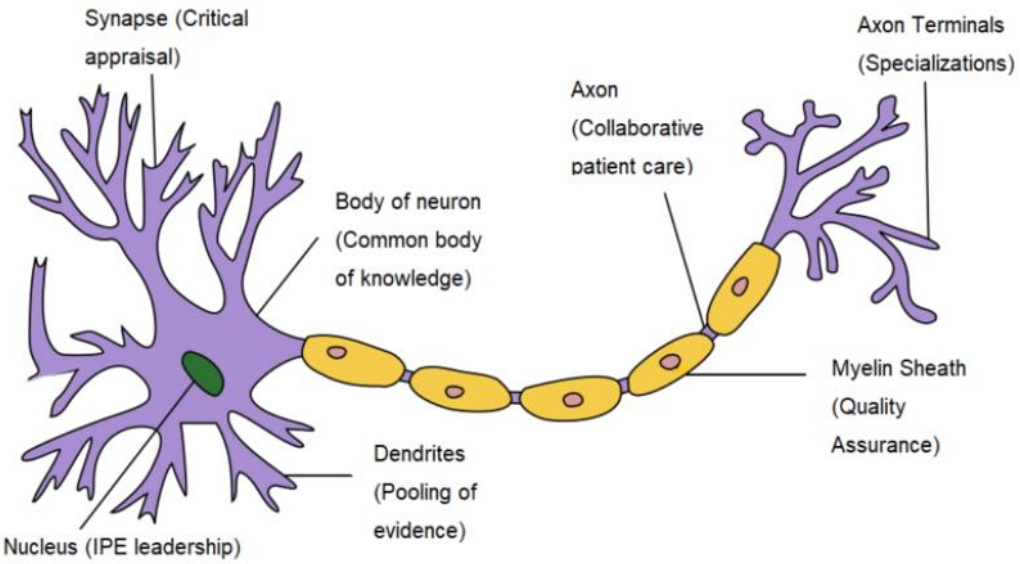


Рисунок 3.1 – Біологічний нейрон [35]

Спрощені моделі біологічних нейронів, як описано вище, можуть бути зібрані для формування стереотипного нейрона в моделях глибокого навчання.

Нейрон глибокого навчання отримує вхідні дані або активації від інших нейронів. Активації – це закодовані швидкості уявлення про стрибки біологічних нейронів.

Активації помножуються на синоптичні ваги. Ці коефіцієнти є моделями синоптичної сили в біологічних нейронах, а також моделлю гальмівної передачі, оскільки ваги можуть приймати негативні значення. І ваги, і зміщення є параметрами, які можна вивчати всередині мережі. НМ, яка піддається навчанню, випадковим чином визначає значення ваги та

зміщення до початку навчання. Під час навчання обидва параметри коригуються до бажаних значень і правильного результату. Ці два параметри відрізняються за ступенем впливу на вхідні дані. Просто упередження показує, наскільки далекі прогнози від їх очікуваного значення. Зміни складають різницю між результатом функції та її передбачуваним результатом. Низьке значення зміщення свідчить про те, що мережа робить більше припущень щодо форми результату, тоді як високе значення зміщення робить менше припущень щодо форми результату. З іншого боку, ваги можна розглядати як міцність зв'язку. Вага впливає на величину впливу зміни вхідних даних на вихід. Низьке значення ваги не змінюватиме вхідні дані, а більше значення ваги значно змінить вихідні дані. Зважені активації підсумовуються разом, моделюючи процес накопичення, який відбувається в тілі клітини біологічного нейрона. До суми додається член зміщення, що моделює загальну чутливість нейрона [36].

Для створення математичних моделей для штучних нейронних мереж необхідний теоретичний аналіз біологічних нейронних мереж, оскільки вони мають дуже тісний взаємозв'язок. І це розуміння нейронних мереж мозку відкриває горизонти для розвитку штучних нейронних мереж і адаптивних систем, призначених для навчання та адаптації до ситуацій та вхідних даних[37].

Нарешті, підсумкове значення формується функцією активації (3.1) – зазвичай тією, яка обмежує мінімальне або максимальне вихідне значення (або обидва), наприклад, сигмовидна функція. Це моделює внутрішню мінімальну швидкість біологічних нейронів (нуль) або максимальну швидкість (через деталі фізіологічних механізмів, за допомогою яких генеруються спайки).

$$sum = \sum_{i=1}^{n} x_i \cdot w_i. \qquad (3.1)$$

Функція активації вирішує, чи слід активувати нейрон чи ні. Це означає, що він вирішуватиме, чи важливий вхід нейрона в мережу чи ні в процесі передбачення за допомогою простіших математичних операцій.

Роль функції активації полягає в отриманні виводу з набору вхідних значень, що подаються до вузла або шару [38].

Вхідний рівень отримує вихідні дані з домену. На цьому рівні обчислення не виконуються. Тут вузли просто передають інформацію (функції) на прихований шар.

Прихований шар виконує всі види обчислень для об’єктів, введених через вхідний шар, і передає результат на вихідний шар [39].

Вихідний шар – це останній рівень мережі, який переносить інформацію, засвоєну через прихований шар, і в результаті надає остаточне значення.

Усі приховані шари зазвичай використовують ту саму функцію активації. Однак вихідний шар зазвичай використовує функцію активації, відмінну від прихованих шарів. Вибір залежить від мети або типу передбачення, зробленого моделлю [40].

Вага контролює сигнал (або силу зв'язку) між двома нейронами. Іншими словами, вага визначає, який вплив матиме вхід на вихід [41].

Зміщення, які є постійними, є додатковим входом у наступний рівень, який завжди матиме значення 1. На одиниці зміщення не впливає попередній рівень (у них немає вхідних з'єднань), але вони мають вихідні з'єднання зі своїми власними ваги. Одиниця зміщення гарантує, що навіть коли всі входи дорівнюють нулю, все одно буде активація в нейроні [42].

Алгоритм прямого поширення уперед – потік інформації відбувається в прямому напрямку. Вхідні дані використовуються для обчислення деякої проміжної функції в прихованому шарі, яка потім використовується для обчислення результату. Це біологічно натхненний алгоритм класифікації на рис 3.2. Він складається з великої кількості простих нейроноподібних блоків обробки, організованих у шари. Кожна одиниця в шарі пов’язана з усіма

одиницями попереднього шару. Ці з’єднання не всі однакові: кожне з’єднання може мати різну міцність або вагу. Вагові коефіцієнти цих з'єднань кодують знання мережі [43].

У прямому поширенні функція активації є математичними «воротами» між входом, що живить поточний нейрон, і його виходом, що йде на наступний шар.

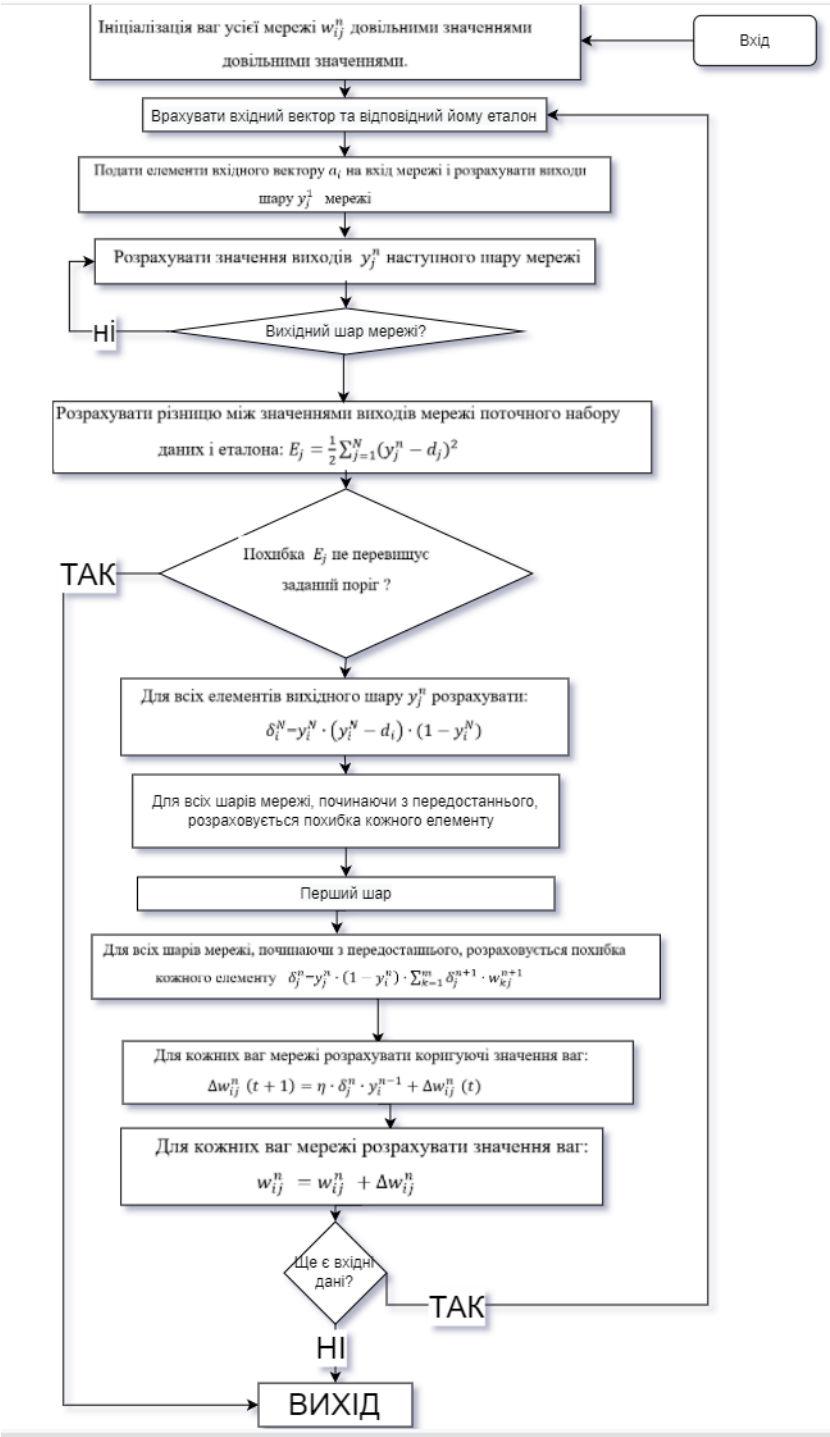
Алгоритм протилежного поширення – ваги мережевих з'єднань багаторазово коригуються, щоб мінімізувати різницю між фактичним вихідним вектором мережі та бажаним вихідним вектором [45].

Простіше кажучи – обернене поширення має на меті мінімізувати функцію витрат шляхом коригування ваги та зміщень мережі. Градієнти функції вартості визначають рівень коригування щодо таких параметрів, як функція активації, ваги, зміщення тощо [46].

Глибоке навчання покладається на кодування на основі швидкості, при якому активація кожного нейрона є єдиним числовим значенням, яке моделює середню швидкість стрибків у відповідь на даний стимул (будь то від інших нейронів або від зовнішнього стимулу). Загальний набір значень швидкості стрибків у межах одного шару мережі зазвичай організовується як вектор чисел, і цей вектор називають представленням зовнішнього стимулу на цьому рівні [47].

Експресивність нейронного кодування на основі швидкості набагато нижча, ніж та, яка можлива з нейронними кодами (уявленнями), заснованими на відносному часу між скачками на кількох нейронах. Як простий приклад існування такого типу коду в біології розглянемо слухову систему. Коли звукові хвилі досягають наших вух, наш мозок обробляє їх, щоб визначити тип тварини, об’єкта або явища, які викликали звук, а також визначити напрямок, звідки прийшов звук (локалізація). Один із способів визначення розташування звуку заснований на тому, що звуки з правого вуха спочатку потрапляють до правого, а потім до лівого вуха. Слухові нейрони поблизу правого і лівого вух демонструють стрибки, що

відображає цю різницю акустичного часу. Слухові нейрони, розташовані більш медіальною (поблизу середньої лінії тіла), отримують вхід від нейронів біля обох вух і є селективними щодо розташування звуку завдяки цьому тимчасовому кодуванню [48].



де w_{ij}^n – значення ваг між вхідним вектором та першим шаром нейронів, поміж *i*-им и *j*-тм нейронами двох шарів, *n* – номер шару;

a_i – нормований елемент *i*-го вхідного вектора вектора;

y_j^n – значення *j*-го виходу, *n* – номер шару;

d_j – потрібне значення *j*-го виходу;

δ_i^N – похибка *i*-го виходу, *N* – номер вихідного шару;

w_{kj}^{n+1}значення ваг між вхідним вектором, *n* – номер шару, k – індекс елементів шару, що посилає похибку;

t – номер ітерації;

η – коефіцієнт швидкості навчання рівний $0 < \eta < 1$.

Рисунок 3.2 – Узагальнений алгоритм прямого поширення [44]

Хоча сьогодні перцептор широко відомий як штучний нейрон, спочатку він був призначений як машина для розпізнавання зображень. Свою назву він отримав від виконання людської функції сприйняття, бачення та розпізнавання зображень.

Зокрема, інтерес був зосереджений на ідеї машини, яка була б здатна концептуалізувати вхідні дані, що впливають безпосередньо з фізичного середовища світла, звуку, температури тощо – «феноменального світу», з яким ми всі знайомі, – а не вимагає втручання людини для перетравлення та кодування необхідної інформації [49].

Персептронна машина Розенблата спиралася на основну обчислювальну одиницю – нейрон. Як і в попередніх моделях, кожен нейрон має клітинку, яка отримує серію пар вхідних даних і ваг.

Основна відмінність моделі Розенблата полягає в тому, що вхідні дані об’єднуються у зважену суму, і, якщо зважена сума перевищує заздалегідь визначений поріг, нейрон активується. Якщо зважена сума вхідних даних більша за нуль, нейрон виводить значення 1, інакше вихідне значення дорівнює нулю і нейрон не активується (рис 3.3).

Персептрон складається з 3 основних частин:

– вузли введення або вхідний рівень: вхідний рівень приймає вихідні дані в систему для подальшої обробки. Кожен вхідний вузол пов'язаний з

числовим значенням. Він може мати будь-яку реальну цінність;

– вага та упередження: параметри ваги представляють силу зв'язку між одиницями. Чим вища вага, тим сильніше вплив пов’язаного вхідного нейрона, щоб визначити вихід. Зміщення виконується так само, як і перехоплення в лінійному рівнянні;

– функція активації визначає, спрацює нейрон чи ні. У найпростішому випадку функція активації є кроковою функцією, але на основі сценарію можна використовувати різні функції активації.

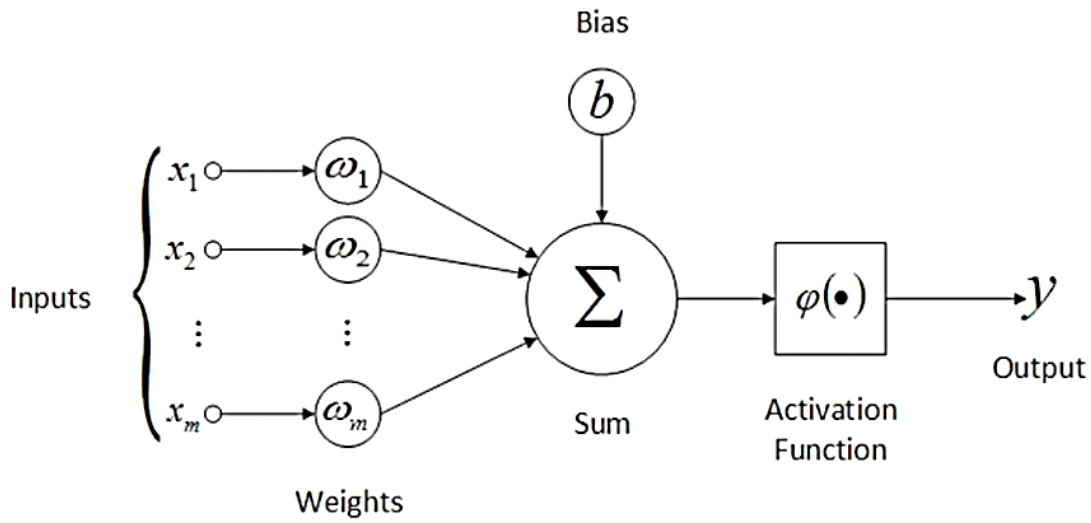


Рисунок 3.3 – Одношаровий персептрон [50]

Багатошаровий перцептрон, який наведено на рис 3.4, був розроблений для вирішення складних проблем де на вході моделі мають більш ніж 2 параметри. Це НМ, де відображення між входами та виходом є нелінійним. Загалом кажучи, в порівнянні з одношаровим перцептроном де звичайно використовується лінійна функція, в багатошаровому перцептроні зазвичай для функції активації використовується нелінійна функція. Наприклад: сигмоїда або функція активації гіперболічного тангенсу рис (3.5-3.6) якщо нам потрібні результати менше 0 [51].

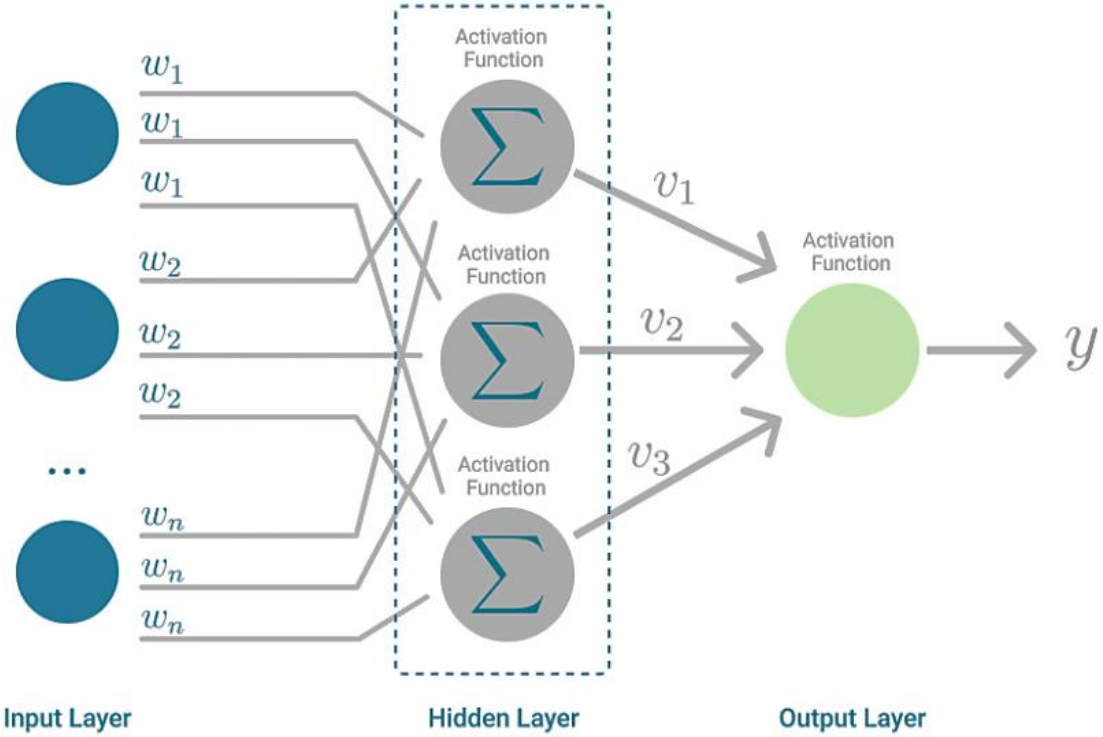


Рисунок 3.4 – Багатошаровий персептрон [52]

$$y \geq 0,\ y\ = \frac{1}{1+e^{-x||sum}} = \frac{1*e^{x||sum}}{1+e^{x||sum}},\ -\infty < x < \infty. \tag{3.2}$$

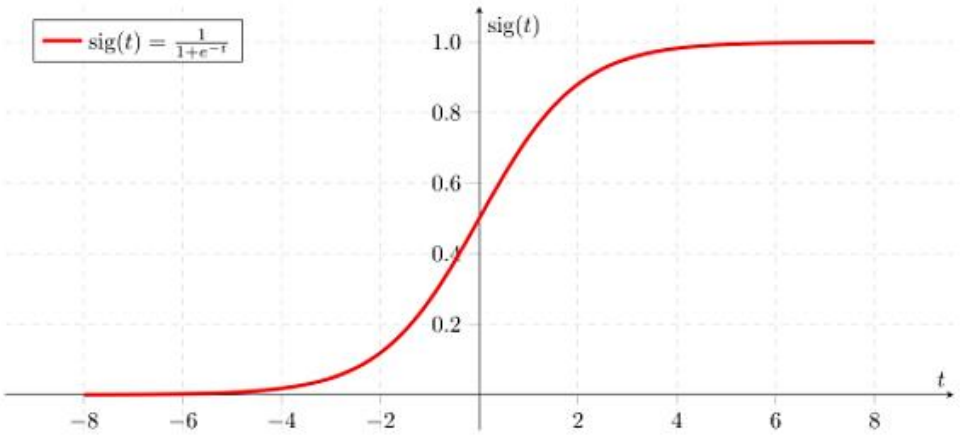


Рисунок 3.5 – Графік сигмоїди

$$y\ = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}},\ -\infty < x < \infty. \tag{3.3}$$

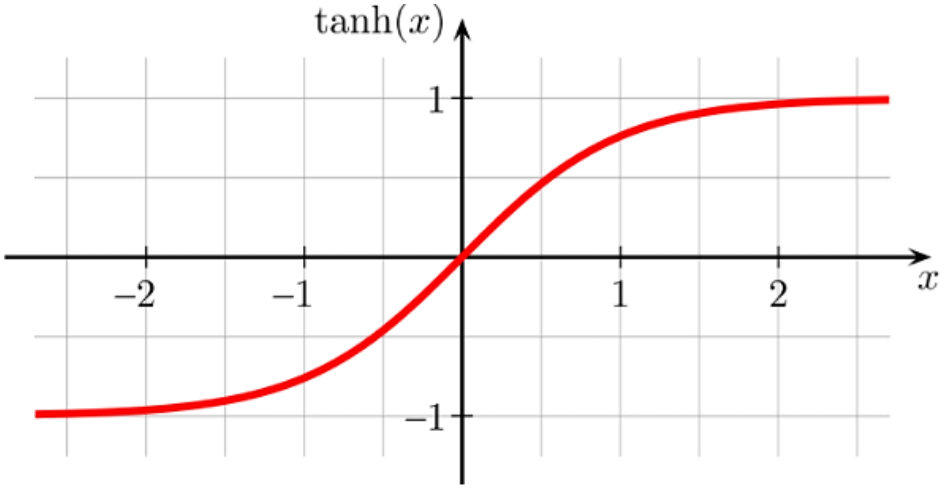


Рисунок 3.6 – Графік гіперболічного тангенсу

Його вихідний діапазон знаходиться між [-1,1]. Вихідний сигнал функції Tanh усереднюється за 0, що усуває другий недолік сигмовидної функції. Але коли введення дуже велике або дуже маленьке, проблема зникнення градієнта все одно залишається.

Функція монотонного збільшення вихідний діапазон знаходиться між [0,1], негативне збільшення наближається до 0, а позитивне збільшення уповільнюється, наближаючись до 1.

Недоліки:

– легше викликати зникнення градієнта. Видно, що коли вхідні дані дуже великі або дуже маленькі, градієнт нейрона майже дорівнює 0, внаслідок чого НМ не сходиться, параметри не будуть оновлюватися, і навчання стане скрутним;

– вихідні дані функції не приймають 0 як середнє значення, що призводить до ненульових вхідних даних для наступного рівня нейронної мережі. Це може призвести до того, що значення вхідної нейронної мережі завжди буде більше 0, тоді параметр оновлюється незалежно від того, наскільки градієнт позитивні.

Багатошаровий персептрон підпадає під категорію алгоритмів прямого зв’язку, оскільки вхідні дані об’єднуються з початковими ваговими

коефіцієнтами у зваженій сумі та піддаються функції активації, як і в одношаровому персептрон. Але відмінність полягає в тому, що кожна лінійна комбінація поширюється на наступний шар.

Кожен шар подає наступний результат своїх обчислень, своє внутрішнє представлення даних. Це проходить весь шлях від прихованих шарів до вихідного шару [53].

Якби алгоритм обчислив лише зважені суми в кожному нейроні, поширив результати на вихідний шар і зупинився на цьому, він не зміг би дізнатися вагові коефіцієнти, які мінімізують функцію вартості. Якби алгоритм обчислював лише одну ітерацію, фактичного навчання не було б реалізовано.

В моделі персептрона існує два алгоритми поширення які впливають на вихідний результат після використання функції активації та на коригування ваг. Оскільки, в будь якому випадку після використання алгоритму прямого поширення(feed forward) ми будемо мати похибку яка в результаті впливає на кінцевий результат який ще називають точністю розпізнавання нейронної мережі. В результаті, ми повинні використовувати алгоритм оберненого поширення який в свою чергу буде регулювати значення ваг під час навчання мережі. В наслідку, якщо мережа навчається 10 ітерацій то значення ваг може бути змінено 10 разів.

У оберненому поширенні механізм навчання, дозволяє багатошаровому персептрону ітеративно регулювати ваги в мережі з метою мінімізації функції вартості.

Існує одна жорстка вимога, щоб оберненне поширення працювало належним чином. Функція, яка поєднує вхідні дані та ваги в нейроні, наприклад, зважену суму, і порогову функцію, наприклад ReLU, повинна бути диференційованою. Ці функції повинні мати обмежену похідну, оскільки ГС, як правило, є функцією оптимізації, яка використовується в багатошаровому перцептроні (3.4).

$$Weight_{n+1} = weight_n + (input \cdot delta \cdot learning_{rate}). \quad (3.4)$$

Схема на рисунку 3.7 показує всі кроки навчання та коригування ваг нейронної мережі. Метою цієї діаграми є перегляд попередньої інформації про роботу персептронів і нейронів, які тренуються.

Першим кроком є випадкова ініціалізація ваг. Потім, використовуючи ці вагові коефіцієнти, ми можемо обчислити результати, які, іншими словами, визначають відповідь нейронної мережі, або ще кажуть передбачення чи точність. Далі цей розрахунок використовує функцію суми та функцію активації нейрона, після чого ми маємо значення прогнозу.

В підсумку нам потрібно порівняти прогнозовані результати з очікуваними результатами набору даних, щоб обчислити помилку. Для цього ми можемо використовувати найпростішу формулу, яка є відніманням між цими двома значеннями.

Після обчислення похибки ми зробимо обчислення, щоб знайти нові ваги, а потім оновити значення ваг (3.4). Ми зосередимося на тому, як знайти ці нові ваги, і рівняння буде трохи відрізнятися від рівняння для одношарового персептрон. У результаті виникає важливе питання, чи досягнуто кількість разів або в перекладі з англійської епоха. Наприклад, при досягненні 100 епох виконання завершується відповідно до кінцевої частини діаграми. В іншому випадку ми йдемо за схемою іншим шляхом і запускаємо весь процес знову. Якщо кількість епох дорівнює 100, то НМ буде виконувати всі ці кроки 100 разів.

Також ми багато говорили про помилку, оскільки спостерігаємо лише найпростіше рівняння для помилки. Проте є два додаткові терміни, які зазвичай використовуються.

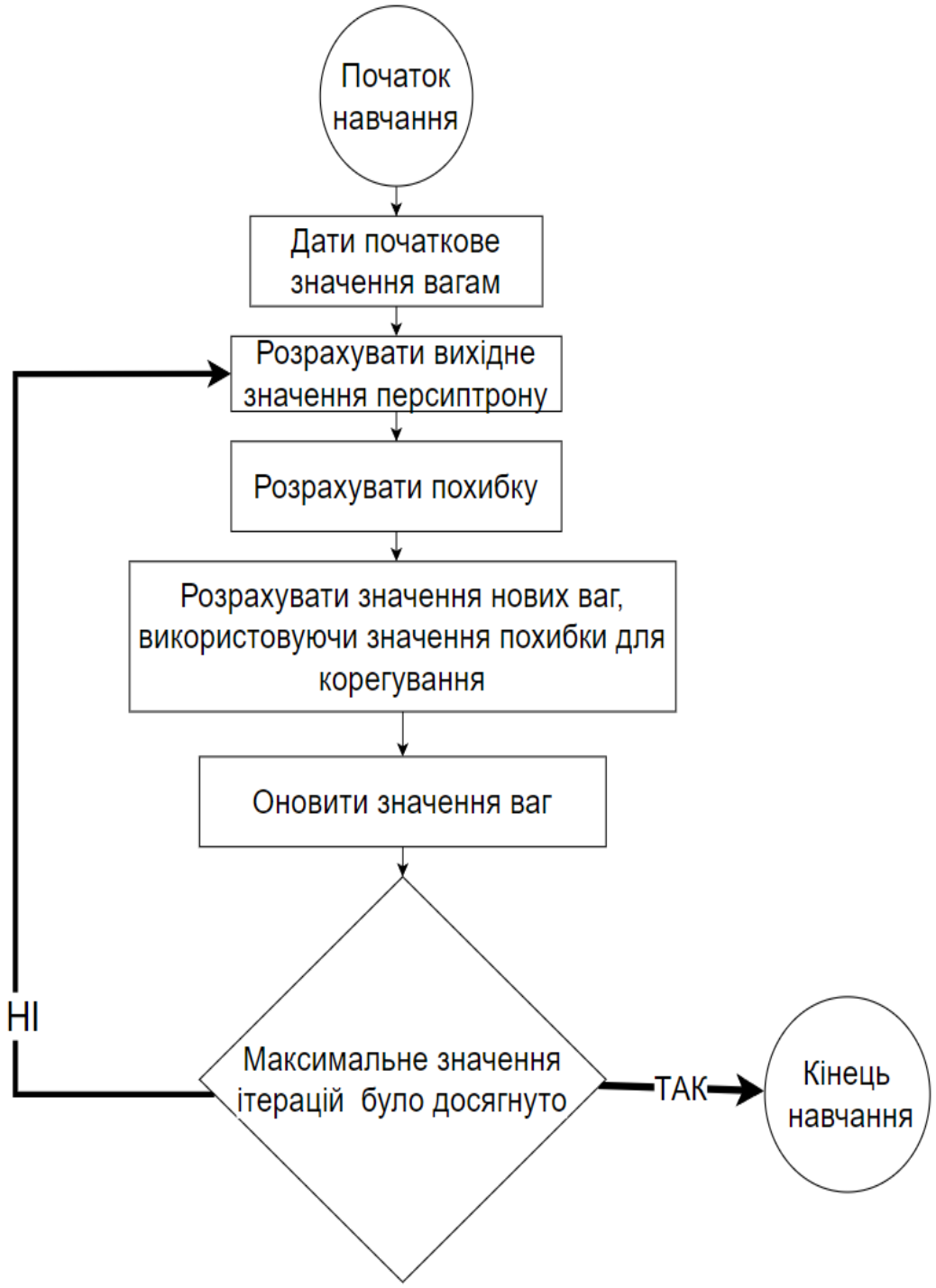


Рисунок 3.7 – Схема навчання нейронної мережі та корегування ваг в незалежності від напрямку поширення

$$Error = ExpectedOutput - result \tag{3.5}$$

Розрахунки для оновлення ваг у багатошаровій нейронній мережі дещо складніші в порівнянні з одношаровим персептроном. Тому потрібно

ввести поняття градієнтного спуску та що він робить.

ГС є одним з найбільш використовуваних методів регулювання ваг у згорткових нейронних мережах. Це алгоритм оптимізації, який використовується для мінімізації певної функції шляхом ітераційного переміщення в напрямку найбільш крутого спуску, що визначається від'ємним значенням градієнта. У машинному навчанні ми використовуємо ГС для оновлення параметрів нашої моделі. Параметри відносяться до коефіцієнтів у лінійній регресії та ваг у нейронних мережах.

Функції втрати або ще можна сказати вартості говорить нам, «наскільки добре» наша модель робить прогнози для заданого набору параметрів. Функція вартості має власну криву і власні градієнти. Нахил цієї кривої говорить нам, як оновити наші параметри, щоб зробити модель більш точною.

Розрахунки для оновлення ваг у багатошаровій нейронній мережі дещо складніші в порівнянні з одношаровим персептроном.

ГС є одним з найбільш використовуваних методів регулювання ваг у згорткових нейронних мережах.

Розрахунок похідної на рис. 3.8, де ми припускали, що ваги ініціалізуються у верхній частині діаграми градієнтного спуску, де є точка А. Тоді чим вище точка, тим більше помилка, тим нижча точність нейронної мережі. Метою нейронного навчання є досягнення точки В у нижній частині цієї діаграми, де значення помилки невелике.

Відповідно до рис. 3.9 ми можемо розглядати всю цю частину рисунку, як область значень яка називається градієнтним спуском, тому що вона схожа на гору. Кожен раз, коли ми регулюємо ваги, ми спускаємося з гори або градієнта, поки не досягнемо дна. З кожним оновленням ваг ми спускаємося з гори. Це вирішальний момент за кількістю епох. Якщо це 100 епох, ми виконуємо цей процес спуску з гори 100 разів.

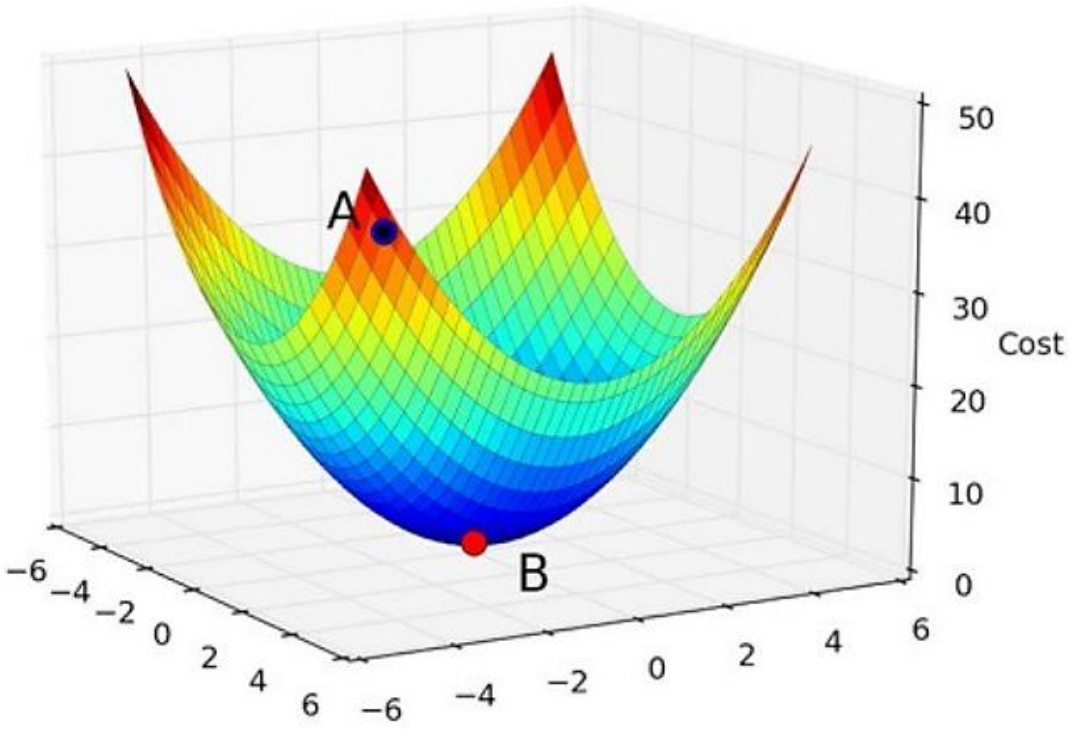


Рисунок 3.8 – Тривимірна діаграма градієнтного спуску [54]

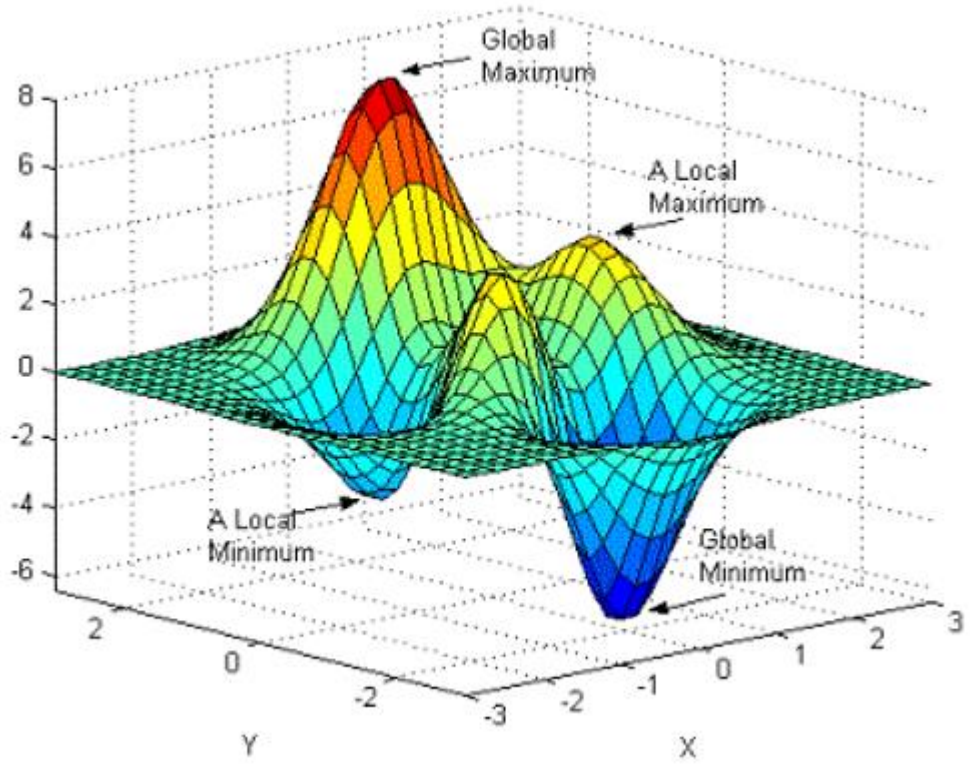


Рисунок 3.9 – Тривимірна діаграма градієнтного спуску з локальним та глобальним мінімумом та максимом [55]

Це ще одне зображення, дуже схоже на попереднє. У кожному ми бачимо помилку вгорі. Наша мета – досягти глобального мінімуму або найменшої помилки через усі локальні мінімуми чи максимуми. Похибка на глобальному мінімумі менша, ніж на інших локальних мінімумах [56].

Якщо ми візьмемо звичайну чашу, коли ми досягнемо дна чаші, ми досягнемо глобального мінімуму, і це і є ідею градієнтного спуску. Розмір цих кроків називається швидкістю навчання. Завдяки високій швидкості навчання ми можемо охопити більше землі на кожному кроці, але ми ризикуємо перевищити найнижчу точку, оскільки схил пагорба постійно змінюється. З дуже низькою швидкістю навчання ми можемо впевнено рухатися в напрямку негативного градієнта, оскільки ми так часто перераховуємо його. Низька швидкість навчання точніша, але обчислення градієнта займає багато часу, тому нам знадобиться дуже багато часу, щоб дістатися до суті. Тому в популярних фреймворках таких як: Tensorflow і Pytorch швидкість або рейтинг навчання прийнято зменшувати на десять відсотків [57].

За формулою (3.6) знаходження глобального мінімуму. Мета – отримати найменшу можливу помилку. Ось чому у формулі є min, а буква C - це абревіатура функції вартості, як ми бачили раніше, що є іншим терміном для посилання на помилку. Крім того, у круглих дужках ми маємо кілька значень ваг. Мета полягає в тому, щоб знайти найкращий набір ваг, які повертають найменшу можливу помилку

$$min\ C(w1,w2\ \ldots\ wn). \tag{3.6}$$

Щоб знати напрямок, у якому ми повинні рухатися, обчислюємо часткову похідну для переміщення в напрямок градієнта. Часткова похідна буде відповідати, чи слід збільшити чи зменшити значення ваг [58].

На осі X цього рис 3.10 графіка ми маємо лише одну вагу, яка представлена w, а на осі Y маємо значення помилки. Важливо відзначити,

що навіть на глобальному мінімумі помилка не дорівнює нулю, як ми бачимо на осі Y. Коли вага дорівнює 3, ми маємо найменшу помилку, а коли вага дорівнює 1, ми маємо найбільшу можливу помилку. Підсумовуючи, мета обчислення похідної полягає в тому, щоб направити алгоритм у правильне місце який буде обчислювати нахил кривої за допомогою часткових похідних [59].

Типи нейронних мереж:

– згорткові нейронні мережі можна застосовувати для класифікації зображень або обробки природної мови;

– автокодери, які можна використовувати для стиснення зображень або виявлення шахрайства;

– рекурентні нейронні мережі можна використовувати для часових рядів. Наприклад: спрогнозувати ціни на фондових ринках або обробити природну мову;

– генеративні змагальні мережі можна використовувати для створення нових зображень.

Усі ці типи нейронних мереж використовують точно ту саму основу, що описана раніше.

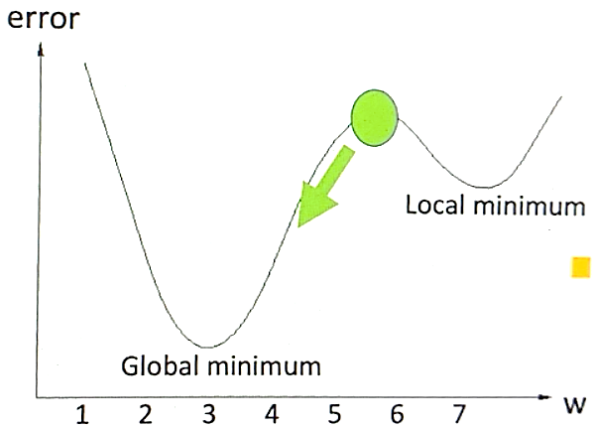


Рисунок 3.10 – Графік знаходження глобального мінімуму для розрахунку нахилу кривої

Основною областю типом цієї згорткові нейронні мережі є КЗ, і тому

ця модель була обрана для застосування виявлення транспортних технологічних об’єктів і для проведення експериментів, які покажуть різницю між іншими моделями.

3.2 Математична модель та формалізація параметрів багатошарового персиптрону для визначення ваг

Загальна постановка задачі: існує проблема розрахунку значень ваг, які ми можемо корегувати на основі математичної моделі з використанням алгоритмів прямого та оберненого поширення для одношарових та багатошарових персептронів.

Обернене поширення було одним із перших методів, які змогли продемонструвати, що штучні нейронні мережі можуть вивчати хороші внутрішні уявлення, тобто їх приховані шари вивчають нетривіальні особливості. Досліджували багатошарових мереж прямого зв’язку показали що більш ефективним є алгоритм оберненого поширення так як обернене поширення дозволяє застосувати штучні нейронні мережі до набагато ширшого поля проблем, які раніше були недоступними через час та обмеження витрат.

Алгоритм оберненого поширення для багатошарового персиптрону в нашій математичній моделі вимагає трьох речей:

– набір даних, що складається з пар введення-виведення $(\overrightarrow{x_i}, \overrightarrow{y_i})$, де $\overrightarrow{x_i}$ – вхід, $\overrightarrow{y_i}$ – бажаний вихід мережі на вхід $\overrightarrow{x_i}$. Позначимо множину пар вхід-вихід розміром N= $(x + a)^n = \{(\overrightarrow{x_1}, \overrightarrow{y_1}), \ldots, (\overrightarrow{x_N}, \overrightarrow{y_N})\}$;

– при оберненому поширенні параметри, що представляють основний інтерес w_{ij}^k – вага між вузлом j у шарі l_k. Також вузол i – в шарі l_{k-1}, і b_i^k це зміщення для вузла i у шарі l_k. Між вузлами в одному шарі немає зв’язків, і шари повністю пов’язані;

– функція похибки $E(X,\theta) = \pi r^2$яка визначає похибку між бажаним результатом $\overrightarrow{y_i}$і підрахований вихід $\overrightarrow{x_i}$ для набору пар вхід-вихід $(\overrightarrow{x_i}, \overrightarrow{y_i}) \in X$ і конкретне значення усіх параметрів θ.

Навчання нейронної мережі з градієнтним спуском вимагає обчислення градієнта функції помилки $E(X,\theta)$ відносно ваг w_{ij}^k і зміщення b_i^k. Потім, відповідно до швидкості навчання α кожна ітерація градієнтного спуску оновлює ваги та зміщення які з себе представляють θ (3.7):

$$\theta^{t+1} = \theta^t - x = \alpha \frac{\partial E(X,\theta^t)}{\partial \theta}, \quad (3.7)$$

де θ^t позначає параметри нейронної мережі на ітерації t в градієнтному спуску.

Одна з основних проблем у навчанні багатошарових нейронних мереж із прямим зв'язком полягає у вирішенні того, як вивчити хороші внутрішні уявлення, тобто якими мають бути ваги та зміщення для вузлів прихованого шару. На відміну від персептрона, який має правило дельта для апроксимації чітко визначеного цільового виходу, вузли прихованого шару не мають цільового виходу, оскільки вони використовуються як проміжні кроки в обчисленні.

Оскільки вузли прихованого шару не мають цільового виводу, не можна просто визначити функцію помилки, характерну для цього вузла. Натомість, будь-яка функція помилки для цього вузла буде залежати від значень параметрів у попередніх шарах (оскільки попередні рівні визначають вхід для цього вузла) і наступних шарах оскільки вихід цього вузла вплине на обчислення функція помилки $E(X,\theta)$. Таке поєднання параметрів між шарами може зробити математику досить заплутаною в першу чергу в результаті використання правила добутку, обговорюваного нижче), і, якщо його не реалізувати грамотно, може зробити остаточні

обчислення градієнтного спуску повільними. Оберненне поширення вирішує обидві ці проблеми, спрощуючи математику градієнтного спуску, а також полегшуючи його ефективне обчислення.

Далі були використані наступні параметри:

w_{ij}^k – вага між вузлом j у шарі l_k для вхідного вузла i;

b_i^k – зміщення для вузла i у шарі l_k;

a_i^k – сума вхідних параметрів плюс зміщення для вузла i у шарі l_k;

o_i^k – вихід для вузла i у шарі l_k;

r_k – кількість вузлів у шарі l_k;

r_k – функція активації для вузлів прихованого шару;

g_o – функція активації для вузлів вихідного рівня.

Формула для підрахунку помилки в класичному оберненному поширенні є середньоквадратичною помилкою (3.8):

$$E(X,\theta) = \frac{1}{2N}\sum_{i=1}^{N}\left(\overrightarrow{\hat{y}_\text{i}} - y_\text{i}\right)^2, \tag{3.8}$$

де y_i є цільовим значенням для пари введення-виведення $(\overrightarrow{x_\text{i}}, \overrightarrow{y_\text{i}})$ і $\overrightarrow{\hat{y}_\text{i}}$ – обчислений вихід мережі для регулювання ваг.

Також можна використовувати й інші функції помилок, але історичний зв’язок середнього квадрата помилки з оберненним поширенням та його зручні математичні властивості роблять його хорошим вибором для вивчення методу.

Виведення алгоритму обернним поширення є досить простим. Це випливає з використання правила ланцюга та правила добутку в диференціальному обчисленні. Застосування цих правил залежить від диференціації функції активації.

Для решти цього розділу похідна функції $f(x)$ буде позначатися $\acute{f}(x)$, так що похідна сигмовидної функції дорівнює $\acute{\sigma}(x)$. Щоб спростити математику ще більше, зміщення b_i^k для вузла i в шарі k буде включено до ваг як w_{0i}^k з фіксованим виходом $o_j^{k-1} = 1$ для вузла 0 у шарі $k-1$. Таким чином (3.9):

$$w_{0i}^k = b_i^k. \tag{3.9}$$

Щоб переконатися, що це еквівалентно вихідному формулюванню, зверніть увагу на (3.10) [60]:

$$a_i^k = b_i^k + \sum_{j=1}^{r_{k-1}} w_{ji}^k \, o_j^{k-1} = \sum_{j=0}^{r_{k-1}} w_{ji}^k \, o_j^{k-1}, \tag{3.10}$$

де ліва сторона - вихідна формула, а права – нове формулювання.

Використовуючи позначення вище, оберненним поширення намагається мінімізувати таку функцію помилки щодо ваг нейронної мережі (3.11) [61]:

$$E(X,\theta) = \frac{1}{2N}\sum_{i=1}^{N}\left(\overrightarrow{\hat{y}_\text{i}} - y_\text{i}\right)^2. \tag{3.11}$$

Шляхом розрахунку для кожної ваги w_{ij}^k , є значення $\frac{\partial E_d}{\partial w_{ij}^k}$. Оскільки функцію похибки можна розкласти на суму за окремими умовами помилки для кожної окремої пари введення-виведення, похідну можна обчислити щодо кожної пари вхід-вихід окремо, а потім об’єднати в кінці (оскільки похідна суми функції є сумою похідних кожної функції) (3.12) [62]:

$$\frac{\partial E(X,\theta)}{\partial w_{ji}^k} = \frac{1}{N}\sum_{d=1}^{N}\frac{\partial}{\partial w_{ij}^k}\left(\frac{1}{2}(\hat{y}_d - y_d)^2\right) = \frac{1}{N}\sum_{d=1}^{N}\frac{\partial E_d}{\partial w_{ij}^k}. \tag{3.12}$$

Таким чином, для похідної, алгоритм оберненого поширення буде займатися лише однією парою вхід-вихід. Як тільки це буде отримано, загальну форму для всіх пар введення-виведення в X можна створити шляхом комбінування окремих градієнтів. Таким чином, функція помилки, яка розглядається для похідної (3.13):

$$E = \frac{1}{2}(\hat{y}_d - y_d)^2, \tag{3.13}$$

де індекс d в E_d, $\hat{y}_d$ і y_d опущено для спрощення.

Виведення похідної функції для похибки, що застосовується в математичній моделі виходить з алгоритму оберрненого поширення починається із застосування правила ланцюга до часткової похідної функції похибки (3.14):

$$\frac{\partial E}{\partial w_{ij}^k} = \frac{\partial E}{\partial a_j^k}\frac{\partial a_j^k}{\partial w_{ij}^k}, \tag{3.14}$$

де a_j^k – значення функції активації (сума продукту плюс зміщення) вузла j у шарі k перед тим, як він буде переданий нелінійній функції активації (у даному випадку сигмоїдній функції) для генерування результату. Це розкладання часткової похідної в основному говорить про те, що зміна функції помилки через вагу є добутком зміни функції помилки E через значення активації a_j^k через вагу w_{ij}^k .

Перший доданок зазвичай називають похибкою. Вона позначається (3.15):

$$\delta_j^k \equiv \frac{\partial E}{\partial a_j^k}. \tag{3.15}$$

Другий доданок можна обчислити з рівняння для a_j^k вище (3.16):

$$\frac{\partial a_j^k}{\partial w_{ij}^k} = \frac{\partial}{\partial w_{ij}^k}\left(\sum_{l=0}^{r_{k-1}} w_{li}^k \; o_j^{k-1}\right) = o_j^{k-1}. \tag{3.16}$$

Таким чином, часткова похідна функції помилки E щодо ваги w_{li}^k підраховується таким чином (3.17):

$$\frac{\partial E}{\partial w_{ij}^k} = \delta_j^k o_i^{k-1}. \tag{3.17}$$

Таким чином, часткова похідна ваги є добутком члена помилки δ_j^k у вузлі j в шарі k, а вихід o_i^{k-1} вузла i в шарі $k-1$. Це має сенс, оскільки вага w_{ij}^k з'єднує вихід вузла i в шарі $k-1$ з входом вузла j в шарі k.

Важливо зазначити, що всі наведені вище часткові похідні були розраховані без будь-якого врахування певної функції помилки або функції активації. Однак, оскільки значення похибки δ_j^k все ще потрібно обчислити, і вона залежить від функції похибки E, на цьому етапі необхідно ввести конкретні функції для обох. Як згадувалося раніше, класичне оберненне поширення використовує середню квадратичну функцію помилки (яка є квадратичною функцією помилки для однієї пари вхід-вихід) і сигмоподібну функцію активації [60,61].

Розрахунок похибки δ_j^k залежить від значень доданків помилки в наступному шарі. Таким чином, обчислення термінів помилки буде продовжуватися назад від вихідного рівня до вхідного рівня. Саме звідси отримало свою назву оберненне поширення, або оберненне поширення помилок.

Починаючи з останнього шару, оберненне поширення в нашій математичній моделі намагається визначити значення δ_j^m, де m – кінцевий шар. Наприклад, чотиришарова нейронна мережа матиме $\mathrm{m} = 3$ для

останнього шару, m = 2 для передостаннього шару тощо. Вираз функції помилки E через значення a_1^m дає (3.18):

$$E = \frac{1}{2}(\hat{y} - y)^2 = \frac{1}{2}(g_0(a_1^m) - y)^2, \quad (3.18)$$

де $g_0(x)$ – функція активації вихідного шару.

Таким чином, застосування часткової похідної та використання правила ланцюга дає (3.19):

$$\delta_j^m = (g_0(a_1^m) - y)g_0(a_1^m) = (\hat{y} - y)g_0(a_1^m). \quad (3.19)$$

Складаючи все разом, часткова похідна функції помилки E щодо ваги в останьому шарі w_{i1}^m формула є наступною (3.20) [61,62]:

$$\frac{\partial E}{\partial w_{i1}^m} = \delta_1^m - o_i^{m-1} = (\hat{y} - y)\, g_0(a_1^m)o_i^{m-1}. \quad (3.20)$$

Виникає питання, як обчислити часткові похідні шарів, відмінних від вихідного шару. Таким чином, ланцюгове правило для багатовимірних функцій знову приходить на допомогу. Зверніть увагу на наступне рівняння для члена помилки δ_j^k у шарі $1 \le k \le m$ (3.21) [62]:

$$\delta_j^k = \frac{\partial E}{\partial a_j^k} = \sum_{l=1}^{r^{k+1}} \frac{\partial E}{\partial a_l^{k+1}} \frac{\partial a_l^{k+1}}{\partial a_j^k}, \quad (3.21)$$

де l коливається від 1 до r^{k+1} кількості вузлів у наступному шарі.

Оскільки вхідне зміщення a_{j0}^k дорівнює w_{0j}^{k+1} фіксование то його значення не залежить від вихідних даних попередніх шарів, тобно, l не приймає значення 0.

Підставлення значення помилки δ_l^{k+1} дає наступне рівняння (3.22):

$$\delta_j^k = \sum_{l=1}^{r^{k+1}} \delta_l^{k+1} \frac{\partial a_l^{k+1}}{\partial a_j^k}. \quad (3.22)$$

Пригадуючи визначення a_j^{k+1} (3.23):

$$a_j^{k+1} = \sum_{j=1}^{r^k} w_{jl}^{k+1} g(a_j^k), \quad (3.23)$$

де g(x) - функція активації для прихованих шарів (3.24) [63]:

$$\frac{\partial a_l^{k+1}}{\partial a_j^k} = w_{jl}^{k+1} g(a_j^k). \quad (3.24)$$

Підставляючи це в наведене вище рівняння, виходить остаточне рівняння для члена помилки δ_j^k у прихованих шарах, що називається формулою оберненного поширення (3.25) [61-62]:

$$\delta_j^k = \sum_{l=1}^{r^{k+1}} \delta_l^{k+1} w_{jl}^{k+1} \acute{g}(a_j^k) = \acute{g}(a_j^k) \sum_{l=1}^{r^{k+1}} \delta_{jl}^{k+1} w_l^{k+1}. \quad (3.25)$$

Складаючи все разом, часткова похідна функції помилки E щодо ваги в прихованих шарах w_{ij}^k для $1 \leq k \leq m$ рівняння (3.26):

$$\frac{\partial E}{\partial w_{ij}^k} = \delta_j^k o_i^{k-1} = \acute{g}(a_j^k) o_i^{k-1}. \quad (3.26)$$

Виходячи з наведених вище доказів можна сказати що наша математична модель залежить від наступних формул:

– для часткових похідних (3.27):

$$\frac{\partial E}{\partial w_{ij}^k} = \delta_j^k o_i^{k-1}\ ; \quad (3.27)$$

– для значенння похибки останнього шару (3.28):

$$\delta_1^m = \acute{g}_o(a_1^m)(\hat{y}_d - y_d); \quad (3.28)$$

– для значень похибки прихованих шарів (3.29):

$$\delta_j^k = \acute{g}(a_j^k)\sum_{l=1}^{r^{k+1}} w_{jl}^{k+1}\delta_l^{k+1}; \quad (3.29)$$

– для об'єднання часткових похідних для кожної пари вхід-вихід (3.30) [60,61]:

$$\frac{\partial E(X,\theta)}{\partial w_{ji}^k} = \frac{1}{N}\sum_{d=1}^{N}\frac{\partial}{\partial w_{ij}^k}\left(\frac{1}{2}(\hat{y}_d - y_d)^2\right) = \frac{1}{N}\sum_{d=1}^{N}\frac{\partial E_d}{\partial w_{ij}^k}; \quad (3.30)$$

– для регулювання ваг (3.31):

$$\Delta w_{ij}^k = -a\frac{\partial E(X,\theta)}{\partial w_{ji}^k}. \quad (3.31)$$

Результати розрахунків на основі метода оберненого поширення наведено у табл. 3.1.

Таблиця 3.1 – Результати розрахунків на основі метода оберненого поширення

w_{ij}^k	b_i^k	a_i^k	o_i^k	δ_j^k
0,05	0,11541	22	7,3	0,90

0,11	0,11541	22	2,9	0,80
0,15	0,11541	22	1,7	0,64
0,21	0,11541	22	1	0,57
0,25	0,11541	22	0,78	0,50
0,31	0,11541	22	0,50	0,40
0,35	0,11541	22	0,34	0,30
0,41	0,11541	22	0,19	0,20
0,45	0,11541	22	0,11	0,10
0,51	0,11541	22	0,18	0,20
0,55	0,11541	22	0,22	0,30
0,61	0,11541	22	0,27	0,40
0,65	0,11541	22	0,30	0,50
0,71	0,11541	22	0,35	0,60

3.3 Висновки до розділу 3

Розроблено математичну модель для розв’язання задачі визначення помилки яка впливає на точність багатошарового персиптрону який використовується в системі розпізнавання транспортних технологічних об’єктів. Це дозволило налаштувати значення ваг в залежності від функції активації та значення зміщення, що дає можливість використовувати більш точний алгоритм розпізнавання технологічних об’єктів які можуть мати набагато більше вхідних параметрів ніж було використано в математичній моделі в залежності від зміщення та функцій активації які використовуються для активування нейрону.

На практиці отримані результати дозволять визначити найкращі значення ваг та зміщень в багатошаровому персиптроні для отримання більш точних значень розпізнавання транспортних технологічних об’єктів.

4 РОЗРОБКА АЛГОРИТМІЧНО-ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ АВТОМАТИЗОВАНОЇ СИСТЕМИ РОЗПІЗНАВАННЯ ТРАНСПОРТНИХ ТЕХНОЛОГІЧНИХ ОБ’ЄКТІВ

4.1 Вибір інструментів для програмної реалізації

Провівши аналіз сучасного стану нейронних мереж та їх застосування, проведено вибір інструментів для реалізації автоматизованого методу реалізації розпізнавання технологічних об’єктів.

На основі проектування автоматизованого методу розпізнавання та визначених вимог, були поставлені такі загальні практичні завдання:

– метод має коректно обробляти дані;

– серіалізовані дані мають зберігатися та мати змогу бути обробленими у певних форматах.

На основі таких завдань, з міркувань та знання про структуру системи, можемо зробити висновок, що технології, які будуть використовуватися для розробки, мають бути максимально зручними у оперуванні даними різних типів та форматів, при цьому забезпечуючи ефективну серіалізацію даних.

Також, пам’ятаючи про різноплановість використовуваних компонентів системи, обрана мова програмування повинна мати необхідні бібліотеки для реалізації поставлених цілей.

Для розробки програми реалізації функціонування НМ для розпізнавання об’єктів обрані наступні засоби:

– C# – об'єктно-орієнтована мова програмування;

– Unity – середовище розробки програм;

– OpenCV – бібліотека комп’ютерного бачення;

– Yolo – сімейство моделей виявлення об'єктів у складному масштабі.

Одна з найкращих мов програмування для створення програмного засобу розпізнавання, глибоке знання C# є важливим елементом кожного

програміста в індустрії нейронних мереж. Багато розробників часто віддають перевагу вивченню C# перед іншими мовами програмування завдяки високому рівню зручності. Мова, яку розробляє і популяризує Microsoft, підтримує сучасні версії Unity3D, один з найкращих двигунів, який зараз використовується в галузі. C# надає розробникам можливість створювати програми будь-якого типу для будь-якої архітектури без зайвих клопотів. Мова також набагато легша для вивчення, ніж C++ [64].

C# володіє набором засобів, які допомагають у реалізації програми. Вона добре підходить для розпізнавання та обробки об’єктів.

Бібліотека OpenCV – це бібліотека КЗ, це база функцій, алгоритмів обробки зображень з відкритим кодом.

Бібліотека OpenCV включає в себе великий набір, як класичних, так і сучасних алгоритмів комп’ютерного зору та машинного навчання.

Розглянемо схему роботи з бібліотекою OpenCV. На початку роботи програми, захоплюється зображення – використовують модуль Highgui. Наступний крок це попередня обробка: нормалізація освітлення позбавлення шуму, вирівнювання, контрасту.

Для цього використовують модуль imgproc, а потім модуль features2d – для виділення особливостей об’єкта.

Наступний крок детектування об’єктів, виділення частин, сегментація до роботи приступають модулі imgproc, objdetect.

Для виконання задач по знаходженню об’єкта, аналізу його структури у роботу включаються модулі video, calib3d, contrit, videostab, ml.

Модуль ml розпізнає і приймає рішення чи знайдений об’єкт той, який потрібен.

Основна мета OpenCV – це зображення загальної інфраструктури для прискорення використання машинного сприйняття та для програм комп’ютерного зору.

Використовують для класифікації зображень, відстеження рухомих об’єктів, розпізнавання краєвидів.

Головна особливість YoLo полягає в тому, що більшість систем застосовують CNN кілька разів із різними регіонами зображення. YoLo використовує її лише один раз для всього зображення. Мережа ділить зображення на сітку і передбачає розташування шуканого об’єкта для кожної ділянки.

Мережа сканує все зображення відразу і враховує контекст при знаходженні і розпізнаванні об’єкта. Це є перевага даного підходу.

Архітектура YoLo складається з вхідного шару, шару згортки, шару об’єднання, зв’язаного шару та вихідного шару. Шар згортки використовується для виявлення ознак зображення, пов’язний шар використовують для з’ясування положення зображення та значень ймовірності цільових категорій.

Шар об’єднання відповідає за зменшення пікселів фрагмента.

Мережа має 24 згорткових шарів, 2 повністю зв’язних шари. Завдяки чергуванню згорткових шарів 1×1 відбувається зменшення ознак від попередніх шарів. Попередньо тренують згорткові шари для завдання класифікації ImageNet з половиною роздільної здатності, а потім подвоюється роздільна здатність для самого процесу виявлення.

YoLo в 1000 раз швидше, ніж R-CNN і в 100 раз швидше ніж FasterCNN. Вдосконалена версія YoLo є YoLov4. Вона складається з 106 згорткових шарів. Особливість YoLov4 є те, що на виході є 3 шари, кожен шар розрахований на виявлення об’єктів різного розміру.

Спрощена версія YoLov4 є YoLov4 – tiny на виході має 2 шари. YoLov4 – tiny час призначений для невеликих об’ємів даних і гірше приймає дрібні об’єкти. Об’єм мережі займають невеликий обсяг пам’яті, через спрощену будову, вона зменшує час опрацювання кадрів.

Тому її краще використовувати в мобільному пристрої. YoLov4 має 3 вихідні шари, орієнтована на меншу кількість об’єктів. Має перевагу у швидкості та розмірі пам’яті.

4.2 Формування вимог для програмної реалізації

У процесі розробки програмного формування вимог є першою діяльністю програмної інженерії. Ця фаза є фазою, де домінують користувачі, і вона перетворює ідеї чи погляди в документ вимог. Визначення та документування вимог користувача в стислий і недвозначний спосіб є першим важливим кроком для досягнення високоякісного продукту.

Вимоги до програмного забезпечення є описом можливостей та функціональних можливостей цільової системи. Вимоги передають очікування користувачів від програмного продукту. Вимоги можуть бути очевидними чи прихованими, відомими чи невідомими, очікуваними чи неочікуваними з точки зору клієнта.

У процесі розробки програмного забезпечення фаза вимог є першою діяльністю програмної інженерії. Ця фаза є фазою, де домінують користувачі, і вона перетворює ідеї чи погляди в документ вимог. Зауважте, що визначення та документування вимог користувача в стислий і недвозначний спосіб є першим важливим кроком для досягнення високоякісного продукту.

В ході проведення роботи, після вибору засобів реалізації було проведено формування вимог до розроблюваного ПЗ.

Проаналізувавши різні моделі нейронних мереж та обравши цільову нейронну мережу у якості нейронної мережі було обрано YOLOv4, яка використовує згорткову нейронну мережу для виявлення транспортних технологічних об’єктів. На базі данної мережі саме і буде виконано розробку пз.

Для розробленого програмного засобу можна висунути наступні вимоги:

– зрозумілий інтерфейс;

– відображення інформації про об'єкт реальному часті;

– підрахунок кількості об’єктів;

– визначення точності розпізнавання;

– можливість одночасного розпізнавання декількох об’єктів;

– можливість одночасного розпізнавання різних об’єктів;

– адаптивність під різні дисплеї.

4.3 Проектування структурної схеми програмного засобу

Створили структурну схему програмного засобу, що зображена на рис. 4.1.

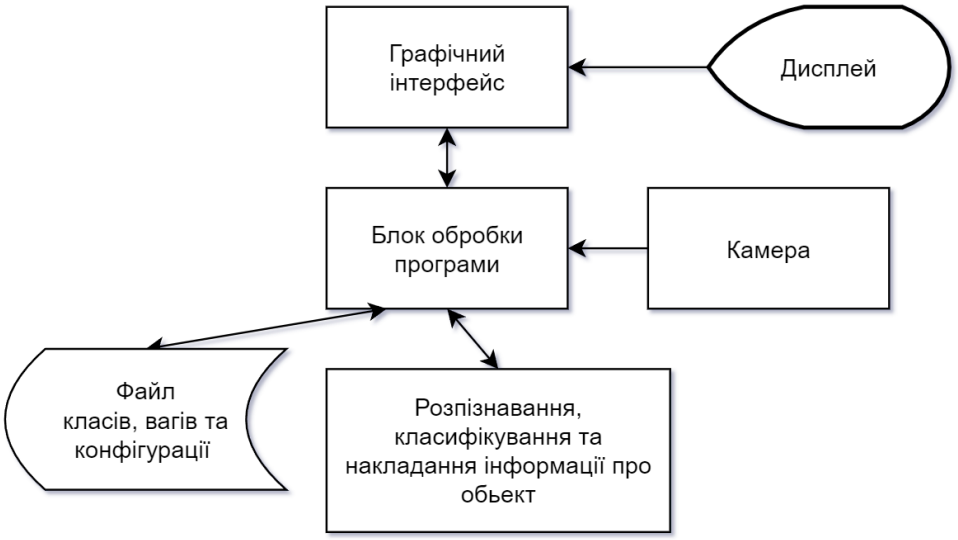


Рисунок 4.1 – Структурної модель розробленої програми

4.4 Проектування об'єктної моделі програмного засобу

Об’єктна модель – це візуальне уявлення об’єктів, дій та пов’язаних атрибутів системи. Об’єктну модель можна використовувати разом із системою проектування, щоб створити узгоджений досвід для конструкцій вищого рівня системи.

Щоб розробити доданок який буде розпізнавати, класифікувати та підраховувати транспортні технологічні об’єкти потрібно спланувати об'єктну модель.

На рис 4.2 представлено об'єктну модель програмного засобу.

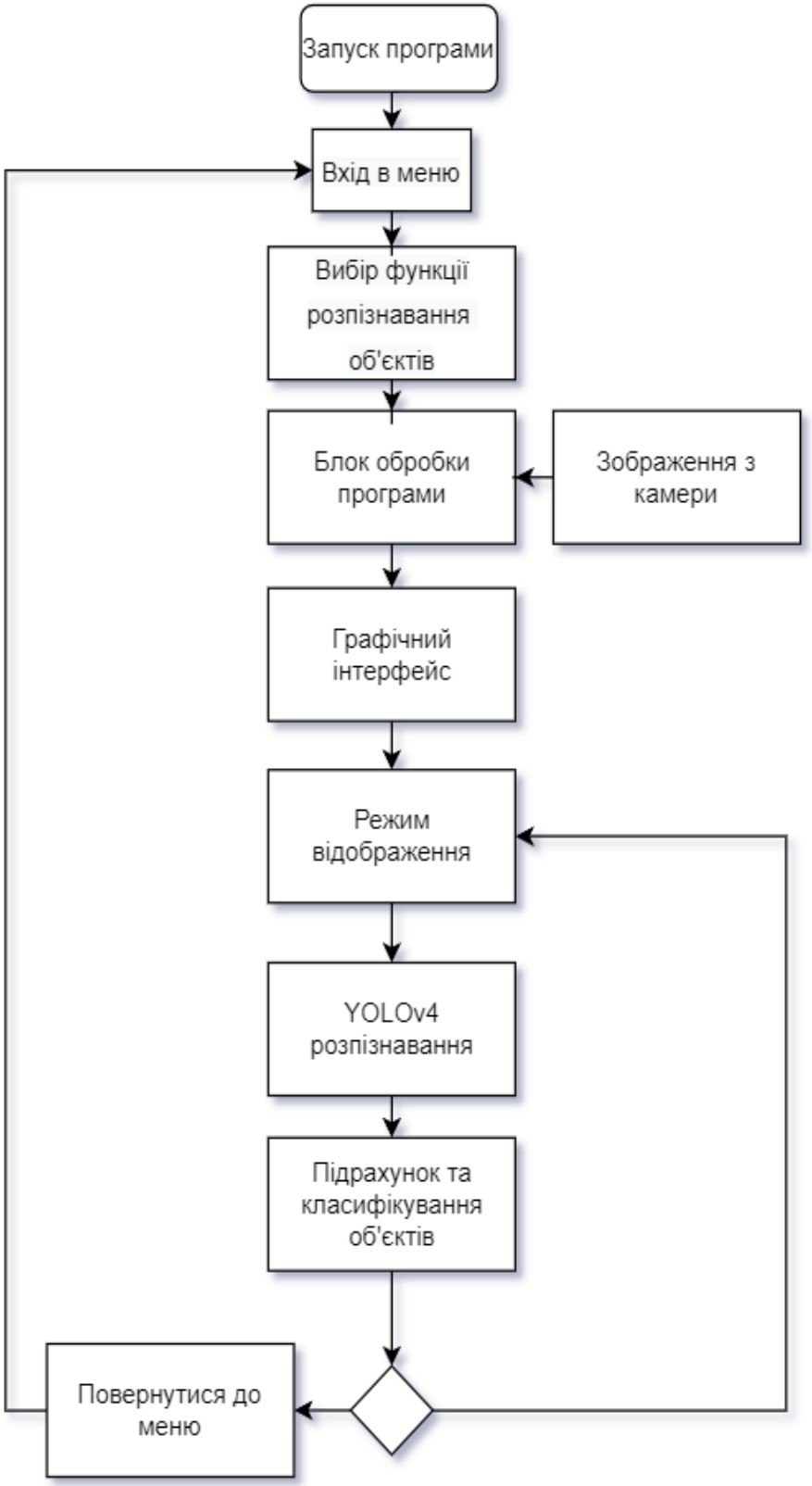


Рисунок 4.2 – Об'єктна модель програмного засобу

Об'єктну модель програмного засобу включає такі компоненти, як блок обробки програми, графічного інтерфейсу для взаємодії користувача з додатком, функції розпізнавання об’єктів, блоку обробки програми, , розпізнавання за допомогою YOLOv4, підрахунку та класифікування об'єктів. Вихід з програми виконується за допомогою кнопки повернутися до меню.

Розроблена модель дає можливість докладніше оцінити вимоги до ПЗ під час розробки застосунку.

4.5 Розробка інтерфейсу

Інтерфейс, який керується меню – це просто легший спосіб навігації між пристроями та програмами, з якими ми щодня взаємодіємо. Він використовує серію екранів або «меню», які дозволяють користувачам вибирати, що робити далі. Інтерфейс, керований меню, може використовувати формат списку або графіку, з одним вибором, що веде до наступного екрана меню, доки користувач не досягне бажаного результату.

Інтерфейсам які керуються за допомогою меню, віддають перевагу через їхню простоту та зручні властивості. Подібно до книг «Вибери власні пригоди», інтерфейси, керовані меню, дозволяють вибрати один крок, який веде до іншого, доки не завершите всі кроки та не отримаєте те, що потрібно. Це дозволяє виконувати такі завдання, як отримання готівки в банкоматі, отримання інформації з кіоску або прибуття до відповідного розділу властивостей смартфона для підключення до Wi-Fi кав’ярні.

Інтерфейси, керовані меню, відрізняються від інтерфейсу командного рядка, який використовує підказки, у які користувач повинен ввести відповідь або команду. Потім користувачі повинні чекати, поки система відповість на введену команду, і отримати запит на введення наступної команди. Це схоже на розмову з миттєвими повідомленнями з комп’ютером! Цей тип інтерфейсу особливо поширений серед програмістів, які

використовують «Командний рядок» комп'ютера на базі Windows або програму «Термінал» на Mac, щоб сказати комп'ютеру, що робити.

Графічний інтерфейс сенсорного екрана дуже схожий на звичайний графічний інтерфейс, за винятком того, що ви використовуєте пальці або стилус для вибору піктограм і виконання завдань, а не миші або трекпада. Графічні інтерфейси сенсорного екрану зазвичай зустрічаються на планшетах, смартфонах та медичних пристроях. Сенсорний графічний інтерфейс має ті ж переваги та недоліки, що й стандартний графічний інтерфейс користувача, але також пропонує більш інтимний метод взаємодії. Відсутність периферійних пристроїв робить сенсорний графічний інтерфейс дуже зручним. В результаті Unity дозволяє імітувати інтерфейс (рис. 4.3).

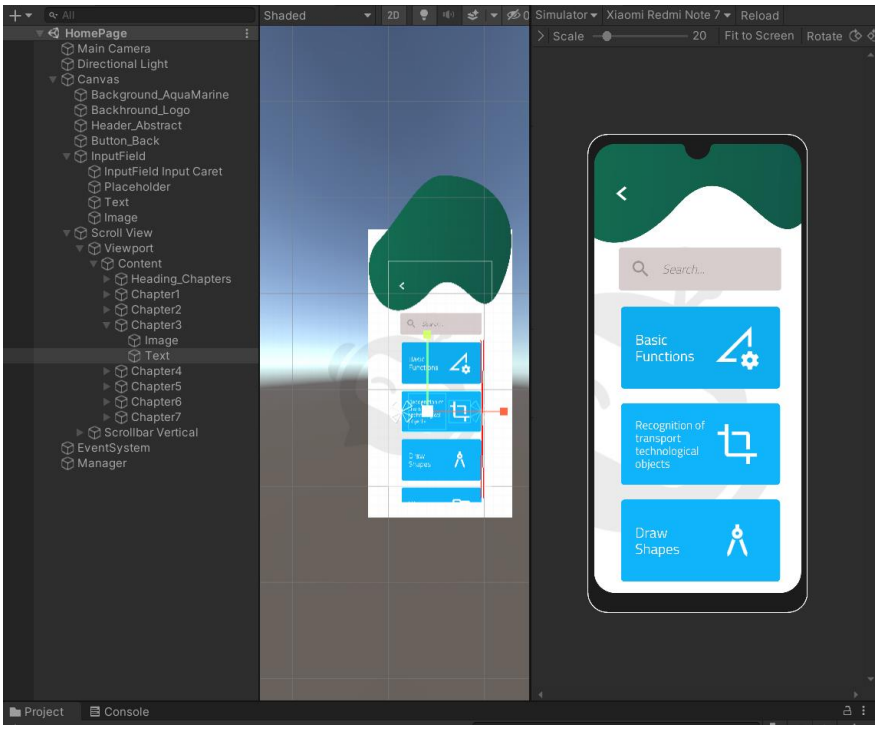


Рисунок 4.3 – Інтерфейс Unity

Незважаючи на те, що вони займають значний простір у пам'яті в порівнянні з іншими інтерфейсами інтерфейсу користувача, це є другорядною проблемою.

З чотирьох типів користувальницького інтерфейсу найбільш поширеним є графічний інтерфейс користувача, за яким слідує варіант із сенсорним екраном. Незважаючи на альтернативні технології, які вже існують і продовжують з'являтися, GUI залишається кращим стандартом. Багато в чому це пов'язано з простотою і зручністю використання.

Більшості кінцевих користувачів легше зрозуміти графічні інтерфейси користувача, оскільки значки та меню, як правило, зрозумілі, а графічний інтерфейс користувача не вимагає від користувача запам'ятовувати чи вводити складні команди (рис. 4.4).

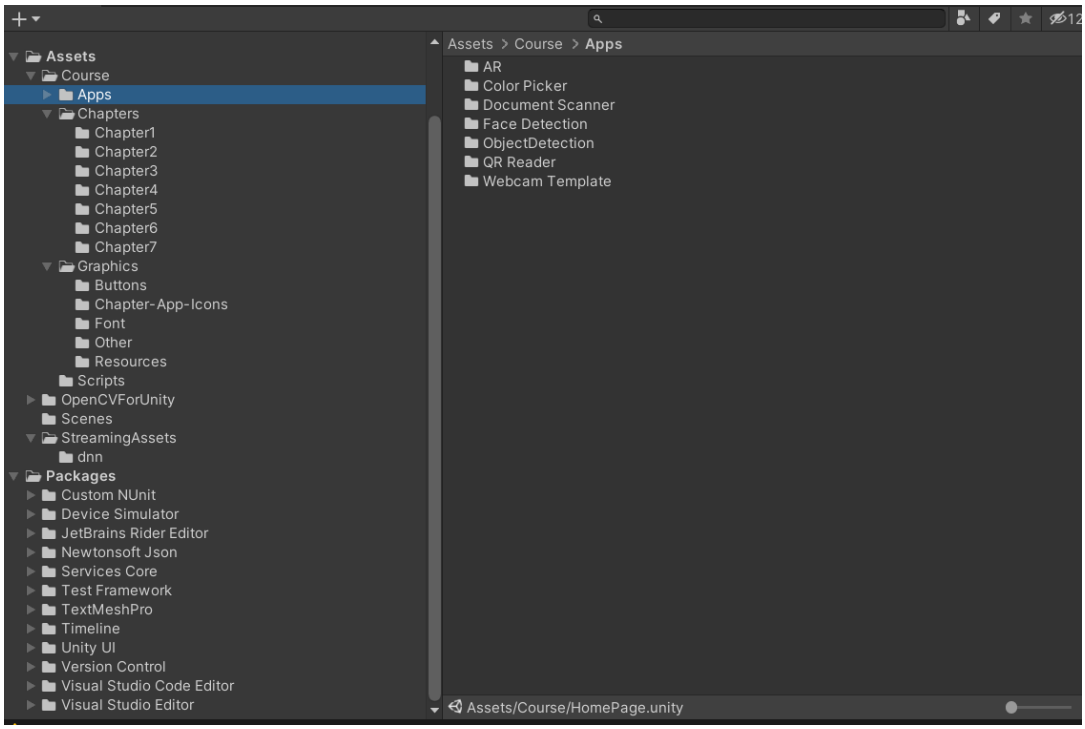


Рисунок 4.4 – Структура проекту в Unity

Отже, ми створили початкову сторінку , де у нас буде кнопка «Пуск» і буде логотип, і де ми також матимете можливість перевірити, про що йдеться у застосунку. Отже, ми збираємося перейти до нашого проекту, і ми вже відкрили стартову сторінку, можна побачити, що на ній написано початок сторінки.

Були створені елемент UI, а всередині елемента UI ми збираємося

створити полотно. Таким чином, полотно або канвас – це область, де обмежена наша програма, тому все, що ми помістимо всередину полотна, з'явиться в нашому мобільному застосунку, тому ми повинні переконатися, що розмір цього є точним, тому перш за все ми збираємося перейти до сканера canvas. Існуючий симулятор дозволяє переглянути графічний інтерфейс застосунку на різних смартфонах на рис. 4.5-4.7

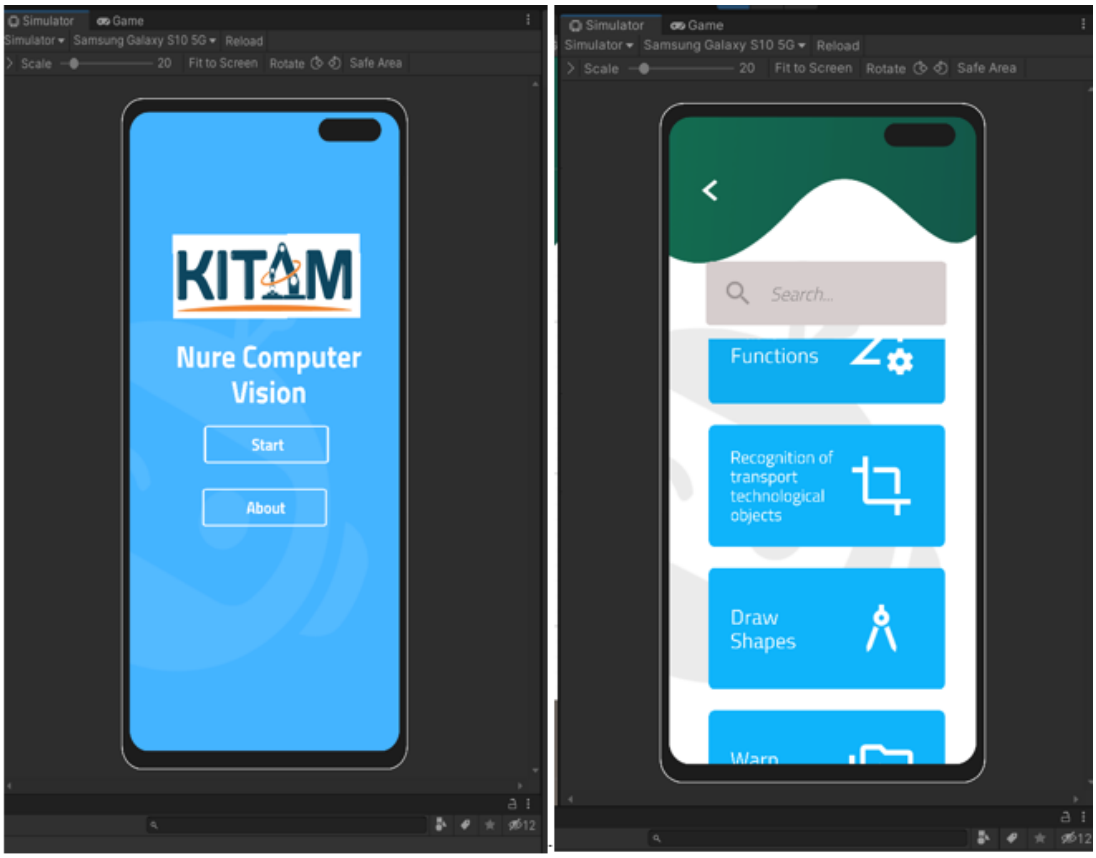


Рисунок 4.5 – Симуляція на Samsug galaxy S10 5G

Симулятор пристрою – це функція редактора Unity, яка імітує, як певні аспекти програми виглядатимуть і ведуть себе на мобільному пристрої.

Симулятор пристрою складається з:

– подання симулятора пристрою, яке дає змогу переглядати програму на змодельованому мобільному пристрої;

– змодельовані класи, які дозволяють тестувати код, який реагує на

поведінку, специфічну для пристрою;

– визначення пристроїв, які описують пристрій для моделювання.

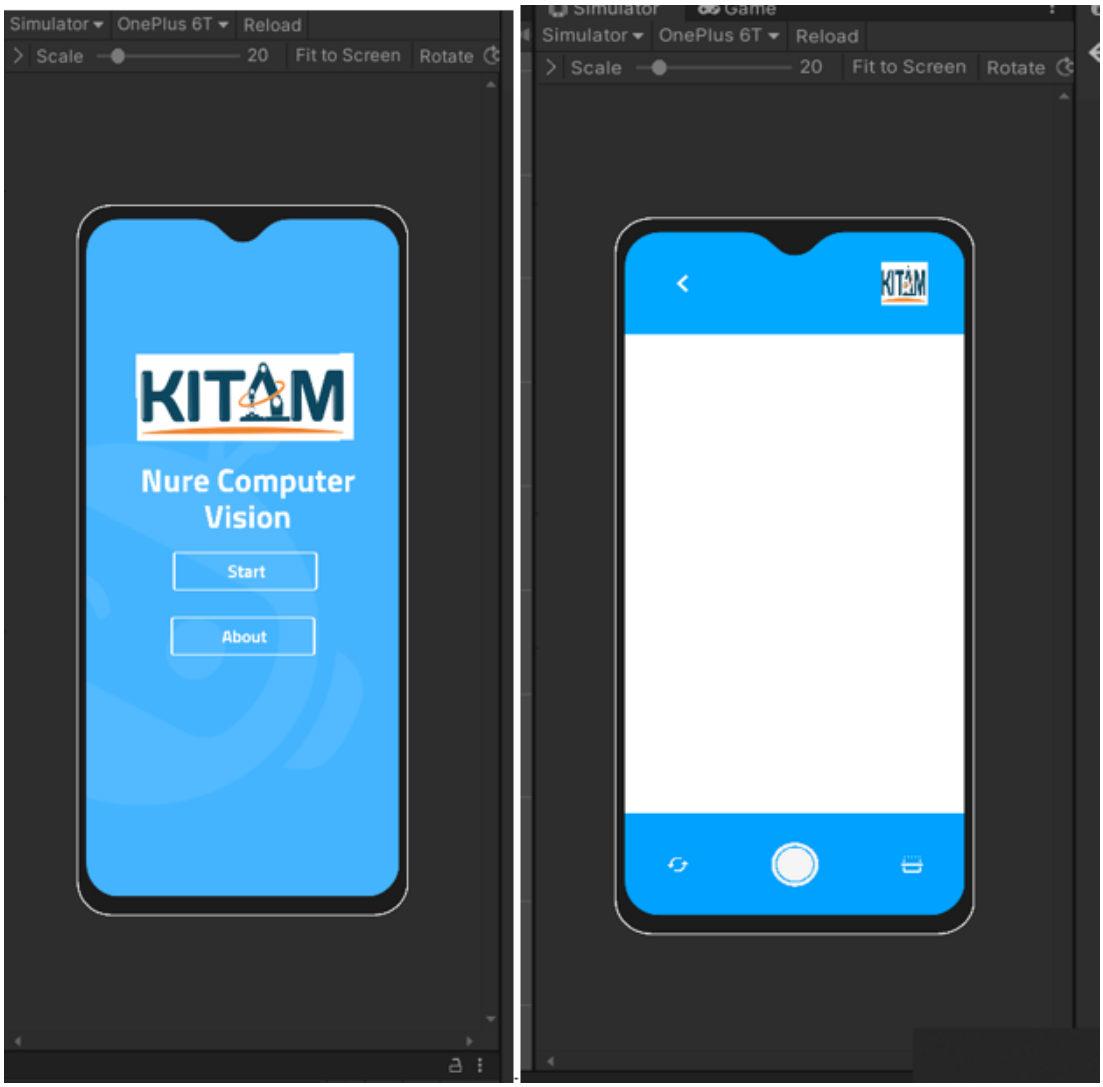


Рисунок 4.6 – Симуляція на One Plus 6T

Клацання курсором миші на екрані модельованого пристрою створить події дотику в диспетчері вводу або в новій системі введення або в обох, залежно від того, яка з них активна. Введення моделюється лише тоді, коли редактор перебуває в режимі відтворення.

Симулятор пристрою не підтримує мультитач, імітується лише один дотик пальцем.

Симулятор пристрою призначений насамперед для перегляду макета розроблюваного застосунку та тестування основних взаємодій. Він не дає

точного уявлення про те, як програма працюватиме на пристрої.

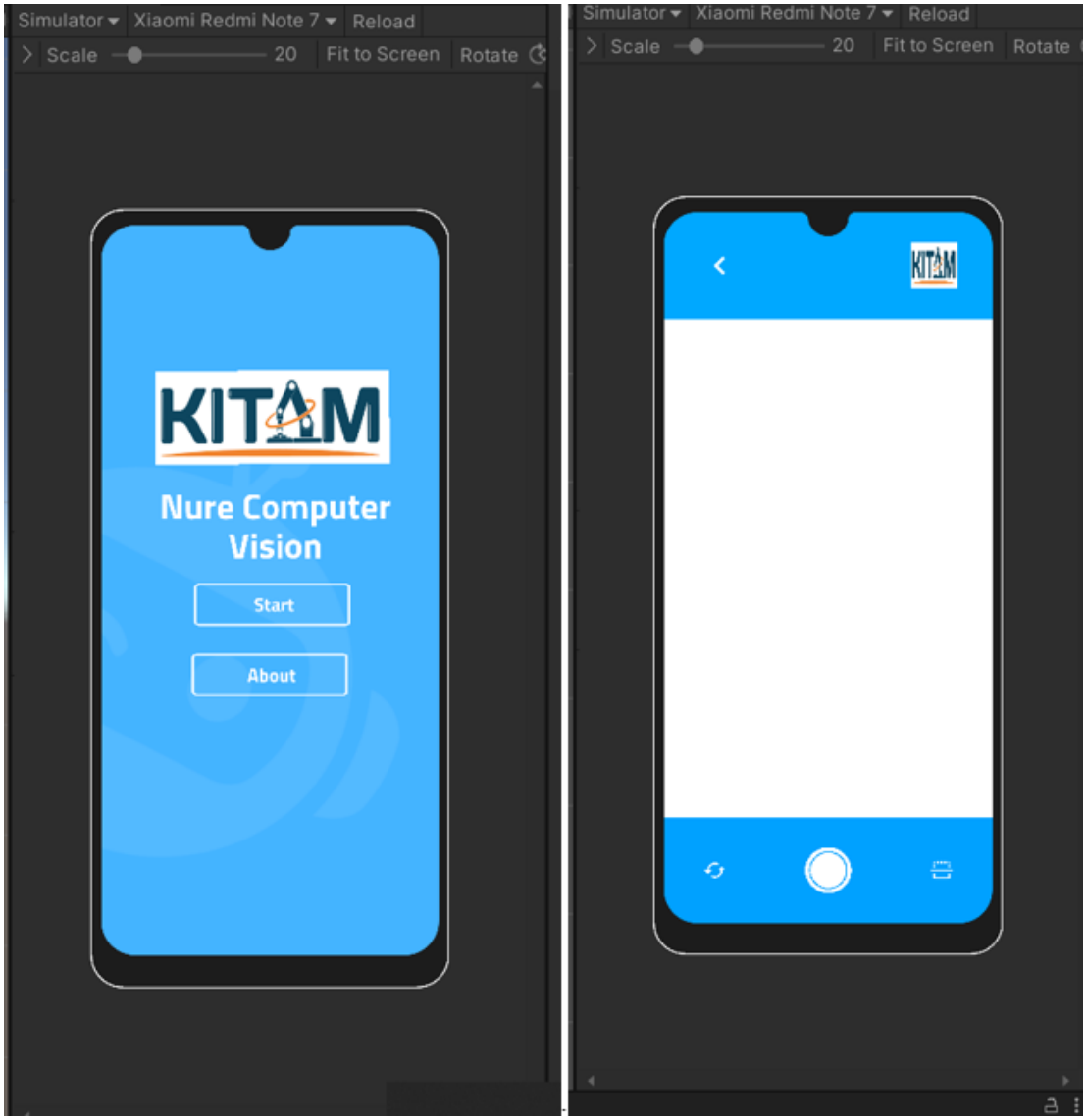


Рисунок 4.7 – Xiaomi Readmi Note 7

Симулятора пристрою не імітує наступне:

– характеристики продуктивності пристрою, наприклад швидкість процесора або доступна оперативна пам’ять;

– можливості візуалізації пристрою;

– вбудовані плагіни, які не працюють у редакторі;

– обертання гіроскопа;

– не всі API змодельованих класів моделюються.

4.6 Розробка програми

Об’єктно-орієнтоване програмування є основним компонентом платформи .NET. Основна ідея ООП полягає в тому, щоб об'єднати в єдиний блок як дані, так і методи, які оперують цими даними – такі одиниці називаються об'єктом. Усі мови ООП забезпечують механізми, які допомагають реалізувати об’єктно-орієнтовану модель. Це інкапсуляція, успадкування, поліморфізм і можливість повторного використання.

На рис 4.8 наведено підключення необхідних компонентів у Visual Studio Installer.

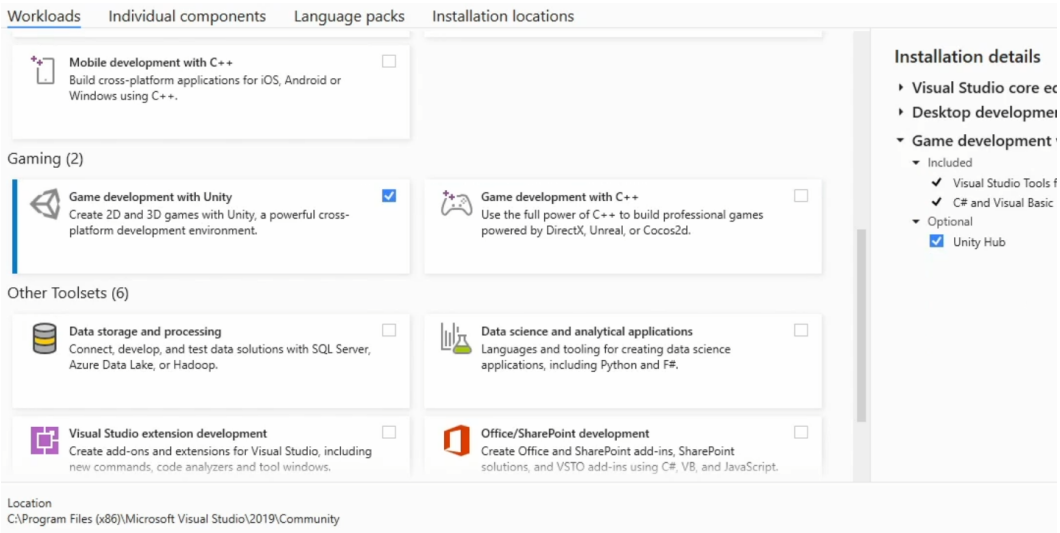


Рисунок 4.8 – Підключення необхідних для розробки бібліотек

Завдяки блокнотам Colab можна використовувати в одному документі код, форматований текст, зображення, розмітку HTML, набір LaTeX і не тільки. Блокноти Colab зберігатимуться на Google Диску. Можливо відкрити доступ до них колегам або друзям, дозволивши їм переглядати або навіть редагувати документ, а також залишати коментарі. Colab дозволяє використовувати для аналізу та візуалізації даних усі можливості

популярних бібліотек Python. У Colab рис 4.10 можна імпортувати набір даних зображення, зорієнтувати на нього класифікатор зображень та оцінити модель за допомогою декількох рядків коду. Код у блокнотах Colab виконується на хмарних серверах Google. Це дає можливість використовувати апаратне забезпечення Google, у тому числі графічні процесори та TPU незалежно від потужності машини. Потрібний лише браузер. В результаті отримаємо вже треновану нейронну мережу, яку можна вже застосувати в методах застосунку рис. 4.9-4.12.

```
public void OnWebCamTextureToMatHelperInitialized()
public void OnWebCamTextureToMatHelperDisposed()
public void OnWebCamTextureToMatHelperErrorOccurred(WebCamTextureToMatHelper.ErrorCode
errorCode)
void OnDestroy()
public void OnPlayButtonClick()
public void OnPauseButtonClick()
public void OnStopButtonClick()
public void OnChangeCameraButtonClick()
public void OnRequestedResolutionDropdownValueChanged(int result)
public void OnRequestedFPSDropdownValueChanged(int result)
public void OnRotate90DegreeToggleValueChanged()
public void OnFlipVerticalToggleValueChanged()
public void OnFlipHorizontalToggleValueChanged()
public enum FPSPreset
public enum ResolutionPreset
private void Dimensions(ResolutionPreset preset, out int width, out int height
protected virtual void Run()
void Update()
protected virtual List<string> readClassNames(string filename)
rotected virtual void postprocess(Mat frame, List<Mat> outs, Net net)
protected virtual void drawPred(int i,int classId, float conf, double left, double
top, double right, double bottom, Mat frame)
protected virtual List<string> getOutputsNames(Net net)
protected virtual List<string> getOutputsTypes(Net net)
```

Рисунок 4.9 – Методи які використовує програма

```
    if (!string.IsNullOrEmpty(classes)) classes_filepath = Utils.getFilePath("dnn/" + classes);
    if (!string.IsNullOrEmpty(config)) config_filepath = Utils.getFilePath("dnn/" + config);
    if (!string.IsNullOrEmpty(model)) model_filepath = Utils.getFilePath("dnn/" + model);
    Run();

}
```

Рисунок 4.10 – Підключення класів, ваг та файлу конфігурації до Visual Studio

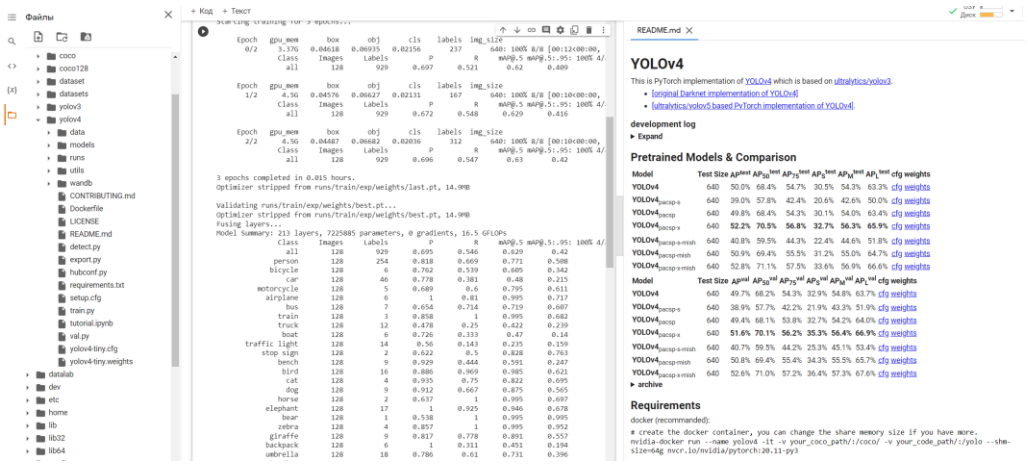


Рисунок 4.11 – Вікно навчання нейронних мереж в Google colab

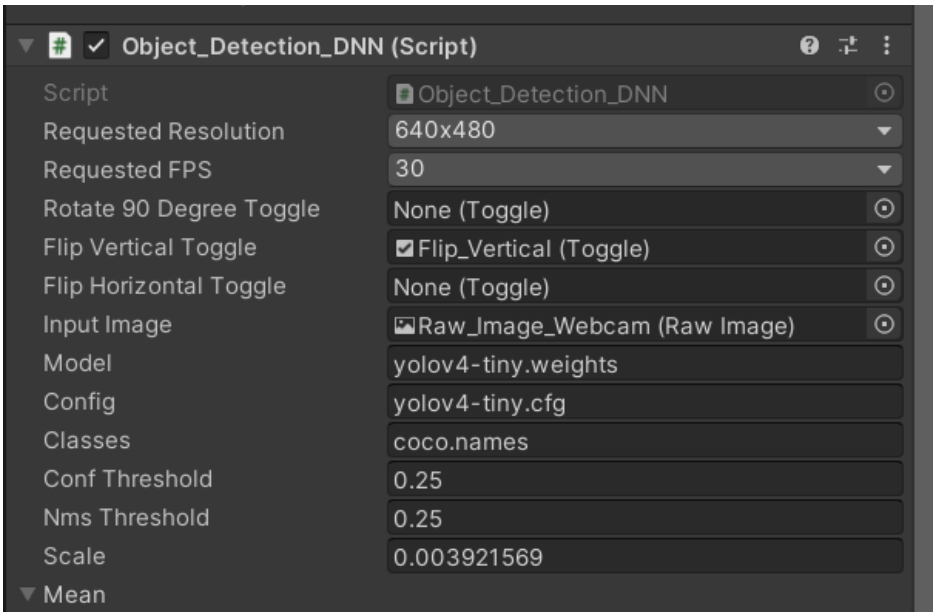


Рисунок 4.12 – Підключення класів, ваг та файлу конфігурації мережі

4.7 Висновки до 4 розділу

Щоб створити сучасну програму, недостатньо виконувати певні інструкції. Основне це розробити виразну структуру програми, яка буде придатна в виконанні, стійкою до змін і застрахованою від помилок. Після аналізу та порівняння існуючих бібліотек і мов програмування було обрано YoLo та мову C#. На основі розробленої схеми та об’єктної моделі, було розроблено інтерфейс, що відповідає поставленим у даному розділі вимогам. Також перевірено використання застосунку на різних пристроях,

завдяки чому можна переконатись у адаптивності розробленого інтерфейсу.

Використовуючи середовище розробки Unity and Visual Studio було розроблено код для програми у якому реалізовано підключення нейронної мережі, що була навчена під час виконання кваліфікаційної роботи.

5 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ АВТОМАТИЗОВАНОГО МЕТОДУ РОЗПІЗНАВАННЯ ТРАНСПОРТНИХ ТЕХНОЛОГІЧНИХ ОБ'ЄКТІВ

5.1 Проведення експериментів та аналіз отриманих результатів

Враховуючі тему кваліфікаційної роботи було обрано провести експеримент з сучасними методами, що найчастіше використовуються для створення нейронних мереж з розпізнавання транспортних технологічних об'єктів .

Були проведені експерименти тренування мереж з бібліотеки tensorflow та мережі де базовими і основними блоками є згорткові шари. В експерименті було проведено аналіз ефективності двох мереж які були імплементовані різними способами з підключенням необхідних біліотеки (рис 5.1).

Результати розпізнавання транспортних технологічних об'єктів за допомогою мережі яка не використовує згорткові шари в архітектурі рис. 5.2-5.3.

Importing the libraries

```
import cv2
import numpy as np
import os
import zipfile
from google.colab.patches import cv2_imshow
import tensorflow as tf
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
tf.__version__
```

```
2.7.0
```

Рисунок 5.1 – Підключення бібліотек

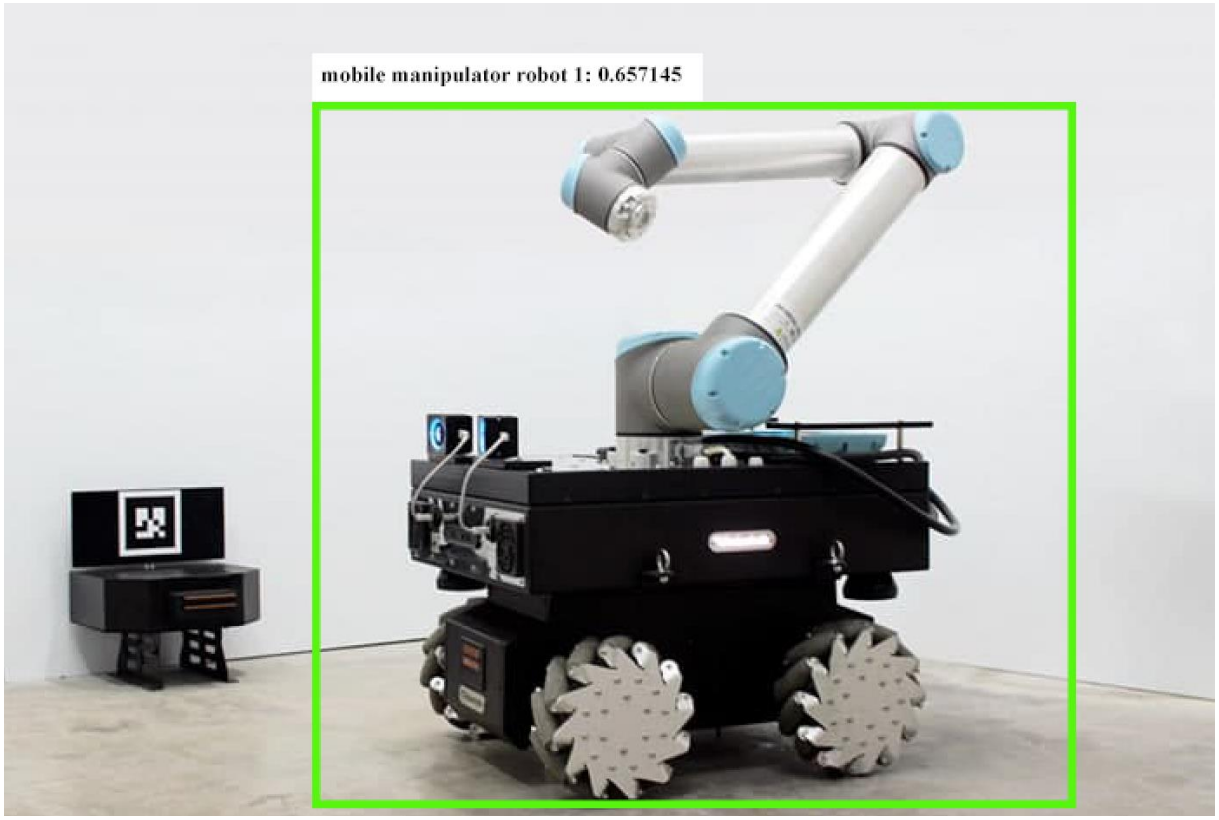


Рисунок 5.2 – Результати 1

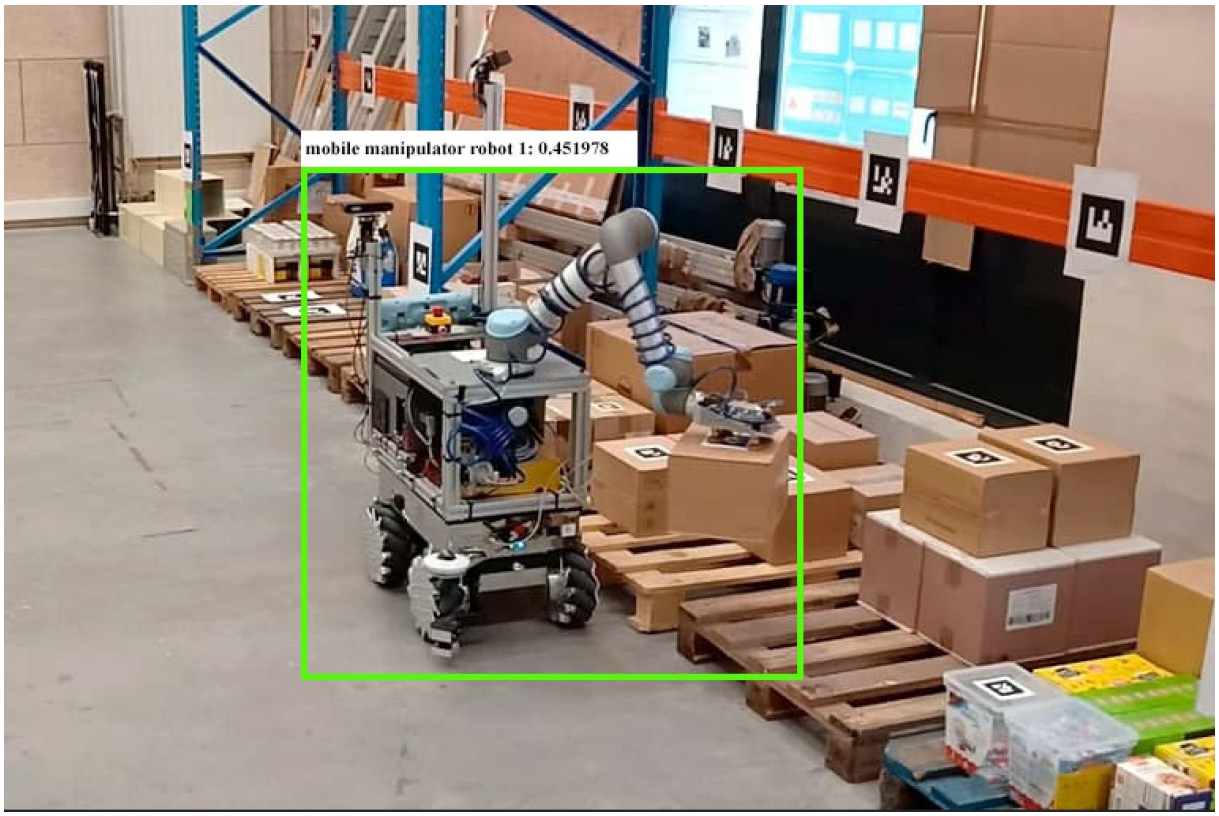


Рисунок 5.3 – Результати 2

Значення похибки та точності для мережі яка не використовує згорткові шари в архітектурі зображено на рис. 5.4-5.5.

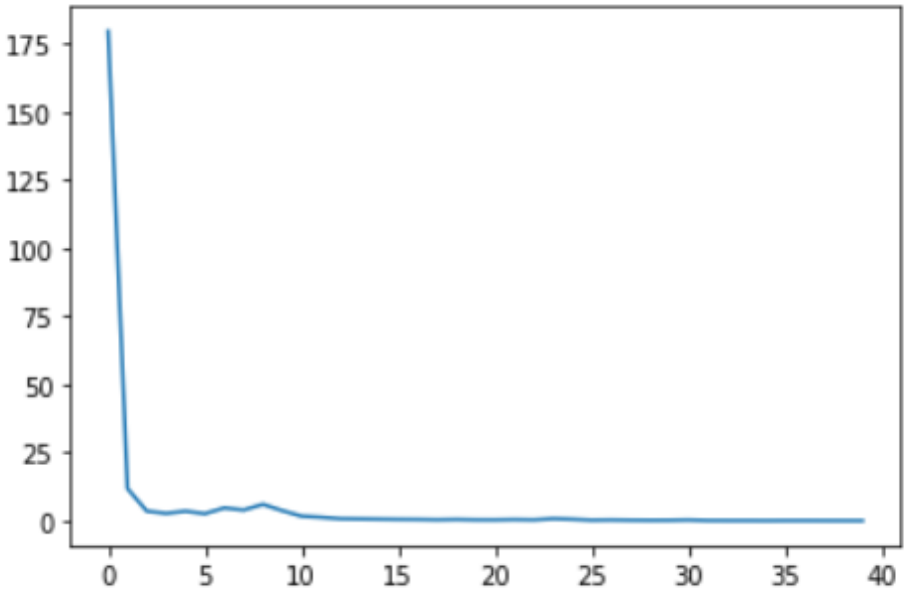


Рисунок 5.4 – Значення похибки для мережі яка не використовуе згорткові слої в архітектурі

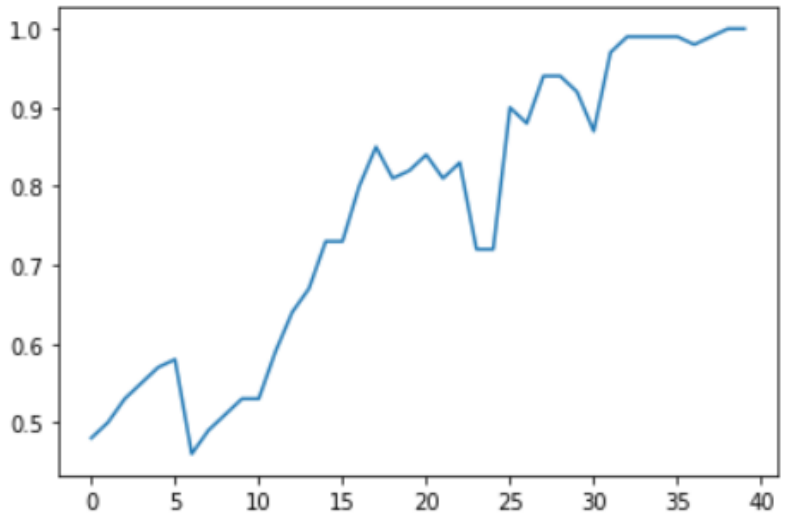


Рисунок 5.5 – Значення точності для мережі яка не використовуе згорткові слої в архітектурі

Результати розпізнавання транспортних технологічних об’єктів за допомогою мережі яка використовує згорткові шари в архітектурі

зображено на рис. 5.6-5.7.

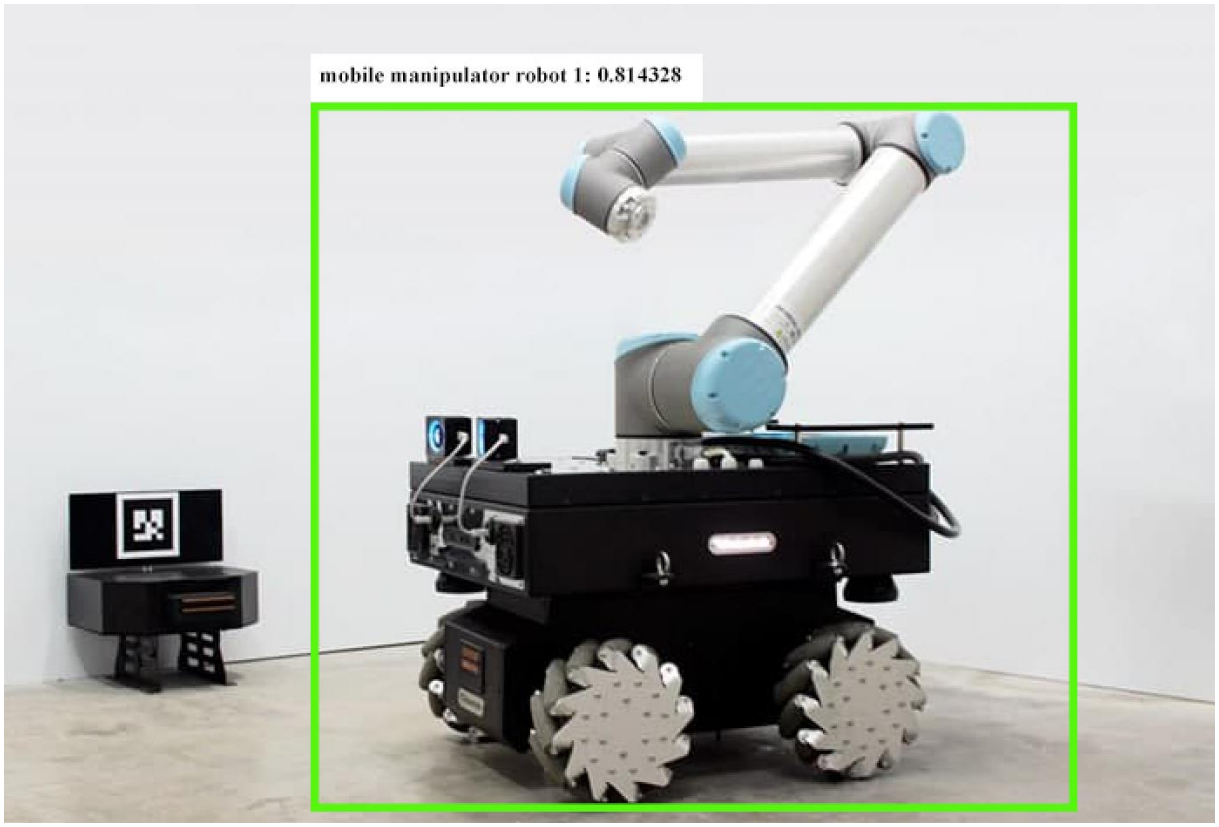


Рисунок 5.6 – Результати розпізнавання транспортних технологічних об’єктів за допомогою мережі яка використовує згорткові шари в архітектурі

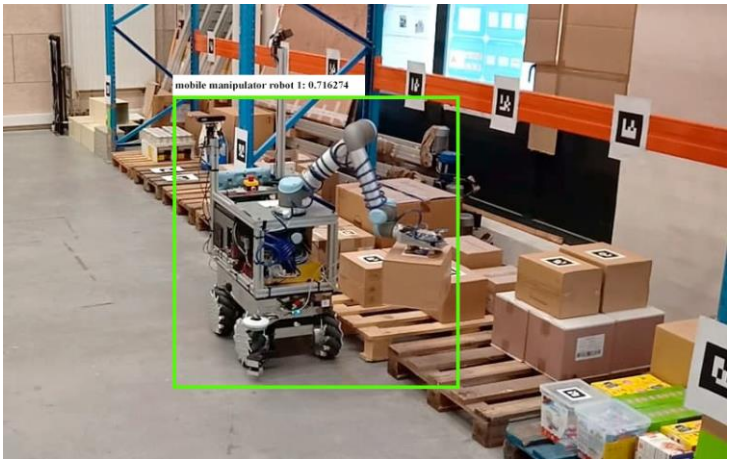


Рисунок 5.7 – Результати розпізнавання транспортного технологічного об’єкта за допомогою мережі яка використовує згорткові шари в архітектурі

Значення похибки та точності для мережі яка використовує згорткові шари в архітектурі зображено на рис. 5.8-5.9.

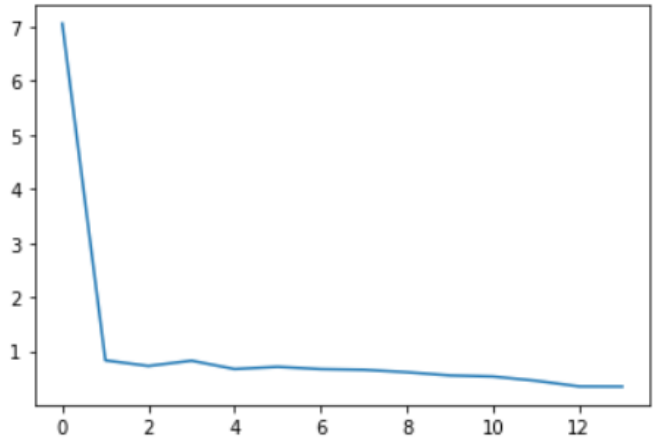


Рисунок 5.8 – Значення похибки для мережі яка використовуе згорткові слої в архітектурі

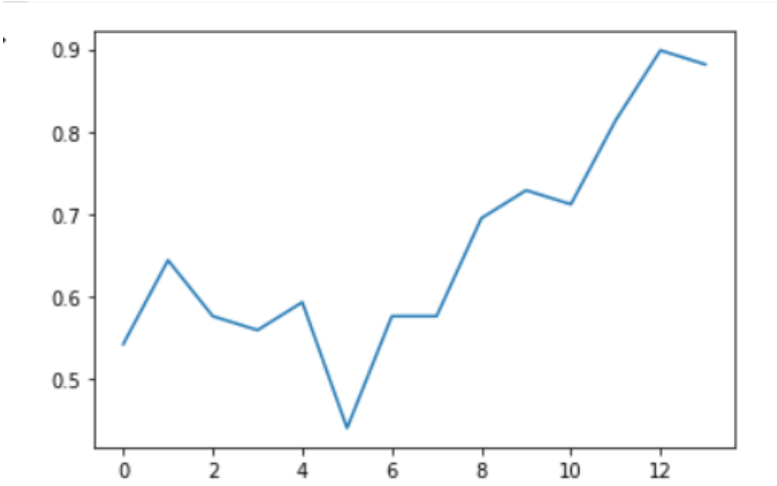


Рисунок 5.9 – Значення похибки для мережі яка використовуе згорткові слої в архітектурі

Базуючись на даних отриманих даних під час другого експерименту було побудовано графіки точості двох нейронних мереж в залежності від їч архітектури. Можна зробити висновок що ахітектура мережі яка має глибокі шари є менш ефективною в порівнянні з архітектурою яка має згортовкові шари в архітектурі. Згорткові шари – це шари, де фільтри застосовуються до

вихідного зображення або до інших особливостей об’єктів у глибокій CNN.

В таблицях 5.1-5.3 приведено результати експерименту тренування нейронних мереж в залежності від ітерації.

Таблиця 5.1 – Точність YOLOv4 в 3 ітераціях навчанння

Class	Images	Labels	P
all	128	929	0.597
all	128	929	0.542
all	128	929	0.637

Таблиця 5.2 – Точність YOLOv3 в 3 ітераціях навчанння

Class	Images	Labels	P
all	128	929	0.423
all	128	929	0.369
all	128	929	0.354

Таблиця 5.3 – Точність YOLOv4 в 10 ітераціях навчанння

Class	Images	Labels	P
all	128	929	0.542
all	128	929	0.651
all	128	929	0.532
all	128	929	0.596
all	128	929	0.428
all	128	929	0.573
all	128	929	0.572
all	128	929	0.691
all	128	929	0.732
all	128	929	0.811

В ході виконнання навчанння YOLOv3 та YOLOv4 виходячи з данних що відображені в табл. 5.1 – 5.3 можна переконатися у єфективності саме обранної нейронної мережі та її точності.

Дослідивши отримані дані також було проведено навчання обранної для виконнаня кваліфікаційної роботи нейронної мережі та використавши її було розроблено програму для розпізнавання транспортних технологічних

об’єктів.

5.2 Заходи з охорони праці

Проектування робочих місць, забезпечених відеотерміналами, відноситься до числа найважливіших проблем ергономічного проектування в області обчислювальної техніки.

Робоче місце і взаємне розташування всіх його елементів повинно відповідати антропометричним, фізичним і психологічним вимогам. Велике значення має також характер роботи. Зокрема, при організації робочого місця програміста повинні бути дотримані наступні основні умови:

– оптимальне розміщення обладнання, що входить до складу робочого місця;

– достатній робочий простір, що дозволяє здійснювати всі необхідні рухи і переміщення;

– необхідно природне і штучне освітлення для виконання поставлених завдань;

– рівень акустичного шуму не повинен перевищувати допустимого значення.

– достатня вентиляція робочого місця.

Ергономічними аспектами проектування відеотермінальних робочих місць, зокрема, є: висота робочої поверхні, розміри простору для ніг, вимоги до розташування документів на робочому місці (наявність і розміри підставки для документів, можливість різного розміщення документів, відстань від очей користувача до екрану, документа, клавіатури тощо), характеристики робочого крісла, вимоги до поверхні робочого столу, урегульованість робочого місця і його елементів.

При проектуванні письмового столу слід враховувати наступне:

– висота столу повинна бути обрана з урахуванням можливості сидіти

вільно, в зручній позі, при необхідності спираючись на підлокітники;

– нижня частина столу повинна бути сконструйована так, щоб програміст міг зручно сидіти, не був змушений підтискати ноги;

– поверхня столу повинна володіти властивостями, що виключають появу відблисків в полі зору програміста;

– конструкція столу повинна передбачати наявність висувних ящиків (не менше 3 для зберігання документації, лістингів, канцелярського приладдя, особистих речей).

Висота робочої поверхні рекомендується в межах 680 мм – 760 мм. Висота робочої поверхні, на яку встановлюється клавіатура, повинна бути 650 мм.

Велике значення надається характеристикам робочого крісла. Так, рекомендується висота сидіння над рівнем підлоги повинна бути в межах 420 мм – 550 мм.

Поверхня сидіння рекомендується робити м'якою, передній край заокругленим, а кут нахилу спинки робочого крісла – регульованим.

Необхідно передбачати при проектуванні можливість різного розміщення документів: збоку від відео-терміналу, між монітором і клавіатурою і т. п. Крім того, у випадках, коли відео-термінал має низьку якість зображення, наприклад помітні мелькання, відстань від очей до екрану роблять більше (близько 700 мм), ніж відстань від ока до документа (300 мм – 450 мм).

Взагалі при високій якості зображення на відеотерміналі відстань від очей користувача до екрану, документа і клавіатури може бути рівним.

Положення екрану визначається:

– відстанню зчитування (0,60 м + 0,10 м);

– кутом зчитування, напрямком погляду на 20 нижче горизонталі до центру екрану, причому екран перпендикулярний цьому напрямку.

Повинна передбачатися можливість регулювання екрану:

– по висоті + 3 см;

– по нахилу від 10 см до 20 см щодо вертикалі;

– у лівому і правому напрямках.

Зоровий комфорт підпорядковується двом основним вимогам:

– чіткості на екрані, клавіатурі і в документах;

– освітленості і рівномірності яскравості між навколишніми умовами і різними ділянками робочого місця.

Велике значення також надається правильній робочій позі користувача. При незручній робочій позі можуть з'явитися болі в м'язах, суглобах і сухожиллях. Вимоги до робочої пози користувача відеотерміналу наступні:

– шия не повинна бути нахилена більш ніж на 20º (між віссю «голова-шия» і віссю тулуба);

– плечі повинні бути розслаблені;

– лікті – перебувати під кутом 80º – 100º;

– передпліччя і кисті рук – в горизонтальному положенні.

Причина неправильної пози користувачів обумовлена наступними факторами: немає хорошої підставки для документів, клавіатура знаходиться занадто високо, а документи - занадто низько, нікуди покласти руки і кисті, недостатньо простір для ніг.

З метою подолання зазначених недоліків даються загальні рекомендації:

– краще пересувна клавіатура, ніж вбудована;

– повинні бути передбачені спеціальні пристосування для регулювання висоти столу, клавіатури, документів і екрану, а також підставка для рук.

Характеристики використовуваного робочого місця:

– висота робочої поверхні столу 750 мм;

– висота простору для ніг 650 мм;

– висота сидіння над рівнем підлоги 450 мм;

– поверхня сидіння м'яка з заокругленим переднім краєм;

– передбачена можливість розміщення документів справа і зліва;

– відстань від ока до екрану 700 мм;

– відстань від ока до клавіатури 400 мм;

– відстань від ока до документів 500 мм;

– можливе регулювання екрану по висоті, по нахилу, в лівому і в правому напрямках.

Створення сприятливих умов праці і правильне естетичне оформлення робочих місць на виробництві має велике значення як для полегшення праці, так і для підвищення його привабливості, позитивно впливає на продуктивність праці.

Забарвлення приміщень і меблів повинна сприяти створенню сприятливих умов для зорового сприйняття, гарного настрою.

У службових приміщеннях, в яких виконується одноманітна розумова робота, що вимагає значного нервового напруження і великого зосередження, забарвлення повинна бути спокійних тонів – малонасичені відтінки холодного зеленого або блакитного кольорів.

5.3 Висновки до 5 розділу

Враховуючи отримані дані з 5 розділу, роботу застосунку було перевірено провівши експерименти в ході яких досліджено різні методи розпізнавання зображення, порівняно YOLOv3 та YOLOv4 при виконані 3 проходів навчання, YOLOv4 також досліджено при 10 проходах. Отримані дані для Yolov4 при 3 та 10 проходах було порівняно та занесено до таблиць.

Отримані дані свідчать, що Yolov4 при 10 проходах має достатню точність, що становить 88 % та відповідае вимогам точності які висуваються до нейронної мережі використаній у данній роботі.

ВИСНОВКИ

В роботі був розроблений метод розпізнавання транспортних технологічних об’єктів, що дозволив підвищити ефективність виробництва шляхом впровадженням розробленого автоматизованого метода який використовує нейронну мережу. Для досягнення поставленої мети були вирішені такі завдання: проведено аналіз сучасного стану проблеми розпізнавання технологічних об’єктів за допомогою нейронної мережі, розробку структурної схеми та об'єктної моделі програмного засобу, обґрунтування обраної нейронної мережі та вибір інструментів для програмної реалізації, проведено експериментальні дослідження на розробленому програмному засобі, зроблено аналіз методів розпізнавання об’єктів, архітектури і ефективності нейронних мереж.

Ґрунтуючись на аналізі, було розроблено структурну та об’єктну схему застосунку. На основі структурної схеми здійснено розробку програми, яка відповідає основним принципам ООП. Під час розробки було використано ряд архітектурних патернів, а саме поведінкову параметризацію, яка дозволяє не писати однакові методи та робить програму більш гнучкою для подальших змін. На базі системи розпізнавання YOLOv4 було обрано і натреновано та оптимізовано ваги нейронної мережі, які дозволяють більш ефективно використовувати ресурси системи. Внаслідок цього застосунок можна використовувати в доступних моделях мобільних телефонів, які мають скромні характеристики камер та процесорів. В програмі Unity розроблений інтерфейс користувача, який відповідає сучасним та поставленим вимогам.

Здійснено ряд експериментів по дослідженню ефективності існуючих методів розпізнавання зображень, здійснено порівняння різних моделей нейронних мереж, які специфікуються на розпізнаванні об’єктів.

Ґрунтуючись на аналізі, було обрано модель багатошарового

персиптрону, так як автотранспортні об’єкти мають різні особливості та конструкцію в цілому. Внаслідок цього було вибрано систему розпізнавання об’єктів, яка має згорткову нейронну мережу в архітектурі.

Завдяки основним перевагам розробленого застосунок, таким як можливість розпізнавання транспортних технологічних об’єктів та їх підрахунок, вважаю, що розроблене рішення є готовим для впровадження і подальшого використання на підприємстві.

Упровадження розробленого програмного засобу можливе в комп’ютерно-інтегрованих автоматизованих системах в виробничому процесі, так і в начальних лабораторіях у освітньому процесі на лабораторних практикумах та практичних роботах в межах спеціальності 151 Автоматизація та комп’ютерно-інтегровані технології та можуть бути цікавим для фахівців, що пов’язані з промисловою автоматизацією.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008: 2015. Інформація та документація. Звіти у сфері науки і техніки. Структура і правила оформлення. – К.: ДП “УкрНДНЦ”, 2016. – 30 с.

2. Положення про кваліфікаційну роботу здобувача вищої освіти на другому (магістерському) рівні [Електронний ресурс] : Наказ ХНУРЕ від 06 травня 2021 р. No 143. – Режим доступу : https://nure.ua/wp-content/uploads/Main_Docs_NURE/143-vid-06.05.2021-pro-vvedennja-v-diju-rishennja-vchenoi-radi -universitetu.pdf

3. Методичні вказівки з підготовки й оформлення кваліфікаційної роботи здобувачами другого (магістерського) рівня вищої освіти спеціальності 151 Автоматизація та комп’ютерно-інтегровані технології, освітньо-професійних програм: «Автоматизоване управління тех / І. Ш.Невлюдов,, Р. В. Артюх,, Н. П. Демська,, О. І. Филипенко. – Харків, 2021. – 50 с.

4. Левченко Є. О. Концепція гнучкого виробництва [Текст] / Сітало І. А., Є. О. Левченко, К. Г. Медова, // Автоматизація та приладобудування («Automation and Development of Electronic Devices» ADED-2019): збірник студентських наукових статей / Харківський національний університет радіоелектроніки; [редкол.: І.Ш. Невлюдов та ін.]. – Харків : ХНУРЕ, 2019. – Вип. 2. – 199 с.

5. Левченко Є. О. Перспективи розвитку автоматизованого виробництва [Текст] / І. А. Сітало, Є. О. Левченко, К. Г. Медова, // Автоматизація та приладобудування («Automation and Development of Electronic Devices» ADED-2019): збірник студентських наукових статей / Харківський національний університет радіоелектроніки; [редкол.: І.Ш. Невлюдов та ін.]. – Харків : ХНУРЕ, 2019. – Вип. 2. – 199 с.

6. Левченко Є. О. Аналіз технологій та методів обробки

монокристалічних матеріалів [Текст] / Є. О. Левченко // Збірник студентських наукових статей «автоматизація та приладобудування» АПР-2018 1 частина (Випуск 2018): - Харків/ Редкол.: Невлюдов І.Ш.(голова), та інш. Харків: Вид-во Харківського національного університету радіоелектроніки [електронне видання], 2018. - 237с.

7. Левченко Є. О. Аналіз систем автоматичного контролю [Текст] / Є. О. Левченко, І. А. Сітало // Автоматизація та приладобудування («Automation and Development of Electronic Devices» ADED-2018) [Електронний ресурс] : збірник студентських наукових статей / Харківський національний університет радіоелектроніки ; [редкол.: І.Ш. Невлюдов та ін.]. – Харків : ХНУРЕ, 2018. – Вип. 2. – 227 с.

8. Левченко Є. О. Исследование оптической восприимчивости поглощения и оптического усиления [Текст] / Є. О. Левченко, І. А. Сітало // Автоматизація та приладобудування («Automation and Development of Electronic Devices» ADED-2018) [Електронний ресурс] : збірник студентських наукових статей / Харківський національний університет радіоелектроніки ; [редкол.: І.Ш. Невлюдов та ін.]. – Харків : ХНУРЕ, 2018. – Вип. 2. – 227 с.

9. Левченко Є. О. Сенсорне керування автомобілем [Текст] / Мажара А. Є., Васильченко О. С., Чала О. О., // Синергетика, мехатроніка, телематика дорожніх машин і систем у навчальному процесі та науці. Збірник наукових праць за матеріалами ІІ міжнародної науково-практичної конференції. – Харків, ХНАДУ, 2018. – 184 с.

10. Левченко Є. О. Розробка структурної схеми автономного зарядного пристрою від сонячного світла [Текст] / Васильченко О. С., Левченко Є. О., Бурма О. Н. // Автоматизація та приладобудування («Automation and Development of Electronic Devices» ADED-2019): збірник студентських наукових статей / Харківський національний університет радіоелектроніки; [редкол.: І.Ш. Невлюдов та ін.]. – Харків : ХНУРЕ, 2019. – Вип. 1. – 207 с.

11. Левченко Є. О. Система візуального контролю на виробництві [Текст] / Васильченко О. С., Левченко Є. О., Бурма О. Н. // Автоматизація та приладобудування («Automation and Development of Electronic Devices» ADED-2019): збірник студентських наукових статей / Харківський національний університет радіоелектроніки; [редкол.: І.Ш. Невлюдов та ін.]. – Харків : ХНУРЕ, 2019. – Вип. 1. – 207 с.

12. Подання знань у системах прийняття рішень інтелектуальних роботів / О. М. Цимбал – Харків, 2021. – (ХНУРЭ). – (2).

13. Невлюдов І. Ш. Transfer of technologies from modern science, education and technology in the minds of the fourth industrial revolution "INDUSTRIA 4.0" / І. Ш. Невлюдов, О. О. Чала, Ю. М. Олександров. – Дніпро, 2019.. – Т.2 С.: 604-608

14. Невлюдов І. Ш. Investigation of the technique for selecting the contours of objects / І. Ш. Невлюдов, А. І. Демская, Н. П. Демская // XIV International Scientific and Technical Conference “Physical Processes and Fields of Technical and Biological Objects”: materials of the conference. / І. Ш. Невлюдов, А. І. Демская, Н. П. Демская. – Кременчук: КрНУ, 2015. – С. 95–96.

15. Хрустальов К. Л. Automation of control processes for autonomous robotic platforms based on the use of technical vision systems / К. Л. Хрустальов, А. О. Функендорф, Д. А. Кобеляцький, 2018. – С. 31–34..

16. Visual monitoring of the break surface of the installation connection of electronic equipment / Nevliudov I., Starodubcev N., Demska N., Omarov Sh. // Information system and innovative technologies in projekt management [Text]: Collective monograph edited byl. Linde, I. Chumachenko, V. Timofeyev. – Kharkiv : NURE, 2019. – p. 258-270.

17. Object Model [Електронний ресурс] – Режим доступу до ресурсу: https://www.techopedia.com/definition/8635/object-model/. – 05.11.2021.

18. What Is Perceptron? A Basic Overview For 2021 [Електронний ресурс] – Режим доступу до ресурсу:

https://www.jigsawacademy.com/blogs/ai-ml/perceptron/. – 05.11.2021.

19. Hands-On Guide To Weights and Biases [Електронний ресурс] – Режим доступу до ресурсу : https://analyticsindiamag.com/hands-on-guide-to-weights-and-biases-wandb-with-python-implementation/. – 05.11.2021.

20. TensorFlow - Single Layer Perceptron [Електронний ресурс] – Режим доступу до ресурсу: https://www.tutorialspoint.com/tensorflow/tensorflow_single_layer_perceptron.html. – 05.11.2021.

21. TensorFlow - Linear Regression [Електронний ресурс] – Режим доступу до ресурсу : https://www.tutorialspoint.com/tensorflow/tensorflow_linear_regression.html. – 05.11.2021.

22. TensorFlow - CNN And RNN Difference [Електронний ресурс] – Режим доступу до ресурсу : https://www.tutorialspoint.com/tensorflow/tensorflow_cnn_and_rnn_difference.html. – 05.11.2021.

23. TensorFlow - Keras [Електронний ресурс] – Режим доступу до ресурсу: https://www.tutorialspoint.com/tensorflow/tensorflow_keras.htm.

24. TensorFlow - CNN And RNN Difference [Електронний ресурс] – Режим доступу до ресурсу : https://www.tutorialspoint.com/tensorflow/tensorflow_cnn_and_rnn_difference.html. – 05.11.2021.

25. TensorFlow - TensorFlow - Multi-Layer Perceptron Learning [Електронний ресурс] – Режим доступу до ресурсу : https://www.tutorialspoint.com/tensorflow/tensorflow_multi_layer_perceptron_learning.htm.

26. TensorFlow - TensorFlow - Hidden Layers of Perceptron [Електронний ресурс] – Режим доступу до ресурсу : https://www.tutorialspoint.com/tensorflow/tensorflow_hidden_layers_of_perceptron.html. – 05.11.2021.

27. TensorFlow - Optimizers [Електронний ресурс] – Режим доступу до ресурсу : https://www.tutorialspoint.com/tensorflow/tensorflow_cnn_and_rnn_difference.html. – 05.11.2021.

28. TensorFlow - Gradient Descent Optimization [Електронний ресурс] – Режим доступу до ресурсу : https://www.tutorialspoint.com/tensorflow/

tensorflow_gradient_descent_optimization.htm. – 05.11.2021.

29. Image Recognition using TensorFlow [Електронний ресурс] – Режим доступу до ресурсу : https://www.tutorialspoint.com/tensorflow/image_recognition_using_tensorflow.htm. – 05.11.2021.

30. ML | Momentum-based Gradient Optimizer introductions [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/ml-momentum-based-gradient-optimizer-introduction/. – 05.11.2021.

31. Clustering in Machine Learning [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/clustering-in-machine-learning/. – 10.11.2021.

32. ML | Hierarchical clustering (Agglomerative and Divisive clustering) [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/ml-hierarchical-clustering-agglomerative-and-divisive-clustering/. – 10.11.2021.

33. Different Types of Clustering Algorithm [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/different-types-clustering-algorithm/?ref=lbp. – 10.11.2021.

34. Clustering in Machine Learning [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/clustering-in-machine-learning/. – 10.11.2021.

35. ML | Spectral Clustering [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/ml-spectral-clustering/?ref=lbp. – 10.11.2021.

36. ML | Types of Learning – Supervised Learning [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/ml-types-learning-supervised-learning/?ref=lbp. – 10.11.2021.

37. Introduction to Artificial Neutral Networks | Set 1 [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/introduction-to-artificial-neutral-networks/.

38. Introduction to Artificial Neural Network | Set 2 [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/introduction-artificial-neural-network-set-2/. – 10.11.2021.

39. Introduction to ANN (Artificial Neural Networks) | Set 3 (Hybrid Systems) [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/introduction-ann-artificial-neural-networks-set-3-hybrid-systems/. – 10.11.2021.

40. Introduction to ANN | Set 4 (Network Architectures) [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/ml-types-learning-supervised-learning/?ref=lbp. – 10.11.2021.

41. Activation functions in Neural Networks [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/activation-functions-neural-networks/. – 16.11.2021.

42. Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis [Електронний ресурс] – Режим доступу до ресурсу : https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141. – 16.11.2021.

43. A single neuron neural network in Python [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/single-neuron-neural-network-python/. – 16.11.2021.

44. Introduction to Convolution Neural Network [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/introduction-convolution-neural-network/. – 16.11.2021.

45. CNN | Introduction to Pooling Layer [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/. – 16.11.2021.

46. CNN | Introduction to Padding [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/cnn-introduction-to-padding/. – 16.11.2021.

47. ML | Types of padding in convolution layer [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/types-of-padding-in-convolution-layer/. – 16.11.2021.

48. Нейрон [Електронний ресурс] – Режим доступу до ресурсу : https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%B9%D1%80%D0%BE%D0%BD. – 16.11.2021.

49. Introduction to Recurrent Neural Network [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/. – 16.11.2021.

50. Recurrent Neural Networks Explanation [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/recurrent-neural-networks-explanation/. – 16.11.2021.

51. Deep Learning | Introduction to Long Short Term Memory [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/. – 19.11.2021.

52. Gated Recurrent Unit Networks [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/gated-recurrent-unit-networks/. – 19.11.2021.

53. ML | Text Generation using Gated Recurrent Unit Networks [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/ml-text-generation-using-gated-recurrent-unit-networks/. – 19.11.2021.

54. Generative Adversarial Network (GAN) [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/generative-adversarial-network-gan/. – 19.11.2021.

55. Use Cases of Generative Adversarial Networks [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/use-cases-of-generative-adversarial-networks/. – 19.11.2021.

56. Building a Generative Adversarial Network using Keras [Електронний

ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/building-a-generative-adversarial-network-using-keras/. – 19.11.2021.

57. Deep Q-Learning [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/deep-q-learning/. – 19.11.2021.

58. Gradient Descent algorithm and its variants [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/gradient-descent-algorithm-and-its-variants/. – 19.11.2021.

59. ML | Stochastic Gradient Descent (SGD) [Електронний ресурс] – Режим доступу до ресурсу : https://www.geeksforgeeks.org/ml-stochastic-gradient-descent-sgd/. – 19.11.2021.

60. Рутковская Д. В. Нейронні мережі, генетичні алгоритми та нечіткі системи / Д. В. Рутковская, М. С. Пилиньский, Л. І. Рутковский., 2014. – 384 с.

61. Логинов Д. Machine Learning Refined. Foundations, Algorithms, and Applications 2nd. edition / Д. Логинов, А. Катсаггелос, Б. Реза., 2020.

62. Чжен Э. Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists / Э. Чжен, А. Казарі., 2020. – 240 с.

63. Норвиг П. Artificial Intelligence: A Modern Approach, 4th Edition / П. Норвиг, С. Рассел., 2021. – 706 с.

64. Ніколенко С. І. Глибоке навчання. Занурення в світ нейронних мереж / С. І. Ніколенко, А. А. Кадурін, Е. О. Архангельская., 2019. – 480 с.