Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

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

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

Другий (магістерський)
(рівень вищої освіти)

Підсистема виявлення вибухонебезпечних предметів для багатофункціональної роботизованої платформи
(тема)

Виконав:
студент 2 курсу, групи КТРСм-22-2

Бузніков В.Р.
(прізвище, ініціали)

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

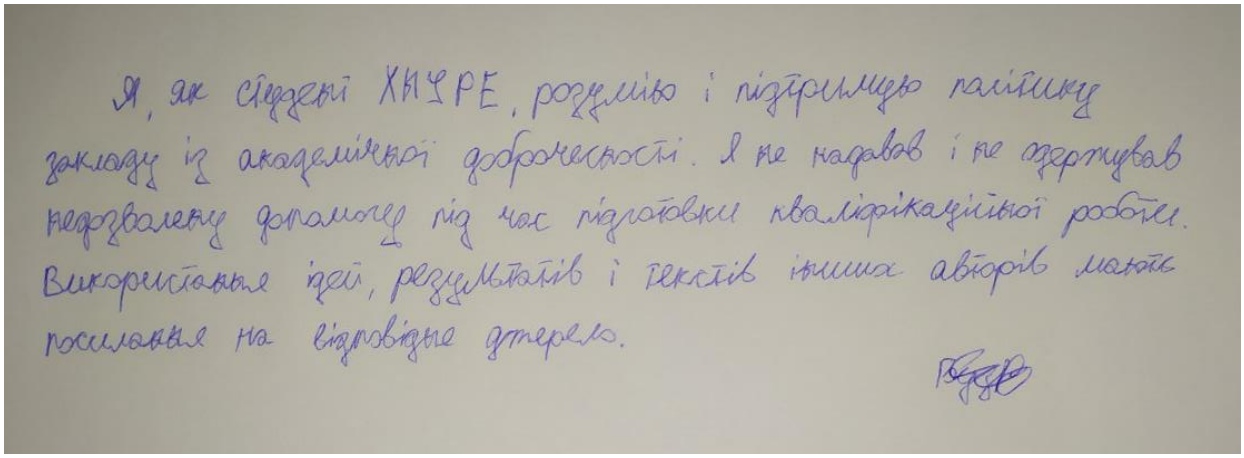
Тип програми Освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп'теризовані та робототехнічні системи
(повна назва освітньої програми)

Керівник доцент Хрустальова С.В.
(посада, прізвище, ініціали)

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

2024 р.



Я, як студент ХНУРЕ, розділяю і підтримую політику закладу із академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідні джерела.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет АКТ
Кафедра КІТАР
Рівень вищої освіти другий (магістерський)
Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
Тип програми Освітньо-професійна
Освітня програма Комп'ютеризовані та робототехнічні системи
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри КІТАР _____
(підпис)
«___»_____20_ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Бузнікову Владиславу Руслановичу
(прізвище, ім'я, по батькові)

1. Тема роботи Підсистема виявлення вибухонебезпечних предметів для багатофункціональної роботизованої платформи

Затверджена наказом по університету від 3.11.2023 № 1288Ст
2. Термін подання студентом роботи до екзаменаційної комісії
3. Вихідні дані до роботи мова програмування Python, YOLOv8, середовище розробки PyCharm, CVAT, LabelImage.

4. Перелік питань, що потрібно опрацювати в роботі
Вступ; Аналіз нейронних мереж та алгоритмів виявлення об'єктів; Розроблення структурної схеми та алгоритму виявлення вибухонебезпечних предметів для багатофункціональної роботизованої платформи; Навчання нейронної мережі для виявлення вибухонебезпечних предметів; Висновки; Додатки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій _____
Графічний демонстративні матеріали в форматі PowerPoint (.ppt) формату*

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	Примітка
1	Аналіз технічного завдання	12.10 – 19.10.23	Виконано
2	Аналіз нейронних мереж та алгоритмів виявлення об'єктів	20.10 – 09.11.23	Виконано
3	Розроблення структурної схеми та алгоритму роботи підсистеми виявлення вибухонебезпечних предметів для багатофункціональної роботизованої платформи	23.11 – 03.12.23	Виконано
4	Навчання нейронної мережі для виявлення вибухонебезпечних предметів	04.12 – 20.12.23	Виконано
5	Подання роботи на перевірку Інтернет-сервісом Unicheck	14.01.2024	Виконано
6	Оформлення пояснювальної записки	15.01.2024	Виконано
7	Подання роботи на рецензію	16.01.2024	Виконано
8	Подання роботи на підпис зав. кафедри	17.01.2024	Виконано
9	Подання кваліфікаційної роботи в ЕК	за день до захисту	Виконано

Дата видачі завдання 10.10.2023

Студент _____ Бузніков В.Р
(підпис) (прізвище, ініціали)

Керівник роботи _____ доцент Хрустальова С.В
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 92 с., 40 рис., 2 дод., 36 джерел.

НЕЙРОННІ МЕРЕЖІ, ВИЯЛЕННЯ ОБ’ЄКТІВ, ШТУЧНИЙ ІНТЕЛЕКТ, YOLO, ГЛИБОКЕ НАВЧАННЯ.

Мета роботи – підвищення ефективності виявлення вибухонебезпечних предметів завдяки використанню комп’ютерного зору для зменшення загроз життю співробітників ДСНС при розмінуванні територій.

Об’єкт розробки – процес виявлення вибухонебезпечних предметів.

Предмет розробки – програмний засіб для виявлення вибухонебезпечних предметів.

Для досягнення мети проаналізовано нейронні мережі, ключові показники ефективності, алгоритми виявлення об'єктів. Розглянуто різницю між машинним та глибоким навчанням, а також типи глибоких нейронних мереж. Обрано мову програмування та середовище розробки. Розроблено алгоритм роботи нейронної мережі. Навчено нейронну мережу виявляти вибухонебезпечні предмети.

Результати навчання приведені у вигляді скріншотів у звіті.

Результати навчання апробовані у науковій статті.

ABSTRACT

Explanatory note: 92 p., 40 pic., 2 applications, 36 sources.

NEURAL NETWORKS, OBJECT DETECTION, ARTIFICIAL INTELLIGENCE, YOLO, DEEP LEARNING.

The purpose of the work is to increase the efficiency of detecting explosive devices through the use of computer vision to reduce threats to the lives of SES employees during mine clearance.

The object of development is the process of detecting explosive objects.

The subject of development is a software tool for detecting explosive objects.

To achieve this goal, neural networks, key performance indicators, and object detection algorithms are analyzed. The difference between machine learning and deep learning, as well as types of deep neural networks are considered. The programming language and development environment are chosen. A neural network algorithm is developed. The neural network is trained to detect explosive objects.

The training results are presented in the form of screenshots in the report.

The training results are tested in a scientific article.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

БШП – багатошарові персептрони (multilayer perceptron, MLP);

ДСНС – державна служба України з надзвичайних ситуацій;

ЗНМ – згорткові нейронні мережі (CNN, ConvNets);

МН – машинне навчання (machine learning, ML);

ПЗ – програмне забезпечення;

ПЗШ – повністю з’єднанні шари (fully connected layers, FC);

РНМ – рекурентна нейронна мережа (recurrent neural network, RNN);

САП – середня абсолютна похибка (Mean Absolute Error, MAE);

ШІ – штучний інтелект;

ROI – регіон інтересів (Region of Interest).

ВСТУП

Впровадження технології комп'ютерного зору в різноманітні галузі стало ключовим компонентом сучасних рішень в області безпеки та автоматизації. Однією з найважливіших задач, яку вирішує ця технологія, є виявлення вибухонебезпечних предметів, яка викликає надзвичайний інтерес і значущість у сфері громадської та таємної безпеки.

Завдяки значному зростанню попиту на розробку автоматизованих систем для виявлення вибухонебезпечних предметів, комп'ютерний зір виявився ключовим інструментом в цьому завданні. Ця технологія дозволяє ефективно аналізувати великі обсяги даних та забезпечує точні та надійні результати без значних втрат часу та ресурсів.

В наш час, коли безпека на транспорті, на великих громадських заходах, у важкодоступних місцях та в інших об'єктах має найвищий пріоритет, розробка автоматизованих модулів для виявлення вибухонебезпечних предметів є вельми актуальною та важливою задачею.

Метою роботи є підвищення ефективності виявлення вибухонебезпечних предметів завдяки використанню комп’ютерного зору для зменшення загроз життю співробітників ДСНС при розмінуванні територій.

Для виконання поставленої мети треба:

– проаналізувати нейронні мережі;

– проаналізувати алгоритми виявлення об’єктів;

– проаналізувати типи глибоких нейронних мереж;

– розробити структурну схему підсистеми виявлення вибухонебезпечних предметів;

– обрати компоненти підсистеми виявлення вибухонебезпечних предметів;

– розробити алгоритм роботи підсистеми виявлення вибухонебезпечних предметів;

– обрати інструменти та розробити алгоритм процесу навчання підсистеми

виявлення вибухонебезпечних предметів;

– реалізувати підсистему виявлення вибухонебезпечних предметів у вигляді програмного засобу.

Оформлення пояснювальної записки відбувалось згідно методичним вказівкам кваліфікаційної роботи магістра [1] та ДСТУ 3008-15 [2], результати навчання нейронної мережі апробовані у науковій статті [3].

1 АНАЛІЗ НЕЙРОННИХ МЕРЕЖ ТА АЛГОРИТМІВ ВИЯВЛЕННЯ ОБ’ЄКТІВ

1.1 Нейрона мережа

Нейронні мережі – це обчислювальні моделі, які імітують складні функції людського мозку. Нейронні мережі складаються із взаємопов'язаних вузлів або нейронів, які обробляють дані та навчаються на них, що дозволяє виконувати такі завдання, як розпізнавання образів і прийняття рішень у машинному навчанні [4].

Найпершим кроком до розуміння того, що робить штучна нейронна мережа, є розуміння нейрона. Нейронні мережі в комп'ютерних науках імітують нейрони людського мозку, звідси і назва "нейронна" мережа. Нейрони мають відростки, що відходять від них з обох кінців, які називаються дендритами. Один нейрон не може зробити багато, але коли тисячі нейронів з'єднуються і працюють разом, вони стають потужними і можуть обробляти складні дії та концепції. Комп'ютерний вузол працює так само, як і людський нейрон, і відтворює реальні нейрони [5]. На рисунку 1.1 зображено вигляд штучного нейрону.

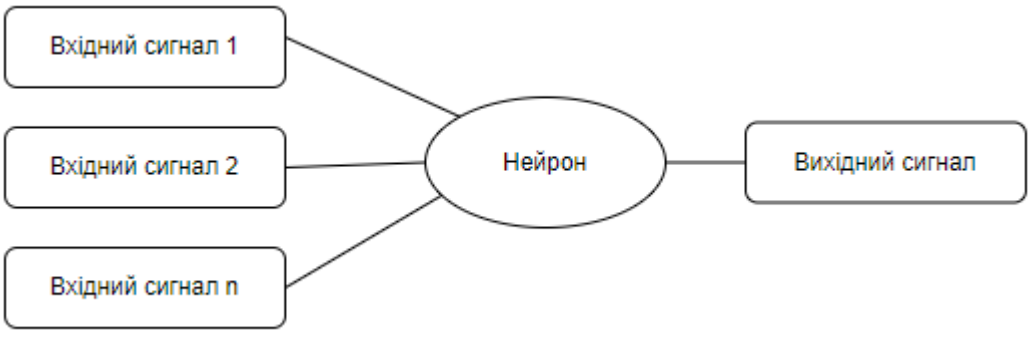


Рисунок 1.1 – Незалежні значення пропускаються через нейрон з метою генерації залежного значення

Для людей вхідні значення надходять від органів чуття. Шари є загальною темою в нейронних мережах, тому що, як і в людському мозку, один шар є відносно слабким, тоді як багато шарів є сильними. Те, що людина чує, відчуває запах, торкається, незалежно від того, що це може бути, обробляється як вхідний шар, а потім надсилається на вихід. Для цифрового нейрона незалежні значення (вхідні сигнали) проходять через нейрон, щоб згенерувати залежне значення (вихідний сигнал). Ці незалежні змінні в одному шарі – це просто ряд даних для одного спостереження. Наприклад, в контексті задачі нейронної мережі один вхідний шар буде означати одну змінну – можливо, вік або стать (незалежна змінна) людини, особу якої ми намагаємося з'ясувати (залежна змінна). Потім ця нейронна мережа застосовується стільки разів, скільки точок даних ми маємо на кожну незалежну змінну.

Вихідні значення можуть бути неперервними, бінарними або категоріальними змінними. Вони просто повинні відповідати одному рядку, який буде надсилатися нейрону як незалежні змінні. По суті, один тип незалежної змінної відповідає одному типу вихідної змінної. Ці вихідні змінні можуть бути однаковими для різних рядів, тоді як вхідні змінні не можуть бути однаковими [6].

Наступне, про що потрібно знати, це те, що відбувається в синапсах. Синапси – це лінії, що з'єднують нейрон з вхідними сигналами. Всім їм присвоюються ваги. Ваги мають вирішальне значення для штучних нейронних мереж, оскільки вони дозволяють мережам навчатися. Ваги вирішують, які вхідні сигнали не є важливими - які з них передаються далі, а які ні. На рисунку 1.2 зображено ваги, які визначають важливість вхідних даних.

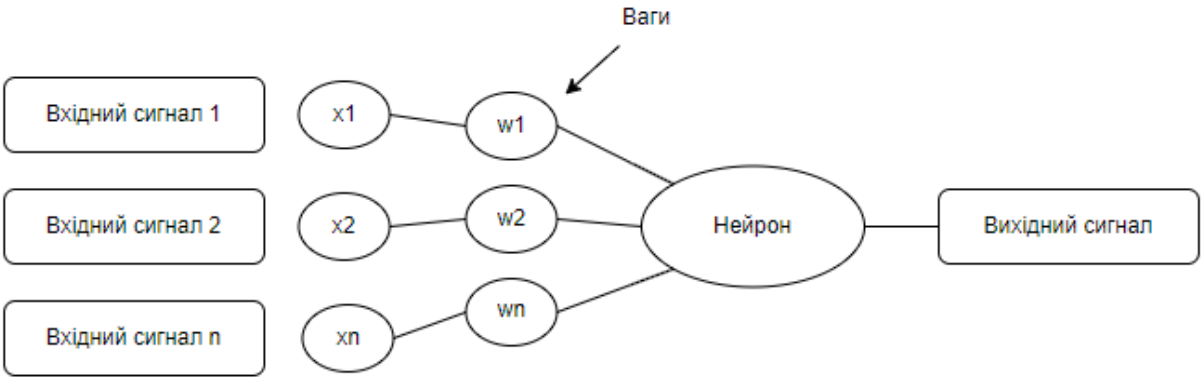


Рисунок 1.2 – Ваги визначають важливість вхідних даних

На першому етапі всі значення, що проходять через нього, підсумовуються. Іншими словами, він бере зважену суму всіх вхідних значень. Потім він застосовує функцію активації. Функції активації – це просто функції, що застосовуються до зваженої суми. Залежно від результату застосованої функції, нейрон або передасть сигнал, або не передасть його.

Більшість алгоритмів машинного навчання можуть бути виконані в такому вигляді, з масивом вхідних сигналів, що проходять через функцію активації (яка може бути будь-якою: логістична регресія, поліноміальна регресія і т.д.), і вихідним сигналом в кінці. На рисунку 1.3 зображено базовий вигляд нейронної мережі.

Робота нейронної мережі аналогічна функціонуванню нейронної системи людського мозку. У цій мережі термін "нейрон" вказує на математичну функцію, яка здійснює збір та класифікацію інформації, враховуючи конкретну архітектуру.

Нейронна мережа складається з шарів взаємопов'язаних вузлів. Кожен вузол, відомий як персептрон, подібний до множинної лінійної регресії. Сигнал, створений множинною лінійною регресією, передається у функцію активації, яка може мати нелінійний характер.

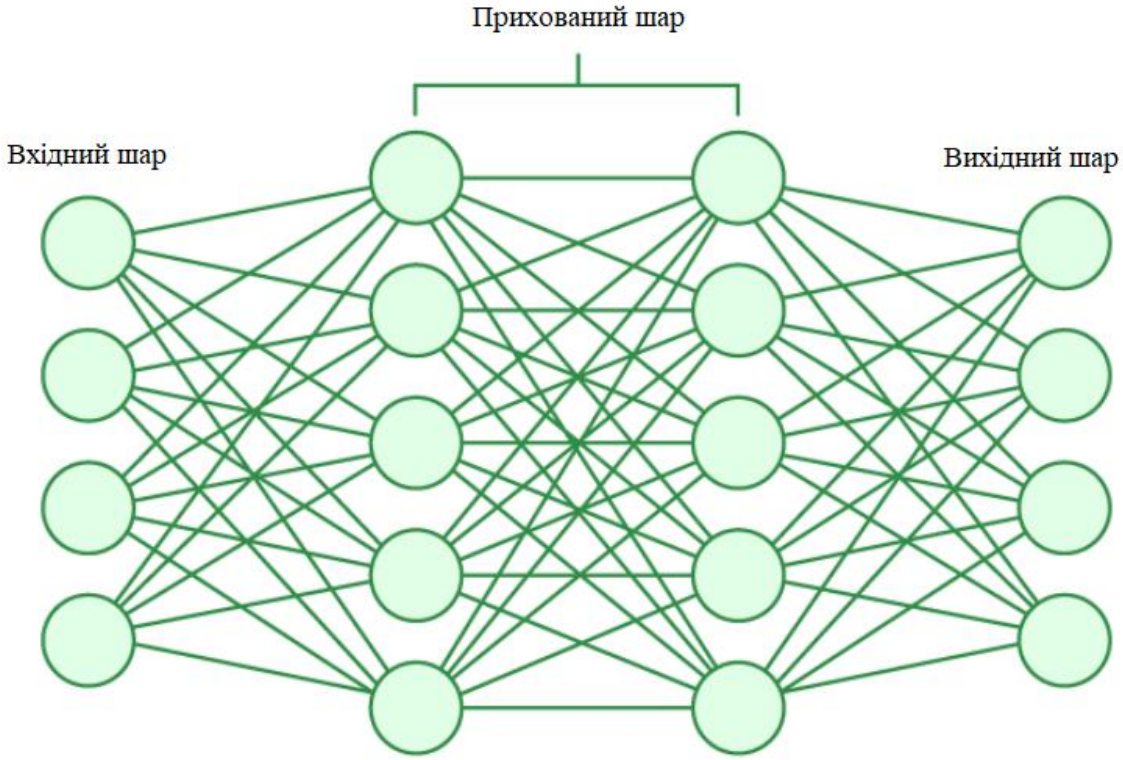


Рисунок 1.3 – Базовий вигляд нейронної мережі

Складові нейронної мережі:

– вхідний шар – кожна функція вхідного рівня представлена вузлом у мережі, який отримує вхідні дані;

– ваги та зв'язки – вага кожного нейронного з'єднання вказує на те, наскільки міцним є зв'язок (протягом тренування ці ваги змінюються);

– приховані шари – кожен нейрон прихованого шару обробляє вхідні дані, множачи їх на ваги, складаючи їх, а потім пропускаючи через функцію активації. Таким чином, вводиться нелінійність, що дозволяє мережі розпізнавати складні закономірності;

– вихідний шар – кінцевий результат, який досягається шляхом повторення процесу [7].

1.2 Ключові показники ефективності нейронної мережі

Прогнозування – це показник продуктивності, який кількісно оцінює точність моделі у створенні позитивних прогнозів. Вона визначається як відношення істинно позитивних прогнозів (правильно ідентифікованих позитивних випадків) до суми істинно позитивних і хибнопозитивних прогнозів (випадків, які були помилково ідентифіковані як позитивні).

Формула для розрахунку прогнозування виглядає так:

$$\text{Прогнозування} = \frac{\text{IC}}{\text{IC}+\text{XC}}, \tag{1.1}$$

де IC – кількість істинних спрацьовувань;

XC – кількість хибнопозитивних спрацьовувань.

Прогнозування важливе, коли ціна помилкових спрацьовувань висока або коли метою є мінімізація помилкових виявлень. Метрика вимірює частку правильних позитивних прогнозів. Це допомагає оцінити, наскільки добре модель розрізняє релевантні та нерелевантні об'єкти на аналізованих зображеннях. У задачах комп'ютерного зору, таких як виявлення об'єктів, сегментація зображень або розпізнавання облич, точність прогнозування дає цінну інформацію про здатність моделі правильно ідентифікувати та локалізувати цільові об'єкти або ознаки, мінімізуючи при цьому помилкові спрацьовування [8].

Відповідь, також відомий як чутливість або коефіцієнт істинних спрацьовувань, є ключовою метрикою в оцінці моделі комп'ютерного зору. Цей коефіцієнт визначається як частка істинно позитивних прогнозів (правильно ідентифікованих позитивних прикладів) серед усіх релевантних прикладів (сума істинних позитивних і хибнонегативних результатів, тобто позитивних прикладів, які модель не змогла ідентифікувати). Таким чином, формула для розрахунку показника має такий вигляд:

$$\text{Відповідь} = \frac{\text{ІП}}{\text{ІП+ХН}}, \tag{1.2}$$

де ІП – істинно позитивні прогнози;

ХН – хибнопозитивні результати.

Важливість цього показника полягає в тому, що він дозволяє виміряти здатність моделі виявляти всі позитивні випадки, що робить його критично важливим показником у ситуаціях, коли пропущені позитивні випадки можуть мати значні наслідки. Відповідь кількісно вимірює частку позитивних випадків, які модель успішно ідентифікувала. Це дає уявлення про ефективність моделі в захопленні повного набору релевантних об'єктів або ознак на проаналізованих зображеннях. Наприклад, в контексті системи безпеки, відповідь відображає частку реальних зловмисників, виявлених системою. Високе значення показника відповіді є бажаним, оскільки вказує на те, що система ефективно виявляє потенційні загрози безпеці, мінімізуючи ризик невиявлених вторгнень [9].

Оцінка F1 – це показник ефективності, який об'єднує прогнозування і відповіді в одне значення, забезпечуючи збалансовану оцінку продуктивності моделі комп'ютерного зору. Він визначається як середнє гармонійне значення прогнозування та відповіді, яке обчислюється наступним чином:

$$\text{Оцінка F1} = \frac{2 \cdot (\text{Прогнозування} \cdot \text{Відповідь})}{\text{Прогнозування} + \text{Відповідь}}. \tag{1.3}$$

Важливість показника F1 випливає з його корисності в сценаріях з нерівномірним розподілом класів або коли хибнопозитивні та хибнонегативні спрацьовування мають різну вартість. Враховуючи як прогнозування (точність позитивних прогнозів), так і відповідь (здатність ідентифікувати всі позитивні випадки), показник F1 дає комплексну оцінку продуктивності моделі, особливо коли баланс між хибнопозитивними і хибнонегативними результатами має вирішальне значення. Наприклад, у системі медичної візуалізації показник F1

допомагає визначити загальну ефективність моделі у виявленні та діагностиці певних станів. Високий показник F1 вказує на те, що модель успішно ідентифікує відповідні ознаки, мінімізуючи як помилкові спрацьовування (наприклад, здорова тканина помилково позначена як патологічна), так і помилкові несприятливі результати (наприклад, стан, який залишився невиявленим). У таких випадках показник F1 слугує цінною метрикою для забезпечення оптимальної роботи моделі комп'ютерного зору та мінімізації потенційних ризиків, пов'язаних з помилковою діагностикою або пропущеним діагнозом [10].

Точність – це фундаментальний показник продуктивності, який використовується для оцінки моделі комп'ютерного зору. Вона визначається як частка правильних прогнозів (як істинно-позитивних, так і істинно-негативних) серед усіх прикладів у заданому наборі даних. Іншими словами, він вимірює відсоток прикладів, які модель класифікувала правильно, враховуючи як позитивні, так і негативні класи. Це формула для розрахунку точності моделі:

$$\text{Точність} = \frac{\text{ІП}+\text{ІН}}{\text{ІП}+\text{ХП}+\text{ІН}+\text{ХН}}, \tag{1.4}$$

де ІП – істинно-позитивні прогнози;

ІН – істинно-негативні прогнози;

ХП – хибно-позитивні прогнози;

ХН – хибно-негативні прогнози.

Важливість точності зумовлена її здатністю надавати прямий вимір загальної продуктивності моделі. Вона дає загальне уявлення про те, наскільки добре модель виконує певне завдання, наприклад, виявлення об'єктів, класифікацію зображень або сегментацію. Однак точність може бути непридатною в ситуаціях зі значним дисбалансом класів, оскільки вона може дати хибне уявлення про роботу моделі. У таких випадках модель може добре працювати на більшості класів, але погано на класах меншості, що призводить до високої точності, яка не точно відображає ефективність моделі в ідентифікації

всіх класів. Наприклад, у системі класифікації зображень точність вказує на частку зображень, які модель класифікувала правильно. Високе значення точності свідчить про те, що модель ефективно призначає правильні мітки зображенням у всіх класах. Важливо враховувати й інші показники ефективності, такі як прогнозування, відповідь та оцінка F1, щоб отримати більш повне уявлення про роботу моделі. Це особливо важливо, коли ми маємо справу з незбалансованими наборами даних або сценаріями з різною вартістю різних типів помилок [11].

Середня абсолютна похибка – це метрика, яка використовується для вимірювання продуктивності моделей машинного навчання, наприклад, тих, що використовуються в комп'ютерному зорі, шляхом кількісної оцінки різниці між передбаченими значеннями і фактичними значеннями. САП є середнім значенням абсолютної різниці між прогнозованими та дійсними значеннями. САП обчислюється шляхом взяття абсолютної різниці між передбаченими та дійсними значеннями для кожної точки даних, а потім усереднення цієї різниці для всіх точок даних у наборі даних. Математично, формула для САП має вигляд:

$$\text{САП} = \frac{1}{\text{n}} \cdot \sum|\text{ПЗ} - \text{ІЗ}|\,, \qquad (1.5)$$

де n – кількість точок у наборі даних;

ПЗ – прогнозоване значення;

ІЗ – істинне значення.

САП допомагає оцінити точність моделі комп'ютерного зору, надаючи єдине значення, яке представляє середню помилку в прогнозах моделі. Нижчі значення САП вказують на кращу продуктивність моделі. Оскільки САП є абсолютною метрикою помилки, її легше інтерпретувати і розуміти порівняно з іншими метриками, такими як середня квадратична помилка (СКП). На відміну від СКП, яка зводить різниці в квадрат і надає більшої ваги більшим помилкам, САП трактує всі помилки однаково, що робить її більш стійкою до викидів даних.

Середню абсолютну похибку можна використовувати для порівняння різних моделей або алгоритмів і для точного налаштування гіперпараметрів. Мінімізуючи САП під час навчання, модель можна оптимізувати для кращої продуктивності на небачених даних [12].

1.3 Різниця між машинним та глибоким навчанням

Штучний інтелект (ШІ) – одна з найстаріших галузей комп'ютерних наук, яка охоплює різні аспекти імітації когнітивних функцій для вирішення реальних проблем і побудови комп'ютерних систем, які навчаються і мислять, як люди. Відповідно, ШІ часто називають машинним інтелектом, щоб протиставити його людському інтелекту. Сфера ШІ виникла на перетині комп'ютерних та когнітивних наук. ШІ може стосуватися будь-чого – від комп'ютерної програми, що грає партію в шахи, до самокерованих автомобілів і систем комп'ютерного зору. Завдяки успіхам у МН, ШІ зараз викликає величезний інтерес. Штучний інтелект, і зокрема машинне навчання, – це здатність машини постійно покращувати свою продуктивність без необхідності пояснювати людині, як саме виконувати всі поставлені перед нею завдання. За останні кілька років машинне навчання стало набагато ефективнішим і доступнішим. Тепер ми можемо створювати системи, які навчаються виконувати завдання самостійно [13]. На рисунку 1.4 зображено галузі штучного інтелекту.

Рисунок 1.4 – Галузі штучного інтелекту

Машинне навчання – це підгалузь штучного інтелекту. Основний принцип машинного навчання полягає в тому, що машина використовує дані, щоб вчитися на їх основі. Таким чином, системи машинного навчання можуть швидко застосовувати знання і навчання з великих масивів даних, щоб досягти успіху в розпізнаванні людей, розпізнаванні мови, виявленні об'єктів, перекладі та багатьох інших завданнях. На відміну від розробки та кодування програми з конкретними інструкціями для виконання завдання, МН дозволяє системі навчитися самостійно розпізнавати закономірності та робити прогнози. Машинне навчання – це дуже практична галузь штучного інтелекту, метою якої є розробка програмного забезпечення, яке може автоматично навчатися на основі попередніх даних, щоб отримувати знання з досвіду і поступово покращувати свою навчальну поведінку, щоб робити прогнози на основі нових даних.

Незважаючи на те, що машинне навчання є підгалуззю ШІ, терміни ШІ і МН часто використовуються як взаємозамінні. Машинне навчання отримує набір вхідних даних, а потім навчається на основі цих даних. Таким чином, методи машинного навчання використовують дані для розуміння контексту, осмислення і прийняття рішень в умовах невизначеності. У складі систем штучного інтелекту алгоритми машинного навчання зазвичай використовуються для виявлення тенденцій і розпізнавання закономірностей у даних. На рисунку 1.5 зображено типи навчальних стилів для алгоритмів машинного навчання.

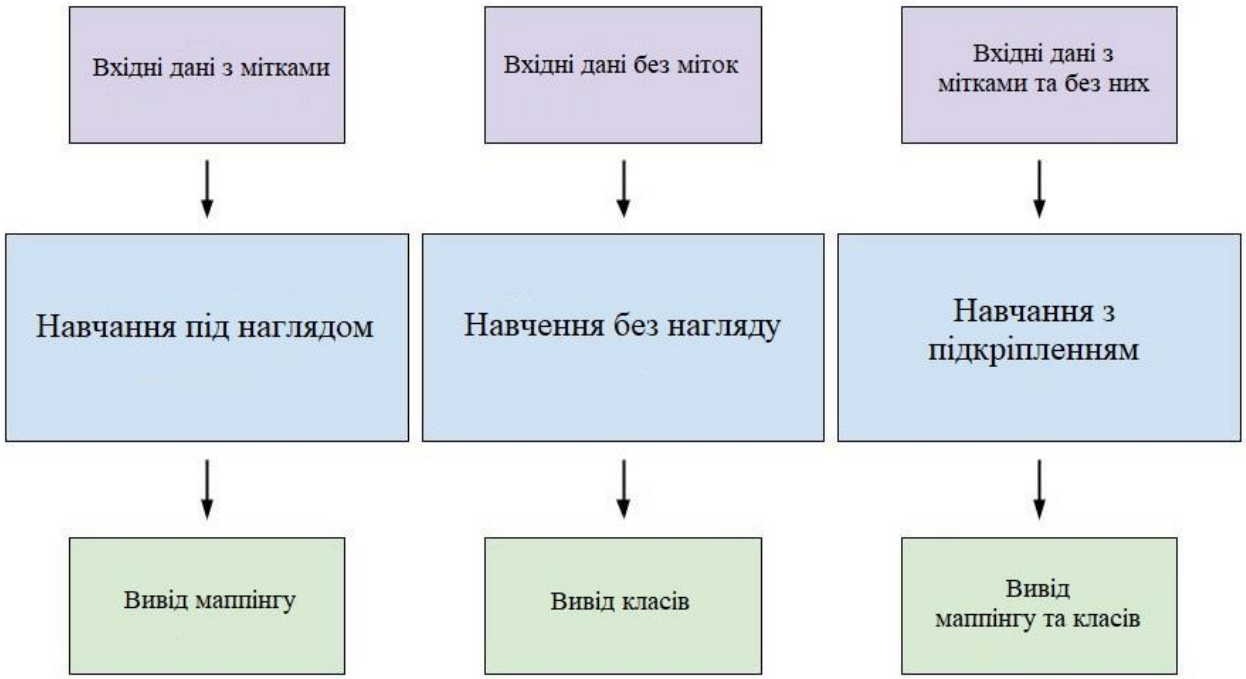


Рисунок 1.5 – Типи навчальних стилів для алгоритмів машинного навчання

Застосування машинного навчання можна знайти скрізь – у науці, інженерії та бізнесі, що сприяє прийняттю більш обґрунтованих рішень. За допомогою машинного навчання створюються різні автоматизовані рекомендаційні системи зі штучним інтелектом. Прикладом машинного навчання є персоналізовані рекомендації щодо фільмів від Netflix або музичні рекомендації від сервісів потокової передачі музики на вимогу. Величезний прогрес у машинному навчанні

був зумовлений розробкою нових статистичних алгоритмів навчання, а також наявністю біг-даних (великих масивів даних) і недорогих обчислень [14].

Глибоке навчання (Deep Learning, DL) є надзвичайно популярним методом машинного навчання. Це сімейство моделей, побудованих на основі глибоких нейронних мереж, які представляють собою підмножину загального області машинного навчання. Глибоке навчання використовує різні методи машинного навчання для вирішення реальних завдань, застосовуючи нейронні мережі для імітації процесу прийняття рішень людиною. Таким чином, йому вдається навчати машину виконувати завдання, аналогічні тим, як природно працює людський мозок. Основною характеристикою глибокого навчання є його багаторівнева структура, що базується на штучних нейронних мережах. Кожен рівень додає нові знання до тих, які були отримані на попередньому рівні. Проте важливо враховувати, що глибоке навчання може бути витратним, вимагаючи значних обчислювальних ресурсів і великих наборів даних для ефективного тренування моделей. Також відзначається тим, що для правильної роботи алгоритму навчання глибокого навчання необхідно визначити велику кількість параметрів, іноді це може призводити до появи помилкових відповідей на етапі навчання.

Глибоке навчання визначає алгоритми, які аналізують дані за логічною структурою, аналогічно тому, як це робить людина, утворюючи висновки. Важливо відзначити, що цей процес може включати як контрольоване, так і неконтрольоване навчання. Для досягнення цієї мети алгоритми глибокого навчання використовують багаторівневу структуру, відому як штучна нейронна мережа. Дизайн такої штучної нейронної мережі взятий за зразок з біологічної нейронної мережі людського мозку, що призводить до навчання, яке виявляється значно ефективнішим, ніж у стандартних моделях машинного навчання.

Наприклад, можна навчити алгоритм глибокого навчання розпізнавати образ собаки, аналізуючи його найдрібніші деталі, щоб відрізнити собаку від інших тварин, таких як лисиця чи пантера. Для досягнення цієї точності необхіден значний обсяг даних у вигляді зображень. Загалом глибоке навчання

виявляється особливо ефективним у створенні систем штучного інтелекту, особливо в контексті комп'ютерного зору, що надає можливість створювати системи, що найбільш схожі на людську спроможність. Ще одним комерційним застосуванням глибокого навчання є візуальне розпізнавання облич, яке використовується для захисту та розблокування мобільних телефонів. Глибоке навчання також знаходить практичне використання у сфері бізнесу, де великі обсяги даних, такі як мільйони зображень, використовуються для розпізнавання конкретних характеристик. Завдання, такі як пошук на основі тексту, виявлення шахрайства, розпізнавання кадрів, рукописного тексту і шаблонів, пошук зображень та розпізнавання облич, можуть бути успішно вирішені за допомогою глибокого навчання. Крупні компанії, що працюють у галузі штучного інтелекту, такі як Meta/Facebook, IBM або Google, використовують глибоке навчання для автоматизації систем, що раніше вимагали ручної обробки. Завдяки автоматизованому створенню функцій та можливостям самонавчання, алгоритми глибокого навчання вимагають лише мінімального втручання людини. Незважаючи на великий потенціал глибокого навчання, існують дві основні причини, що пояснюють, чому цей підхід недавно став настільки зручним у використанні: доступність даних та обчислювальна потужність.

Машинне навчання і глибоке навчання входять у сферу штучного інтелекту, причому глибоке навчання є підмножиною машинного навчання. Таким чином, хоча глибоке навчання входить у машинне навчання, воно відрізняється від традиційних методів цієї галузі. Глибоке навчання володіє конкретними перевагами над іншими формами машинного навчання, що робить його найбільш важливою технологією в сфері алгоритмів сьогодення. Машинне навчання використовує алгоритми, ефективність яких поліпшується зі збільшенням обсягу даних. З іншого боку, глибоке навчання покладається на шари для здійснення навчання, відмінно від машинного навчання, яке опирається на вхідні дані для самонавчання [15].

На рисунку 1.6 зображено різницю машинного та глибокого навчання.

Рисунок 1.6 – Різниця машинного та глибокого навчання

Хоча як машинне навчання, так і глибоке навчання впроваджують процеси вчителювання для навчання комп'ютерів на основі наявних даних, їх методи навчання виявляються відмінними. Як машинне навчання, так і глибоке навчання надають різні результати через відмінності у своїх процесах навчання. При цьому глибоке навчання виражається в підтримці масштабованості, регульованому і нерегульованому навчанні, а також нашаруванні знань. Ці характеристики роблять глибоке навчання однією з найпотужніших "наук про моделювання" для навчання машин. На рисунку 1.7 зображено процес машинного та глибокого навчання.

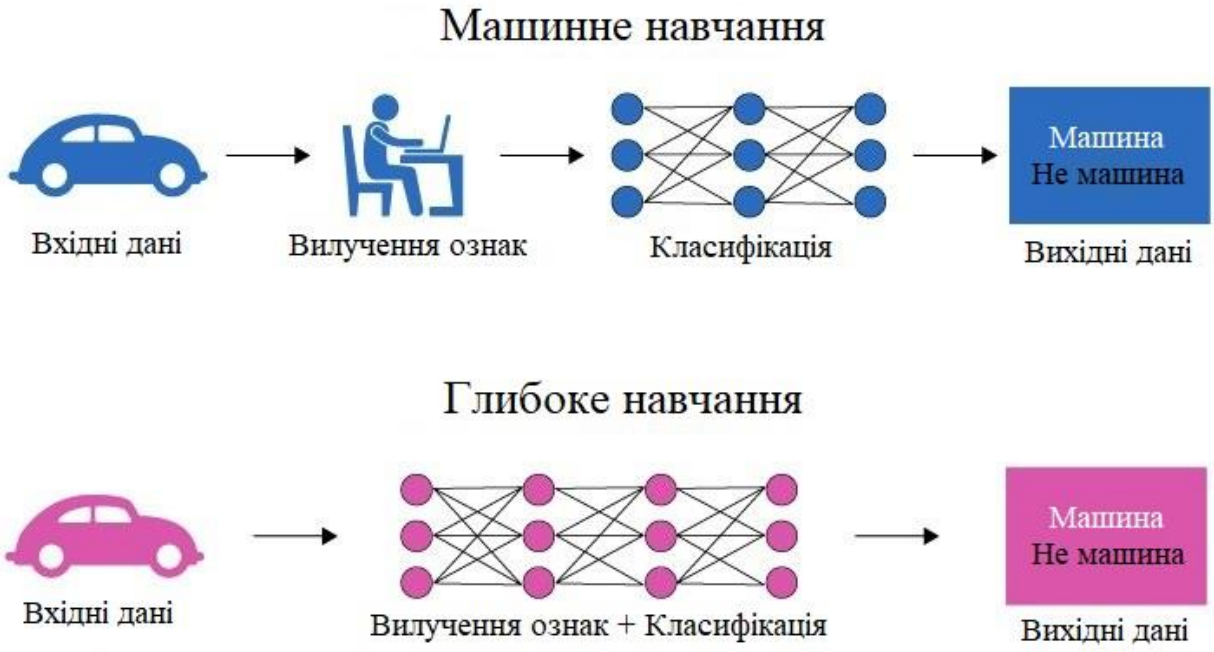


Рисунок 1.7 – Процес машинного та глибокого навчання

Розвиток глибокого навчання швидко прискорився завдяки використанню нейронних мереж та доступності високопродуктивних комп'ютерів. На противагу цьому, інші традиційні форми МН досягли "плато в продуктивності". Відмінності машинного та глибокого навчання:

– навчання – машинне навчання забезпечує швидке тренування моделі на основі даних, де більше даних призводить до поліпшення результатів. У той час як глибоке навчання вимагає значних обчислень для тренування нейронних мереж із численними шарами;

– продуктивність – використання нейронних мереж і доступність більш потужних комп'ютерів прискорили розвиток глибокого навчання, а на противагу цьому, інші форми МН ніяк не поліпшувались;

– ручне втручання – кожен раз, коли в машинному навчанні проводиться нове навчання, розробник повинен втручатися та налаштовувати алгоритм для успішного тренування. На відміну від цього, в глибокому навчанні нейронні мережі сприяють багаторівневому навчанню, де розумні алгоритми можуть

передавати знання від одного шару до наступного, дозволяючи машині навчатися без постійного втручання людини;

– навчання – у традиційному машинному навчанні розробник вказує машині, які ознаки слід враховувати. У глибокому навчанні процес вилучення особливостей повністю автоматизований. Як результат, вилучення ознак у глибинному навчанні стає точнішим і орієнтованим на досягнення результату. Методи машинного навчання вимагають постановки задачі, яка розкладає проблему на окремі частини для вирішення, після чого об'єднання результатів на завершальному етапі. З іншого боку, методи глибокого навчання зазвичай вирішують проблему від початку до кінця, що робить процес навчання ефективнішим і більш надійним;

– дані – для глибокого навчання нейронної мережі використовують багаторівневі знання без втручання людини, де потрібна велика кількість даних для ефективного навчання, а машинне навчання ґрунтується на керованому вивченні патернів даних, які, хоча все ще значні, проте порівняно менші;

– точність – в порівнянні з машинним навчанням, здатність глибокого навчання до автоматичного самонавчання призводить до отримання результатів швидше та точніше. У традиційному машинному навчанні помилки розробника можуть призвести до неправильних рішень і низької точності, що робить машинне навчання менш гнучким у порівнянні з глибоким навчанням;

– обчислювальна техніка – глибоке навчання вимагає високопродуктивних машин, на відміну від традиційних алгоритмів машинного навчання.

На рисунку 1.8 зображено порівняння машинного та глибокого навчання.

Рисунок 1.8 – Порівняння машинного та глибокого навчання

Машинне навчання зазвичай не є оптимальним рішенням для вирішення вкрай складних завдань, наприклад, у сфері комп'ютерного зору, де система імітує людське бачення та аналізує зображення, враховуючи їх особливості. Глибоке навчання виробляє реалістичні результати для комп'ютерного зору завдяки дуже точній архітектурі нейронних мереж, яку не знайдеш у традиційному машинному навчанні. Без глибокого навчання комп'ютерне бачення не досягло б такого рівня точності, як сьогодні [16].

1.4 Типи глибоких нейронних мереж

Модель багатошарового персептрону входить у категорію штучних нейронних мереж зі зворотнім зв'язком. БШП є простою формою глибоких нейронних мереж і складається з послідовних повністю з'єднаних шарів. Сучасні

методи машинного навчання дозволяють використовувати БШП для подолання високих вимог до обчислювальної потужності, які характерні для сучасних архітектур глибокого навчання. Кожен новий шар у БШП представляє собою набір нелінійних функцій зваженої суми всіх виходів (повністю пов'язаних) попереднього шару [17]. На рисунку 1.9 зображено багатошаровий персептрон.

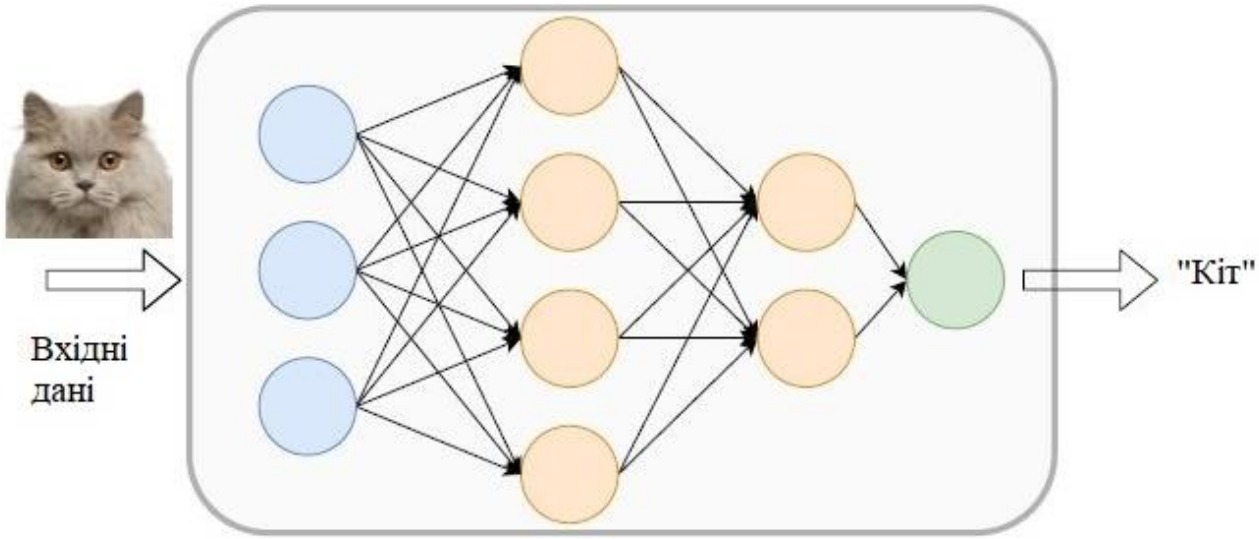


Рисунок 1.9 – Багатошаровий персептрон

Глибокий клас нейронних мереж, відомий як згорткова нейронна мережа, широко використовується в області комп'ютерного зору. За допомогою ЗНМ, опрацьовуючи серії зображень або відео з реального світу, система штучного інтелекту автоматично навчається виділяти особливості цих вхідних даних для виконання різних завдань, таких як класифікація зображень, розпізнавання облич та семантична сегментація зображень. На відміну від повністю пов'язаних шарів у багатошаровому персептроні, у ЗНМ один або кілька шарів згортки визначають прості ознаки вхідних даних, виконуючи операції згортки. Кожен шар представляє собою набір нелінійних функцій зваженої суми за різними координатами просторово близьких підмножин виходів з попереднього шару, що

сприяє повторному використанню ваг [18]. На рисунку 1.10 зображено згорткову нейронну мережу.

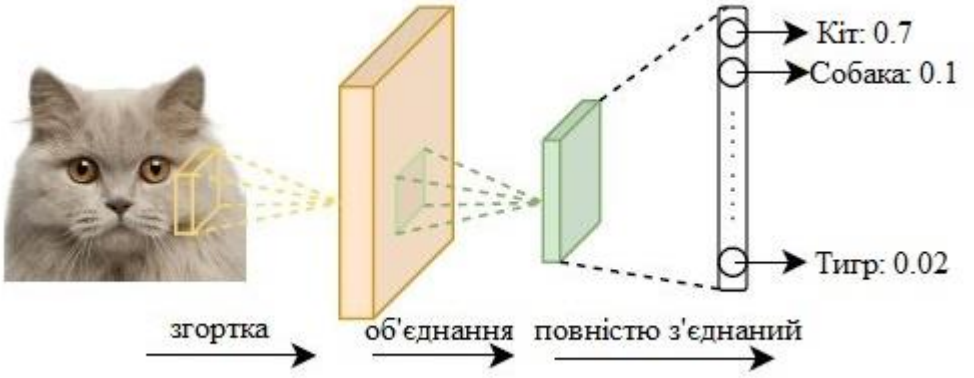


Рисунок 1.10 – Згорткова нейронна мережа

Шляхом використання різноманітних згорткових фільтрів, моделі згорткових нейронних мереж в галузі машинного навчання можуть відображати високорівневі представлення вхідних даних, що робить методи ЗНМ дуже популярними для завдань комп'ютерного зору. Приклади використання цих нейронних мереж включають в себе класифікацію зображень (наприклад, AlexNet, VGG network, ResNet, MobileNet) та виявлення об'єктів (наприклад, Fast R-CNN, Mask R-CNN, YOLO, SSD):

– AlexNet – використовується для класифікації зображень, стала першою нейронною мережею ЗНМ, що перемогла у конкурсі ImageNet Challenge у 2012 році, складається з п'яти шарів згортки і трьох повністю пов'язаних шарів. Таким чином, для класифікації зображення розміром 227×227, AlexNet використовує 61 мільйон вагових коефіцієнтів та виконує 724 мільйони операцій у форматі MAC (multiply-add computation);

– VGG-16 – для досягнення більшої точності VGG-16 була навчена з використанням глибшої структури, що складається з 16 шарів, із них 13 є згортковими, а три – повністю з'єднаними. Цей підхід вимагає 138 мільйонів ваг

та обчислює 15,5 ГБ у форматі MAC для класифікації зображення розміром 224×224;

– GoogleNet – для підвищення точності та одночасного зменшення обсягу обчислень у висновках глибокої нейронної мережі, GoogleNet використовує початковий модуль, що включає фільтри різного розміру. Внаслідок цього GoogleNet досяга кращих результатів точності порівняно з VGG-16, при цьому для обробки зображення того ж розміру використовується лише сім мільйонів ваг та 1,43 ГБ операцій у форматі MAC;

– ResNet – як остання передова розробка, застосовує концепцію "shortcut" (дозволяє створити коротший шлях для передачі інформації вздовж мережі) блоку для досягнення точності на рівні людського виконання, з частотою помилок у топ-5 менше 5%. Крім того, використання "shortcut" модуля дозволяє вирішувати проблему зникнення градієнта під час навчання, що робить можливим навчання штучних нейронних мереж з більш глибокою структурою.

На рисунку 1.11 зображено продуктивність глибоких нейронних мереж на ImageNet.

Рекурентна нейронна мережа – це тип нейронної мережі, яка призначена для роботи з послідовностями даних. РНМ володіє здатністю запам'ятовувати попередні інформаційні стани та використовувати їх для обробки нових вхідних даних. Одна з основних характеристик РНМ полягає у взаємозв'язку між її внутрішніми частинами, що дозволяє їй працювати з даними залежності від контексту.

Основні компоненти РНМ включають:

– рекурентний шар (Recurrent Layer) – це основна частина мережі, яка містить рекурентні нейрони. Кожен нейрон у цьому шарі приймає вхід та попередній стан, виробляючи новий стан, який передається наступному часовому кроці;

– стан (Hidden State) РНМ зберігає внутрішній стан, який представляє собою попередню інформацію, отриману з попередніх кроків вхідної

послідовності. Цей стан використовується для обробки нових даних та зберігає контекст інформації;

– функція активації, як і в інших нейронних мережах, РНМ використовує функції активації для вироблення вихідних значень та створення нелінійності в моделі.

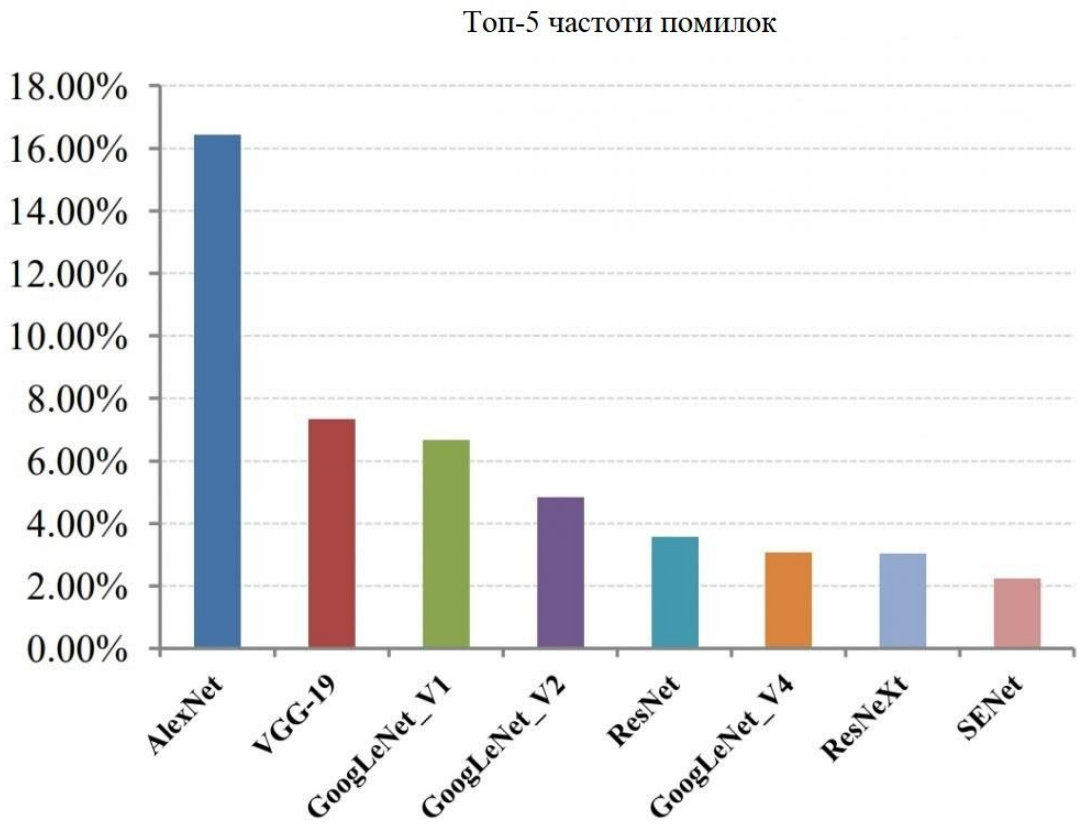


Рисунок 1.11 – Продуктивність глибоких нейронних мереж на ImageNet

Рекурентна нейронна мережа належить до категорії штучних нейронних мереж і використовує послідовний підхід до подачі даних. Цей тип нейронних мереж був створений для вирішення проблеми обробки часових рядів послідовних вхідних даних. Вхідні дані РНМ включають поточний вхід і попередні вибірки. Таким чином, між вузлами утворюється напрямлений граф вздовж часової послідовності. Кожен нейрон в РНМ також має внутрішню пам'ять, яка зберігає інформацію про обчислення з попередніх вибірок.

Моделі рекурентних нейронних мереж широко використовуються в обробці природної мови (NLP), особливо через їхню ефективність у роботі з даними нефіксованої довжини вхідних даних. Однією з основних задач штучного інтелекту в цьому контексті є створення систем, які здатні розуміти природну мову, включаючи моделювання мови, вбудовування слів і машинний переклад.

На рисунку 1.12 зображено рекурентну нейронну мережу.

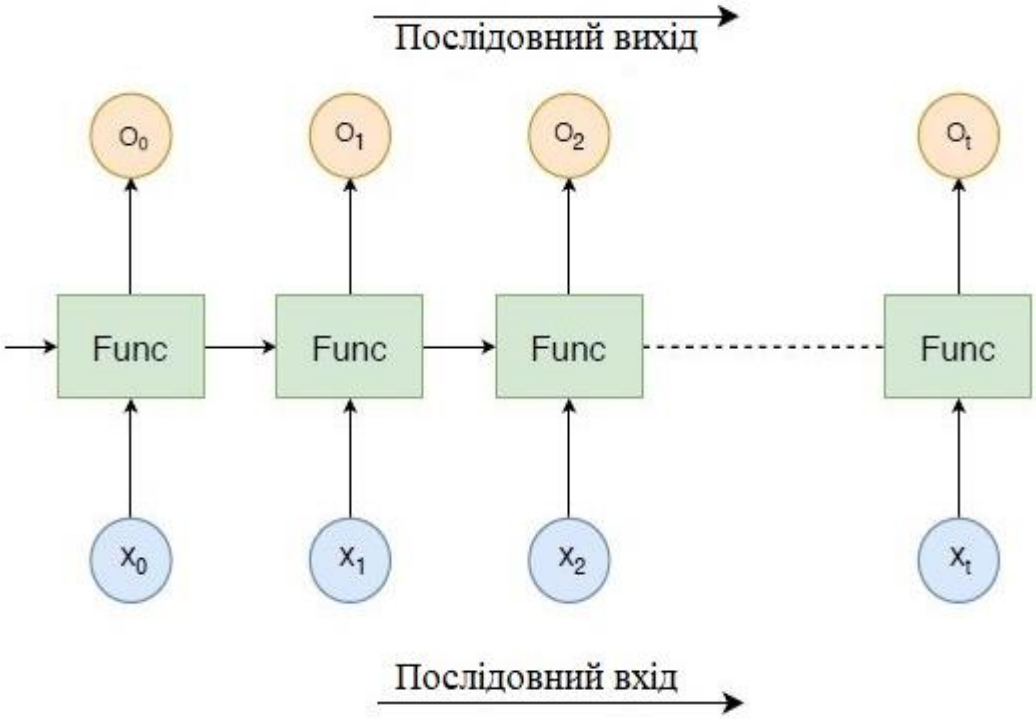


Рисунок 1.12 – Рекурентна нейронна мережа

У структурі РНМ кожен наступний шар представляє собою набір нелінійних функцій зважених сум виходів і попереднього стану. Основною одиницею RNN є "комірка", при цьому кожна комірка складається з шарів і послідовної групи комірок, що дозволяє послідовно обробляти рекурентні нейромережеві моделі [17].

1.5 Комп’ютерний зір

Комп'ютерний зір – це галузь штучного інтелекту (ШІ), яка дозволяє комп'ютерам і системам отримувати важливу інформацію з цифрових зображень, відео та інших візуальних даних, а також вживати заходів або давати рекомендації на основі цієї інформації. Якщо штучний інтелект дозволяє комп'ютерам думати, то комп'ютерний зір дозволяє їм бачити, спостерігати і розуміти. Комп'ютерний зір потребує великої кількості даних. Він аналізує дані знову і знову, доки не помітить відмінності та зрештою не розпізнає зображення. Наприклад, щоб навчити комп'ютер розпізнавати автомобільні шини, йому потрібно подати величезну кількість зображень шин і предметів, пов'язаних з шинами, щоб вивчити відмінності та розпізнати шину, особливо без дефектів [19].

Завдання комп’ютерного зору:

– класифікація зображень – нейронна мережа бачить зображення і класифікує його (собака, яблуко, обличчя людини). Вона здатна точно передбачити, що дане зображення належить до певного класу. Наприклад, компанія, що займається соціальними мережами, може використовувати її для автоматичного виявлення та відокремлення неприйнятних зображень, завантажених користувачами;

– виявлення об'єктів – може використовувати класифікацію зображень для ідентифікації певного класу зображень, а потім виявляти і фіксувати їхню появу на зображенні або відео. Приклади включають виявлення пошкоджень на конвеєрі або ідентифікацію машин, які потребують технічного обслуговування;

– відстеження об'єктів – це стеження за об'єктом після його виявлення. Це завдання часто виконується за допомогою зображень, знятих послідовно, або відеопотоків у реальному часі. Наприклад, автономні транспортні засоби повинні не тільки класифікувати і виявляти такі об'єкти, як пішоходи, інші автомобілі та об'єкти дорожньої інфраструктури, але й відстежувати їх у русі, щоб уникнути зіткнень і дотримуватися правил дорожнього руху;

– пошук зображень на основі вмісту – використовує комп'ютерний зір для перегляду, пошуку і вилучення зображень з великих сховищ даних на основі вмісту зображень, а не тегів метаданих, пов'язаних з ними. Це завдання може включати автоматичну анотацію зображень, яка замінює ручне тегування зображень. Ці завдання можуть бути використані в системах управління цифровими ресурсами і можуть підвищити точність пошуку і вилучення [20].

1.6 Аналіз алгоритму R-CNN

R-CNN був запропонований Россом Гіршиком та ін. у 2014 році для вирішення проблеми ефективної локалізації об'єктів при виявленні об'єктів. Попередні методи використовують вичерпний пошук, який використовує рухомі вікна різного масштабу на зображенні, щоб запропонувати пропозиції регіонів. Натомість запропонований метод використовує алгоритм вибіркового пошуку, який використовує переваги сегментації об'єктів та вичерпного пошуку для ефективного визначення пропозицій регіонів. Вибірковий пошук використовується для визначення областей на зображенні, які слід захопити. Спочатку визначаються невеликі області. Потім подібні області об'єднуються для створення нових більших областей. Цей процес повторюється багаторазово, на кожному кроці створюються більші області, проте об'єкти на зображенні фактично згруповані. Цей алгоритм вибіркового пошуку пропонує приблизно 2000 пропозицій регіонів для кожного зображення. Потім вони передаються в модель ЗНМ (тут використовується AlexNet) [21].

AlexNet – це згорткова нейронна мережа, яка складається з декількох шарів згорткових та об'єднувальних шарів, за якими слідують повністю з'єднані шари. Архітектура включає п'ять згорткових шарів, три об'єднувальні шари і три повністю з'єднані шари.

Перші два згорткові шари використовують ядро розміром 11×11 і застосовують 96 фільтрів до вхідного зображення. Третій і четвертий згорткові шари використовують ядро розміром 5×5 і застосовують 256 фільтрів. П'ятий

згортковий шар використовує ядро розміром 3×3 і застосовує 384 фільтри. Вихідні дані цих згорткових шарів потім пропускаються через шари максимального об'єднання, які зменшують просторові розміри карт об'єктів.

Вихідні дані об'єднаних шарів пропускаються через три повністю з'єднані шари з 4096, 4096 і 1000 нейронів відповідно. Останній повністю з'єднаний шар використовується для класифікації і виробляє розподіл ймовірностей для 1000 класів ImageNet.

AlexNet був навчений на наборі даних ImageNet, який складається з 1,2 мільйона зображень з 1000 класів, і зміг досягти високої точності розпізнавання [22].

На рисунку 1.13 зображено архітектуру R-CNN.

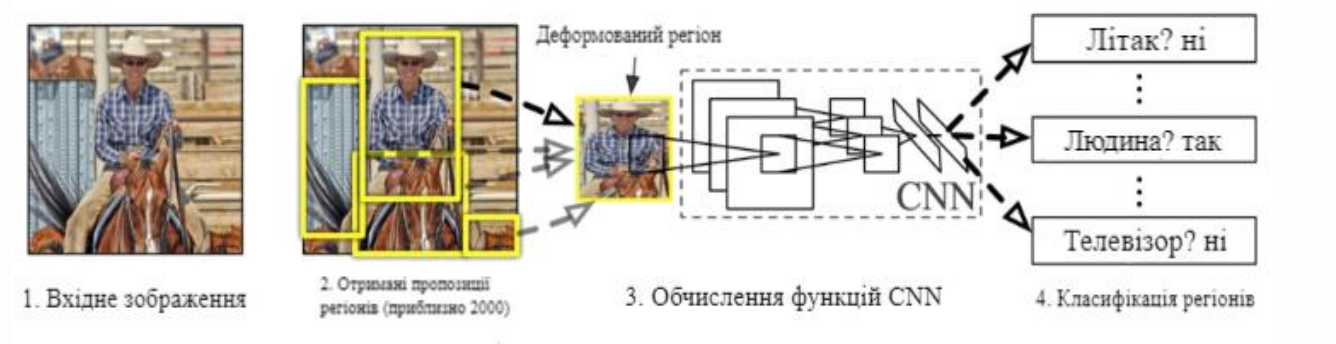


Рисунок 1.13 – Архітектура R-CNN

Проблеми R-CNN:

– навчання мережі все ще займає величезну кількість часу, оскільки доведеться класифікувати 2000 пропозицій регіонів на одне зображення;

– не може бути реалізований в реальному часі, оскільки для кожного тестового зображення потрібно близько 47 секунд;

– алгоритм вибіркового пошуку є фіксованим алгоритмом, а тому на цьому етапі навчання не відбувається (може призвести до генерації поганих пропозицій регіонів-кандидатів) [23].

1.7 Аналіз алгоритму Fast R-CNN

В R-CNN кожна пропозиція регіону пропускається по черзі в архітектуру ЗНМ і вибірковий пошук генерує близько 2000 пропозицій регіонів для зображення. Навчання і навіть тестування зображення за допомогою R-CNN є дуже дорогим з точки зору обчислень. Для вирішення цієї проблеми було запропоновано Fast R-CNN, який використовує ціле зображення та пропозиції регіонів як вхідні дані у своїй архітектурі ЗНМ. Він також поєднує різні частини архітектури (такі як ConvNet, RoI pooling та рівень класифікації) в одну повну архітектуру. Це також усуває вимогу зберігати карту функцій і заощаджує місце на диску. Крім того, для класифікації пропозицій регіонів використовується шар softmax замість SVM, який є швидшим і дає кращу точність [24].

На рисунку 1.14 зображено архітектуру Fast R-CNN.

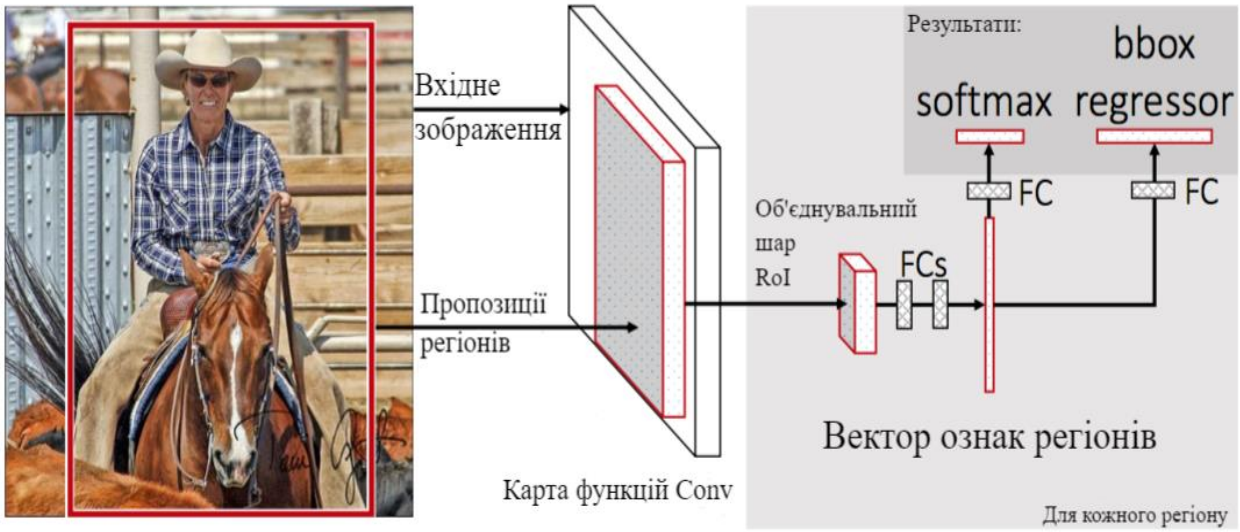


Рисунок 1.14 – Архітектура Fast R-CNN

Все зображення подається на модель ЗНМ для визначення пропозицій регіонів на картах об'єктів. Кожна область відокремлюється за допомогою об’єднувального шару RoI і подається на повністю з'єднані шари (FC). Цей

вектор використовується класифікатором softmax для виявлення об'єкта і лінійним регресором для зміни координат обмежувальної рамки.

Швидкий R-CNN значно покращує час навчання (8,75 години проти 84 годин) і час виявлення в порівнянні з R-CNN. Він також дещо покращує середню точність виявлення об’єктів порівняно з R-CNN.

Проблема Fast R-CNN полягає в тому, що більша частина часу, яку Fast R-CNN витрачає на виявлення, припадає на алгоритм генерації пропозицій вибіркової області пошуку [22].

1.8 Аналіз алгоритму Faster R-CNN

Faster R-CNN був представлений у 2015 році. В алгоритмі Fast R-CNN проблемним місцем архітектури був вибірковий пошук. Оскільки йому потрібно згенерувати 2000 пропозицій на одне зображення. Це становить більшу частину часу навчання всієї архітектури. У Faster R-CNN вона була замінена на мережу регіональних пропозицій. Перш за все, в цій мережі зображення передається в магістральну мережу. Ця магістральна мережа генерує згорнуту карту особливостей. Ці карти особливостей потім передаються в мережу пропозицій регіонів. Мережа пропозицій регіонів бере карту особливостей і генерує опорні точки (центр рухомого вікна з унікальним розміром і масштабом). Потім ці опорні точки передаються до шару класифікації (який класифікує наявність чи відсутність об'єкта) та шару регресії (який визначає обмежувальну рамку, пов'язану з об'єктом).

Faster R-CNN дозволяє мережі безпосередньо вивчати пропозиції регіонів та використовує цю мережу для прогнозування пропозицій регіону замість алгоритму вибіркового пошуку.

На рисунку 1.15 зображено архітектуру Faster R-CNN.

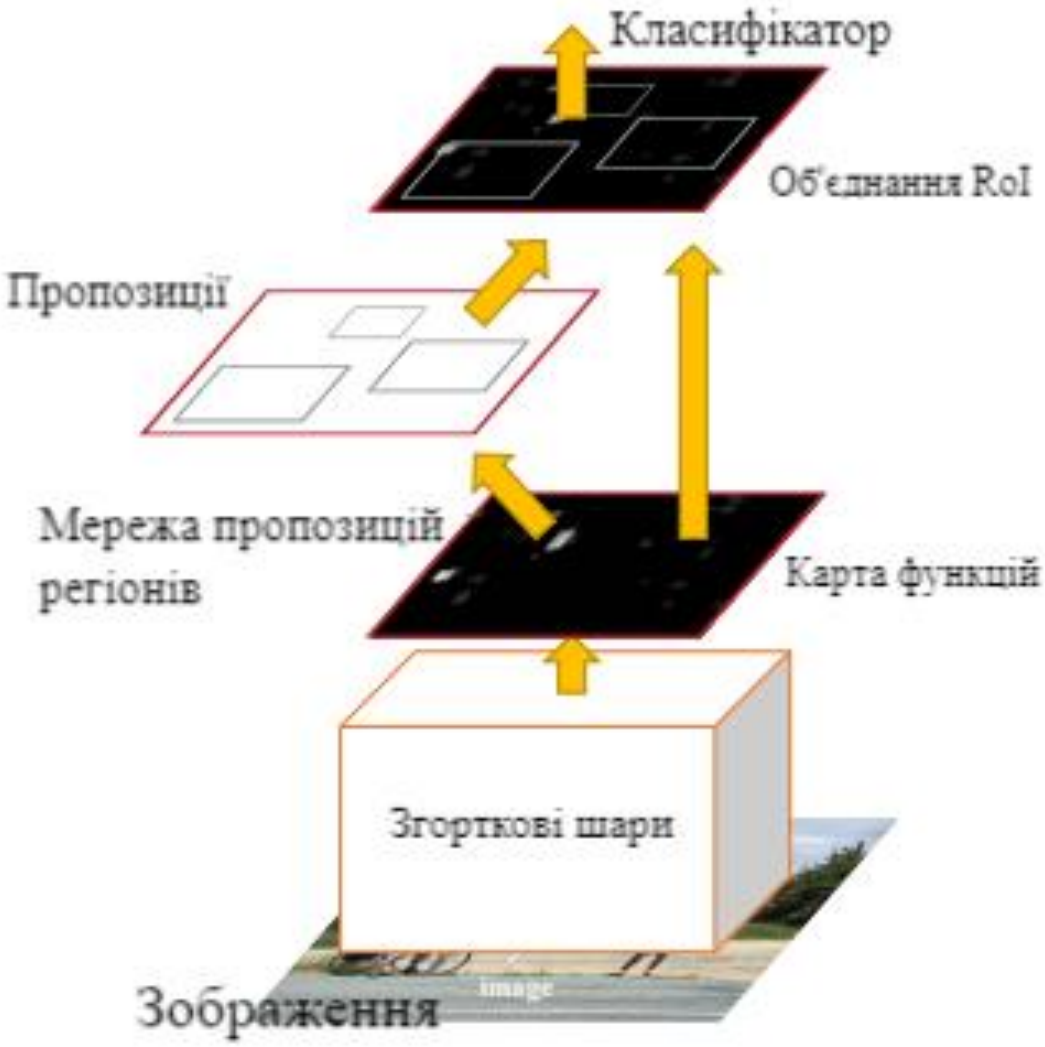


Рисунок 1.15 – Архітектура Faster R-CNN

Основна проблема Faster R-CNN в тому, що алгоритм визначає лише обмежувальні рамки, але не семантичну сегментацію (послідовність) [25].

1.9 Аналіз алгоритму YOLO

Вперше алгоритм був представлений Joseph Redmon у 2016 році і з тих пір пройшов кілька ітерацій, останньою з яких була YOLO v8.

YOLO (You Only Look Once) – це алгоритм виявлення об'єктів, який значно відрізняється від алгоритмів, що базуються на регіонах, розглянутих вище. У YOLO єдина згорткова мережа прогнозує обмежувальні рамки та ймовірності

класів для цих рамок.

На рисунку 1.16 зображено принцип роботи YOLO.

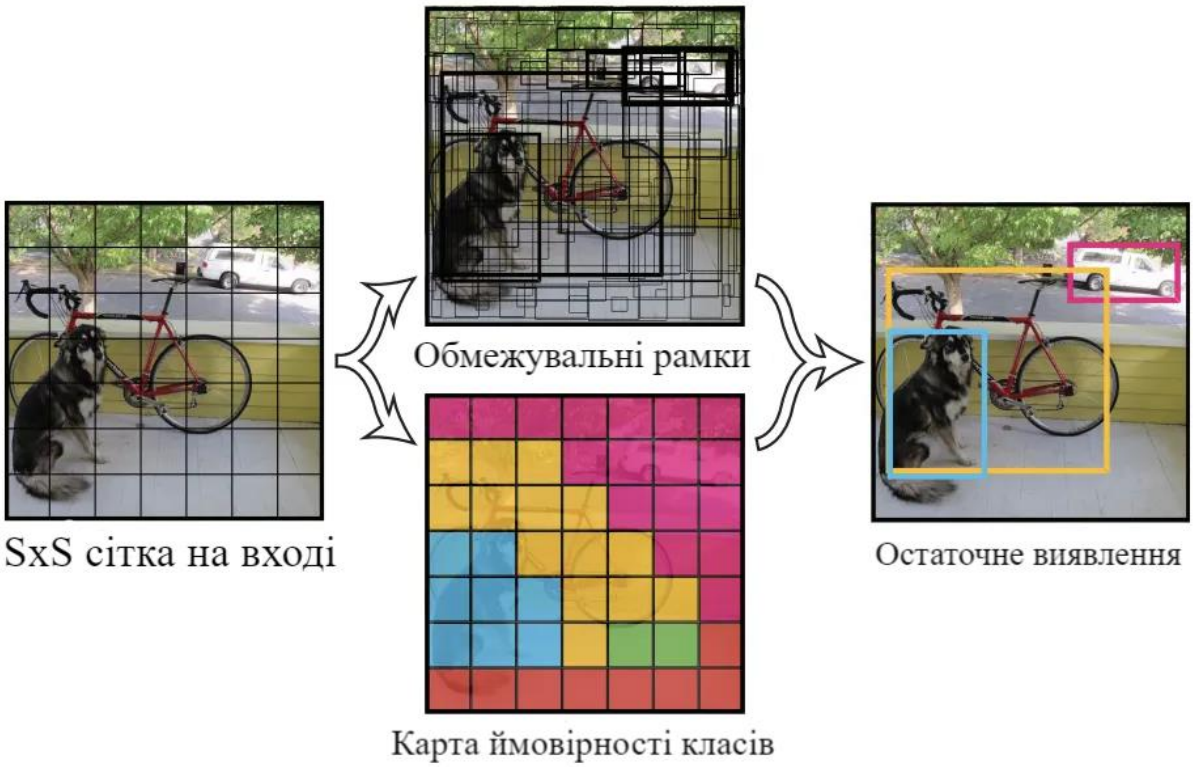


Рисунок 1.16 – Принцип роботи YOLO

Принцип роботи YOLO полягає в тому, що ми беремо зображення і розбиваємо його на сітку SxS, в кожній з яких ми виділяємо m обмежувальних рамок. Для кожного з них мережа виводить ймовірність класу та значення зсуву для нього. Обмеження, ймовірність класу яких перевищує порогове значення, обираються і використовуються для визначення місцезнаходження об'єкта на зображенні.

YOLO працює на порядок швидше (45 кадрів за секунду), ніж інші алгоритми виявлення об'єктів. Обмеженням алгоритму YOLO є те, що він бореться з невеликими об'єктами на зображенні, наприклад, у нього можуть виникнути труднощі з виявленням зграї птахів. Це пов'язано з просторовими обмеженнями алгоритму [3].

1.10 Висновки до розділу

В межах даного розділу було проаналізовано:

– нейронна мережа, як дані потрапляють до нейрону, яким чином розподіляється важливість цих даних та базовий вигляд нейронної мережі;

– критерії оцінки нейронної мережі, за допомогою яких можна дізнатися наскільки продуктивною є мережа, а також наскільки точно виконує поставленні завдання;

– методи навчання нейронних мереж, їх особливості та відмінності;

– типи нейронних мереж, особливості їх роботи;

– алгоритми виявлення об’єктів, а саме R-CNN, Fast R-CNN, Faster R-CNN та YOLO, їх переваги та недоліки.

2 РОЗРОБЛЕННЯ СТРУКТУРНОЇ СХЕМИ ТА АЛГОРИТМУ РОБОТИ ПІДСИСТЕМИ ВИЯВЛЕННЯ ВИБУХОНЕБЕЗПЕЧНИХ ПРЕДМЕТІВ ДЛЯ БАГАТОФУНКЦІОНАЛЬНОЇ РОБОТИЗОВАНОЇ ПЛАТФОРМИ

2.1 Розроблення структурної схеми підсистеми виявлення вибухонебезпечних предметів

Для підсистеми виявлення вибухонебезпечних предметів потрібно враховувати, що роботизована платформа повинна мати змогу підключення Wi-Fi модуля та камери.

На рисунку 2.1 зображено структурну схему підсистеми виявлення вибухонебезпечних предметів.

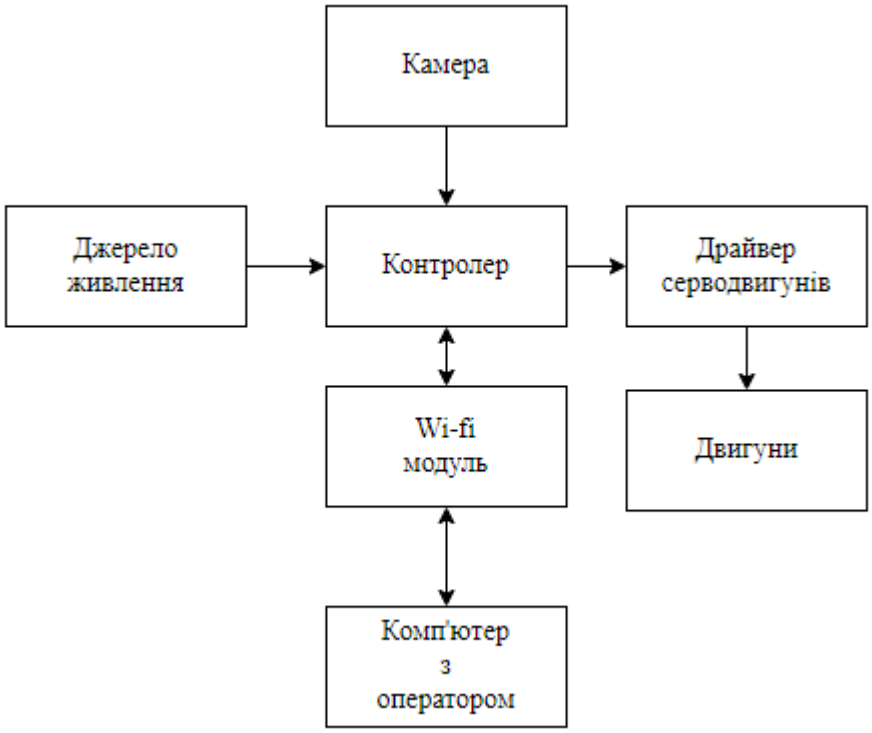


Рисунок 2.1 – Структурна схема підсистеми виявлення вибухонебезпечних предметів

Зі схеми можна побачити, що зображення з камери передаються через Wi-fi модуль оператору на комп’ютер, в цей час підсистема обробляє зображення та виявляє вибухонебезпечні об’єкти за допомогою нейронної мережі.

2.2 Вибір компонентів підсистеми виявлення вибухонебезпечних предметів

Одним із ключових елементів розробки підсистеми виявлення вибухонебезпечних предметів для роботизованої системи є вибір компонентів, а саме контролер, камера, Wi-Fi модуль, двигуни та комп’ютер.

Серед контролерів, які підходять для підсистеми виявлення вибухонебезпечних предметів було обрано Raspberry Pi 3 Model B+.

Raspberry Pi 3 Model B+ – це повноцінний одноплатний комп’ютер, який має невеликі габарити. Базується на SoC (системі на чіпі) BCM2837B0, яка включає в себе 4-ядерний ARMv8 64-бітний процесор з робочою частотою 1,4 ГГц та потужний відеопроцесор VideoCore IV. З головних переваг має вбудовані Wi-Fi та Bluetooth модулі, а також використання камери з розширенням 1080p30.

На рисунку 2.2 зображено вигляд контролера Raspberry Pi 3 Model B+.

Рисунок 2.2 – Контролер Raspberry Pi 3 Model B+ [26]

Контролер Raspberry Pi 3 Model B+ має наступні характеристики:

– чіп Broadcom BCM2837B0 (CPU, GPU, DSP and RAM);

– процесор 64-бітний 4-ядерний ARMv8 Cortex-A53 з тактовою частотою 1,4 ГГц;

– відеопроцесор VideoCore IV 3D;

– пам’ять 1 ГБ LPDDR2 з частотою 900 МГц;

– роз’єми цифровий HDMI, аналоговий 3,5 мм jack (4 pin) та USB 2.0 (4 шт.);

– накопичувач microSD;

– порт дисплея DSI (Displey Serial Interface);

– порт камери CSI-2 (MIPI Camera Serial Interface);

– роз’єм введення-виведення GPIO 40 pin;

– бездроторі інтерфейси Wi-Fi (2,4 ГГц та 5 ГГц) та Bluetooth 4.2.

Також треба пам’ятати, що Raspberry Pi живиться напругою 5 В, а його логіка працює на рівні 3,3 В [26].

В підсистемі виявлення вибухонебезпечних предметів камера один з найважливіших компонентів. Камера повинна мати змогу запису відео з роздільною здатністю не менше ніж 1080p30. Вища роздільна здатність буде значним поліпшенням виявлення вибухонебезпечних предметів, але також збільшить вартість.

Raspberry Pi Camera Module v2 представляє собою високоякісний 8 мегапіксельний датчик зображення Sony IMX219, який включає в себе фіксований фокус, розроблений спеціально для плат Raspberry Pi. Цей датчик має підтримку зображення з роздільною здатністю 3280x2464 пікселів, а також запис відео у форматах 1080p30 та 720p60. Підключається до плати завдяки невеликим роз’ємам на верхній частині плати і використовує особливий CSI інтерфейс, який було спроєктовано спеціально для роботи з камерами.

На рисунку 2.3 зображено камеру Raspberry Pi Camera Module v2.

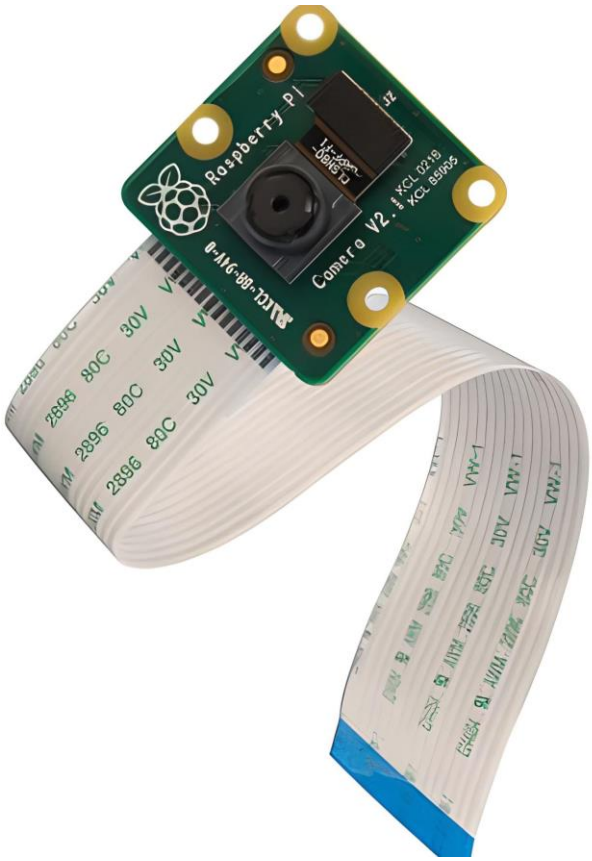


Рисунок 2.3 – Камера Raspberry Pi Camera Module v2 [27]

Габарити камери невеликі, всього 25 мм х 23 мм х 9 мм, а вага близько 3 г, що ідеально підходить для мобільної роботизованої платформи. Підключення камери до плати відбувається за допомогою стрічкового кабелю [27].

Щоб оператор мав змогу керувати мобільною платформою та отримувати відео, де підсистема буде виявляти вибухонебезпечні предмети, потрібно обрати комп’ютер для цих завдань, а отже мінімальні характеристики повинні бути такі:

– процесор Intel Celeron N3350 з частотою 2,4 ГГц;

– відеокарта Intel HD Graphics 500;

– обсяг оперативної пам’яті 4 ГБ;

– екран з роздільною здатністю 1920x1080.

Такі характеристики дозволять використовувати комп’ютер одночасно для керування мобільною роботизованою платформою та для роботи підсистеми виявлення вибухонебезпечних об’єктів.

DC Gearbox Motor "TT Motor" - 200RPM - 3 to 6VDC (Adafruit TT Motor) -

це мотор з редуктором, призначений для робототехнічних застосувань і проектів DIY (Do It Yourself). Він має наступні характеристики:

– номінальна напруга від 3 В до 6 В;

– безперервний струм холостого ходу 150 мА (струм, який споживається мотором, коли він обертається без навантаження);

– мінімальна робоча швидкість при напрузі 3 В 90 обертів за хвилину (це мінімальна швидкість обертання мотора при заданій напрузі);

– мінімальна робоча швидкість при напрузі 6 В 200 обертів за хвилину (це мінімальна швидкість обертання мотора при заданій напрузі);

– розміри корпусу 70 мм х 22 мм х 18 мм (фізичні розміри мотору);

– довжина проводів, які виходять з мотора для підключення живлення та управління 200 мм;

– вага мотора 30,6 г [28].

На рисунку 2.4 зображено мотор DC Gearbox Motor.

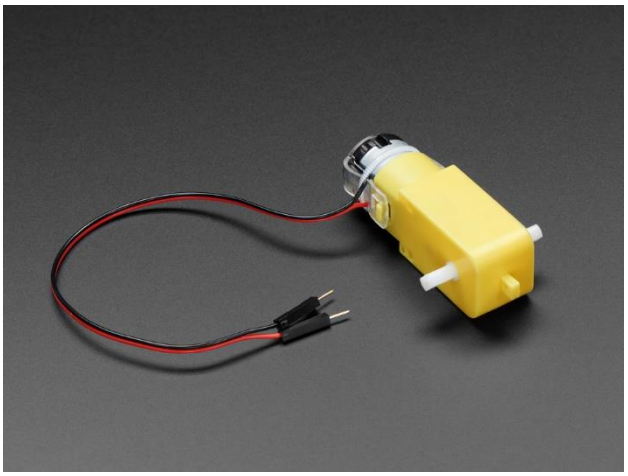


Рисунок 2.4 – Мотор DC Gearbox Motor [28]

Для автономної роботи мобільної роботизованої платформи необхідно використовувати акумулятор, вихідна напруга якого буде становити 5 В. Anker PowerCore акумулятор має наступні характеристики:

– ємність батареї 10000 мА · год;

– вихідна напруга 5 В;

– вихідна сила струму 2,4 А;

– максимальна вихідна потужність 12 Вт;

– розміри 153 мм х 72 мм х 15,5 мм;

– вага 240 г [29].

Вигляд акумулятора Anker PowerCore зображено на рисунку 2.5.

Рисунок 2.5 – Акумулятор Anker PowerCore [29]

Тепер, коли визначено основні компоненти підсистеми виявлення вибухонебезпечних предметів для мобільної роботизованої платформи, можна визначити час автономної роботи. Для цього необхідно розрахувати ефективну потужність з акумулятора [30]:

$$P_{total} = P_{Pi} + P_{camera} + n \cdot P_{servo}, \quad (2.1)$$

де P_{Pi} – потужність контролера;

P_{camera} – потужність камери;

n – кількість двигунів;

P_{servo} – потужність двигуна.

Розрахувати ефективну енергію акумулятора:

$$\mathrm{E_{battery}} = \text{Ємність батареї} \cdot \text{Напруга батареї}. \quad (2.2)$$

Після чого розрахувати час роботи:

$$\mathrm{T} = \frac{\mathrm{E_{battery}}}{\mathrm{P_{total}}}. \quad (2.3)$$

З відомих характеристик обраних компонентів розраховується автономний час роботи мобільної роботизованої платформи, спочатку необхідно розрахувати ефективну потужність акумулятора:

$$\mathrm{P_{total}} = 4 + 2 + 4 \cdot 1 = 10 \text{ Вт}.$$

Наступним кроком треба розрахувати ефективну енергію акумулятора:

$$\mathrm{E_{battery}} = 10 \cdot 5 = 50 \text{ Вт} \cdot \text{год}.$$

Тепер, коли відомо потрібні коефіцієнти, можна розрахувати час роботи акумулятора:

$$\mathrm{T} = \frac{50}{10} = 5 \text{ год}.$$

З урахування обраних компонентів підсистеми виявлення вибухонебезпечних предметів для мобільної роботизованої платформи час автономної роботи від джерела живлення становить близько 5 годин.

2.3 Розроблення алгоритму роботи виявлення вибухонебезпечних предметів для багатофункціональної роботизованої платформи

Перед початком розроблення підсистеми виявлення вибухонебезпечних предметів, необхідно розробити алгоритм роботи. Це допоможе в подальшій реалізації цієї підсистеми.

Розроблений алгоритм роботи виявлення вибухонебезпечних предметів зображено на рисунку 2.4.

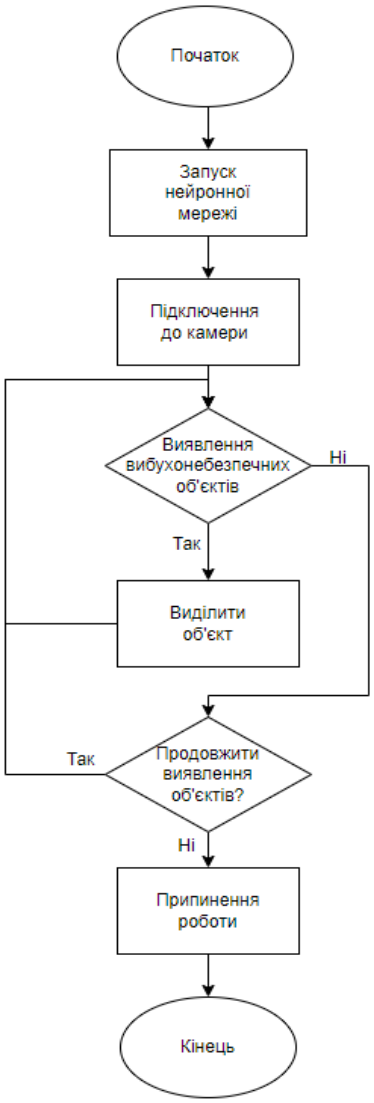


Рисунок 2.4 – Розроблений алгоритм роботи підсистеми виявлення вибухонебезпечних предметів

Робота підсистеми виявлення вибухонебезпечних предметів працює в декілька етапів:

– на першому етапі відбувається запуск нейронної мережі оператором;

– другий етап – підключення нейронної мережі до камери;

– третім етап є обробка вхідного зображення з камери та виявлення вибухонебезпечних предметів. YOLOv8 за допомогою ЗНМ розбиває ціле зображення на сітку, прогнозує обмежувальні рамки та ймовірності класів (на основі даних навчання). При потраплянні вибухонебезпечного предмету в поле зору камери, цей предмет, одразу виділяється обмежувальною рамкою, яку фіксує оператор. Нейронна мережа зберігає результати тільки при обробці відео або зображення, але не в реальному часі;

– четвертим етап – припинення роботи підсистеми. Третій етап зациклений, тому виявлення вибухонебезпечних предметів буде відбуватися до вимкнення підсистеми.

2.4 Висновки до розділу

У цьому розділі було побудовано структурну схему та обрано компоненти підсистеми виявлення предметів для мобільної роботизованої платформи, а саме контролер Raspberry Pi 3 Model B+, камеру Raspberry Pi Camera Module v2, двигуни DC Gearbox Motor, джерело живлення Anker PowerCore та мінімальні вимогу до характеристик комп’ютера, з якого буде відбуватися керування мобільною платформою та оброблюватись зображення підсистемою виявлення вибухонебезпечних предметів. Обрані компоненти є недорогими, а їх характеристики дають змогу для реалізації підсистеми виявлення вибухонебезпечних предметів.

Розраховано час автономної роботи, з урахуванням обраних компонентів, підсистеми виявлення вибухонебезпечних предметів для мобільної роботизованої платформи. Час автономної роботи складає 5 годин.

Розроблений алгоритм роботи підсистеми виявлення вибухонебезпечних

предметів наглядно демонструє принцип роботи підсистеми. Також додано поетапне пояснення роботи підсистеми виявлення вибухонебезпечних предметів. Оператору залишається лише зосередитись на керуванні мобільною роботизованою платформою, а підсистема виявлення вибухонебезпечних предметів, за допомогою нейронної мережі, розпізнає небезпечний об’єкт та виділить його на екрані.

3 НАВЧАННЯ НЕЙРОННОЇ МЕРЕЖІ ДЛЯ ВИЯВЛЕННЯ ВИБУХОНЕБЕЗПЕЧНИХ ПРЕДМЕТІВ

3.1 Вибір інструментів та розробка алгоритму процесу навчання підсистеми виявлення вибухонебезпечних предметів

Для тренування мережі використовувалась мова програмування Python, яка володіє зрозумілим синтаксисом, є відкритою та оптимально підходить як для початківців, так і для досвідчених інженерів.

Python є широко прийнятою високорівневою інтерпретованою мовою програмування. Вона дозволяє програмістам використовувати різні стилі програмування для створення як простих, так і складних програм, отримувати швидкі результати і писати код майже так, ніби спілкуються людською мовою. Python широко використовується в розробці різних систем і додатків, таких як пошук Google, YouTube, BitTorrent, Google App Engine, Eve Online, Maya та iRobot. Це популярна мова програмування, яка володіє багатьма бібліотеками та фреймворками для навчання нейронних мереж [31].

Серед алгоритмів виявлення об’єктів, з урахуванням переваг та недоліків, кращим вибором є YOLO з кількох причин:

– швидкість виявлення – YOLO визначає об’єкти в одному проході мережі, що дозволяє йому працювати дуже швидко, що особливо важливо, коли потрібно виявляти об’єкти в реальному часі;

– обробка зображення – опрацьовує повне зображення, а не фрагменти або регіони, що дозволяє взаємодіяти з контекстом усього сценарію. Це може бути корисним для виявлення вибухонебезпечних об’єктів, оскільки контекст і взаємодія з іншими об’єктами важливі для правильної ідентифікації;

– масштабність об’єктів – приділяє увагу об’єктам різних розмірів і масштабів, що дозволяє виявляти вибухонебезпечні об’єкти незалежно від їхнього розміру чи положення на зображені;

– висока точність виявлення – сучасні версії YOLO, такі як YOLOv7 або YOLOv8, мають покращену точність виявлення порівняно з попередніми версіями. Це робить їх ефективними для виявлення навіть малих об’єктів, що може бути важливим для виявлення маленьких вибухонебезпечних предметів;

– простота інтеграції – має прості конфігурації та є легким у використанні, що робить його привабливим для інтеграції в різні системи, включаючи виявлення вибухонебезпечних об’єктів [3].

Для нейронної мережі виявлення вибухонебезпечних об’єктів обрано YOLOv8. Це потужна модель для виявлення об'єктів на зображеннях та відео. Вона входить в родину YOLO-моделей і представлена командою Ultralytics. YOLOv8 поєднує в собі високу швидкість та точність виявлення об'єктів, роблячи її ефективною для різноманітних завдань у комп'ютерному зорі.

Ця модель використовує передові архітектурні рішення, зокрема залишкові блоки та концепцію "shortcut connections", що полегшує тренування глибоких мереж і покращує їхню ефективність. YOLOv8 є швидкою та придатною для використання в реальному часі, що робить її популярною в області відеоспостереження, автономних систем і в інших сценаріях.

Модель підтримує різні архітектури та може бути тренована на власних даних, що надає користувачам гнучкість і можливість адаптації до різних завдань виявлення об'єктів. Завдяки своїм характеристикам, YOLOv8 стає потужним інструментом для вирішення задач реального світу, пов'язаних із виявленням та розпізнаванням об'єктів на зображеннях та відео [32].

Для розробки використовувалося інтегроване середовище PyCharm.

PyCharm є інтегрованим середовищем розробки, спеціально призначеним для мови програмування Python. Воно надає інструменти для аналізу коду, графічного зневаджувача, запуску модульних тестів та підтримки веб-розробки. Для налагодження і тестування використовується вбудований графічний відладчик, а також можливість роботи з командним рядком. Використання профайлера Python допомагає виявляти неоптимальні фрагменти коду, які виконуються довго, і оптимізувати їх.

PyCharm також підтримує роботу з віддаленими інтерпретаторами та надає інтегровані засоби для створення та конфігурування віртуальних середовищ розробки через Vagrant. Крім того, воно підтримує розробку в середовищах з використанням контейнеризації Docker.

PyCharm дозволяє перевіряти правильність роботи окремих ділянок коду, модулів та процедур, що значно полегшує процес локалізації помилок в програмному коді, особливо у великих проектах з численними рядками коду [33].

Для навчання нейронної мережі важливим є анотації зображень, серед існуючих інструментів для анотації було обрано Computer Vision Annotation Tool (CVAT).

CVAT – це відкрите програмне забезпечення для анотації даних у галузі комп'ютерного зору. CVAT дозволяє анотувати зображення та відео для тренування моделей машинного навчання, забезпечуючи можливість спільної роботи анотаторів, підтримку різних форматів даних, імпорт та експорт анотацій, а також використання різних типів завдань, включаючи анотацію об'єктів, класифікацію та трасування. CVAT є важливим інструментом для підготовки даних у галузі комп'ютерного зору та сприяє автоматизації процесу анотації.

Основні характеристики CVAT включають:

– анотація різних типів об'єктів – це дозволяє користувачам анотувати об'єкти різних класів, такі як прямокутники, полігони, ключові точки та інші, що робить його гнучким для різних завдань виявлення об'єктів;

– підтримка різних форматів даних надає можливість імпорту та експорту анотацій у різні формати, що дозволяє легко обмінюватися даними між різними системами;

– візуалізація результатів відбувається завдяки зручному інтерфейсу та інструментам візуалізації, що дозволяють зручно переглядати та редагувати анотації [34].

Для кращого розуміння процесу навчання нейронної мережі було розроблено алгоритм, який зображено на рисунку 3.1.

Рисунок 3.1 – Алгоритм процесу навчання нейронної мережі

З цього алгоритму можна зробити висновок, що перший кроком у навчанні нейронної мережі є пошук зображень. Зібрані зображення необхідно поділити на дві групи, а саме тренувальну (80% всього набору даних) та валідаційну (20%

всього набору даних). Анотувати весь набір даних та розподілити в окремі теки, разом з анотаціями. За допомогою середовища розробки запустити процес навчання та очікувати результати. Після отримання результатів можна перевіряти працездатність навченої нейронної мережі.

3.2 Реалізація підсистеми виявлення вибухонебезпечних предметів для багатофункціональної роботизованої платформи у вигляді програмного засобу

Файлову структуру проєкту зображено на рисунку 3.2.

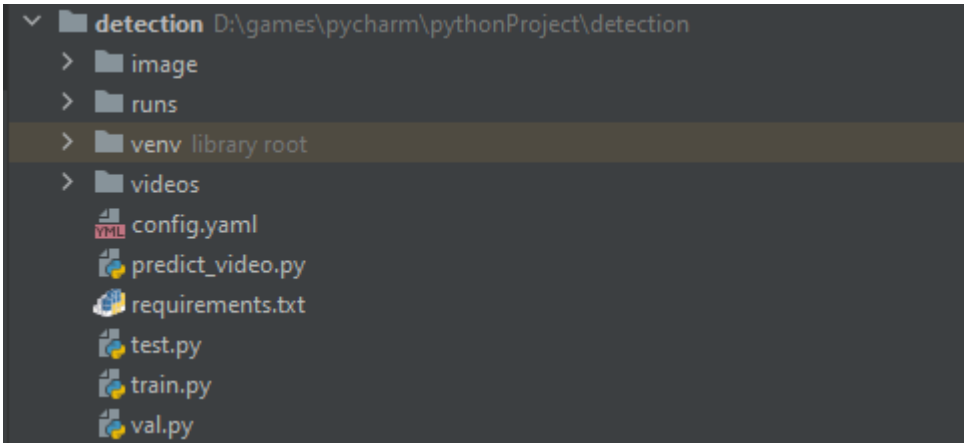


Рисунок 3.2 – Файлова структура проєкту

У структурі проєкту зберігаються усі файли, які створюються і виконуються під час навчання нейронної мережі. На початку треба встановити бібліотеку ultralytics, яка надає можливість користуватися вже готовими моделями YOLO або навчати власну. Для цього було створено файл requirements.txt в який прописується назва бібліотеки і середовище розробки завантажує її.

Щоб навчати нейронну мережу треба створити теку, де будуть зберігатися зображення на яких буде навчатися мережа та зображення на яких вона буде проходити перевірку. Для цього створюється тека під назвою «data», потім в ній треба створити ще дві теки «train» та «val». Після чого в кожній було створено

теки «images» та «labels». Наступний крок пошук зображень для навчання нейронної мережі та додавання анотацій для цих зображень. Інструментом анотації зображень було обрано CVAT. Процес анотації було зображено на рисунку 3.3.

Рисунок 3.3 – Анотація зображень

Цей процес треба зробити для всіх зображень на яких буде навчатися нейронна мережа. Після чого CVAT надає можливість експорту даних для потрібної моделі мережі, ці дані містять координати, які вказують мережі де саме на зображені знаходиться потрібний об’єкт. Зображення треба помістити в теку «images», а файли, які було експортовано, в теку «labels».

Далі створюється файл «config.yaml» в якому вказується шлях розташування матеріалу, який мережа буде використовувати для навчання, та вказуються класи об’єктів розпізнавання. Вигляд файлу «config.yaml» зображено на рисунку 3.4.

```yaml
path: D:\Object detection\yolo_train_custom_model\core\data  # dataset
train: train/images  # train images
val: val/images # val images


# Classes
names:
  0: grenade
```

Рисунок 3.4 – Вигляд файлу «config.yaml»

Тепер, як були підготовлені дані для навчання, необхідно створити файл «train.py» за допомогою якого і буде проводитися навчання. Для цього імпортуємо з бібліотеки ultralytics мережу YOLO. Далі завантажуємо пусту модель і прописуємо параметри навчання мережі. Вигляд файлу «train.py» зображено на рисунку 3.5.

```python
from ultralytics import YOLO

# Load a model
model = YOLO("yolov8n.yaml")  # build a new model

# Use the model
results = model.train(data="config.yaml", epochs=200, patience=0)  # train the model
```

Рисунок 3.5 – Вигляд файлу «train.py»

Після запуску цього файлу почнеться процес навчання. Навчання цієї моделі відбувалося на 405 зображеннях циклом в сто епох, що тривало чотири години. Як процес було завершено автоматично створюється тека runs в яку зберігаються результати тренування, валідації та передбачення. Загальний вигляд теки runs зображено на рисунку 3.6.

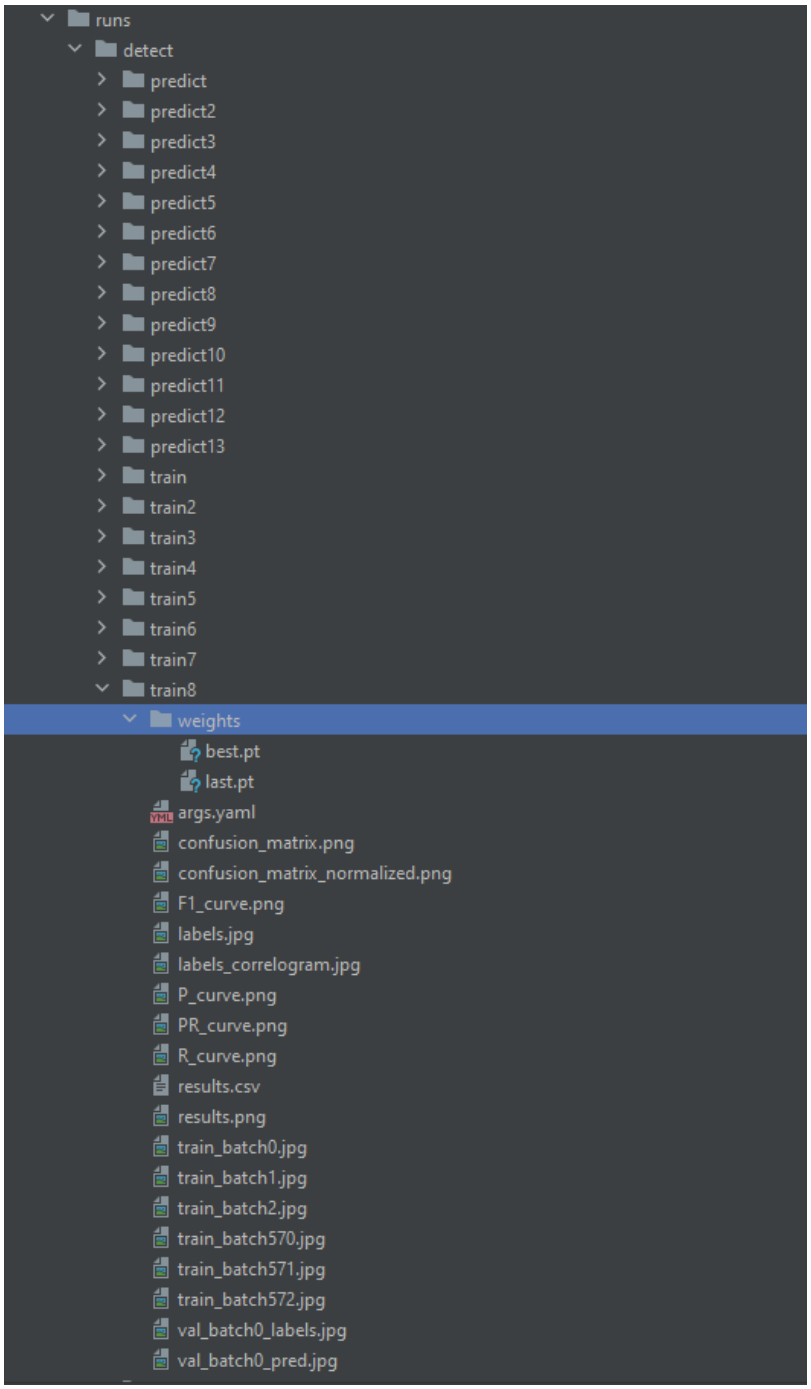


Рисунок 3.6 – Загальний вигляд теки runs

Вигляд процесу навчання нейронної мережі в консолі зображено на рисунку 3.7-3.8.

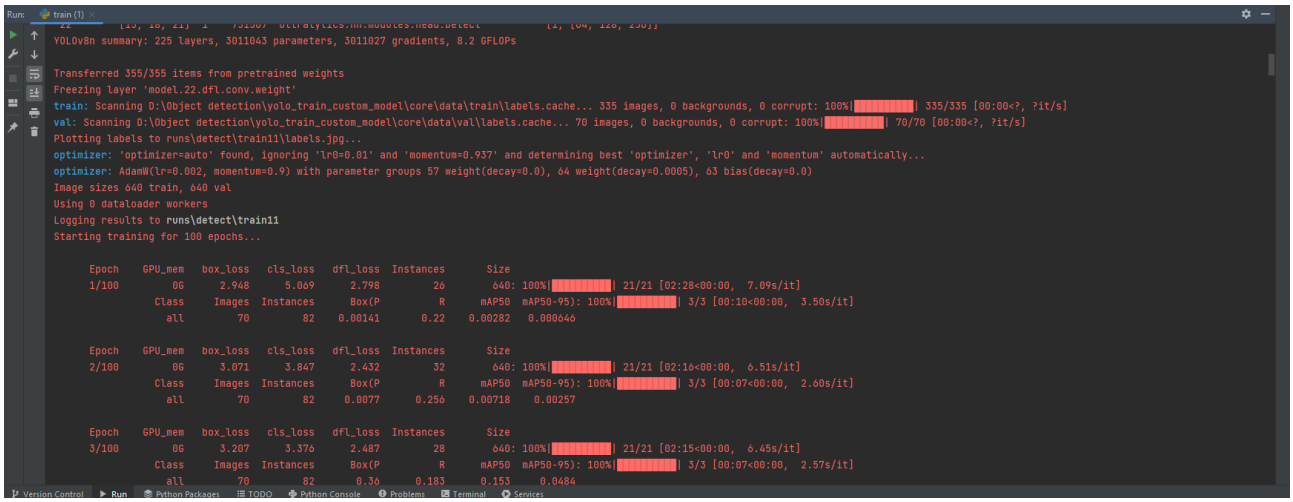


Рисунок 3.7 – Початок процесу навчання в консолі

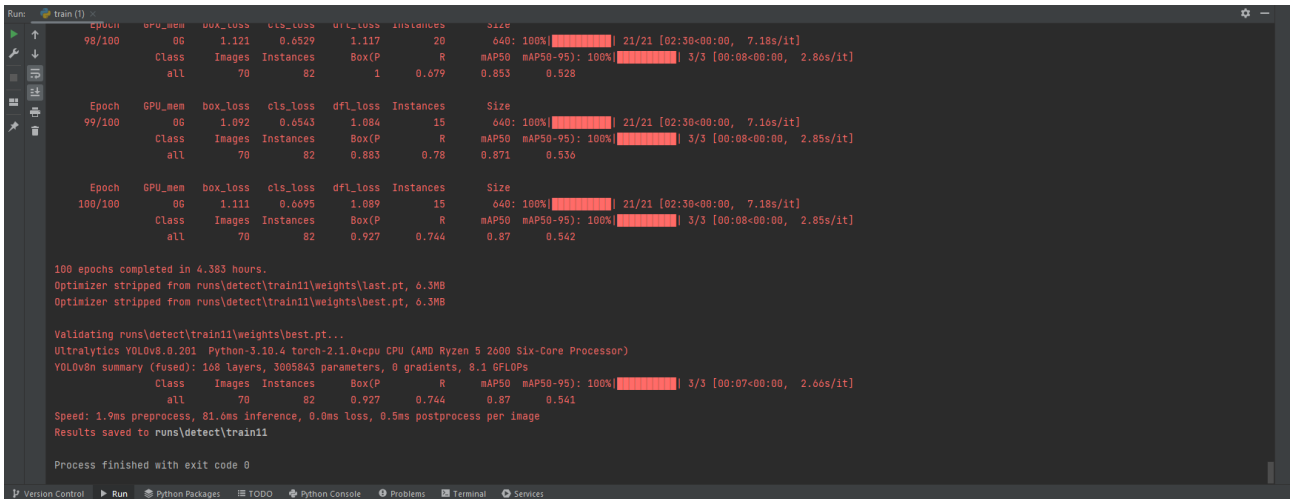


Рисунок 3.8 – Завершення процесу навчання в консолі

В теці weights автоматично створюються два файли з найкращим результатом навчання та останнім. Тобто файл best.pt це вже мережа, яку навчили виявляти потрібні об’єкти. Результати тренування зображені на рисунках 3.9-3.15.

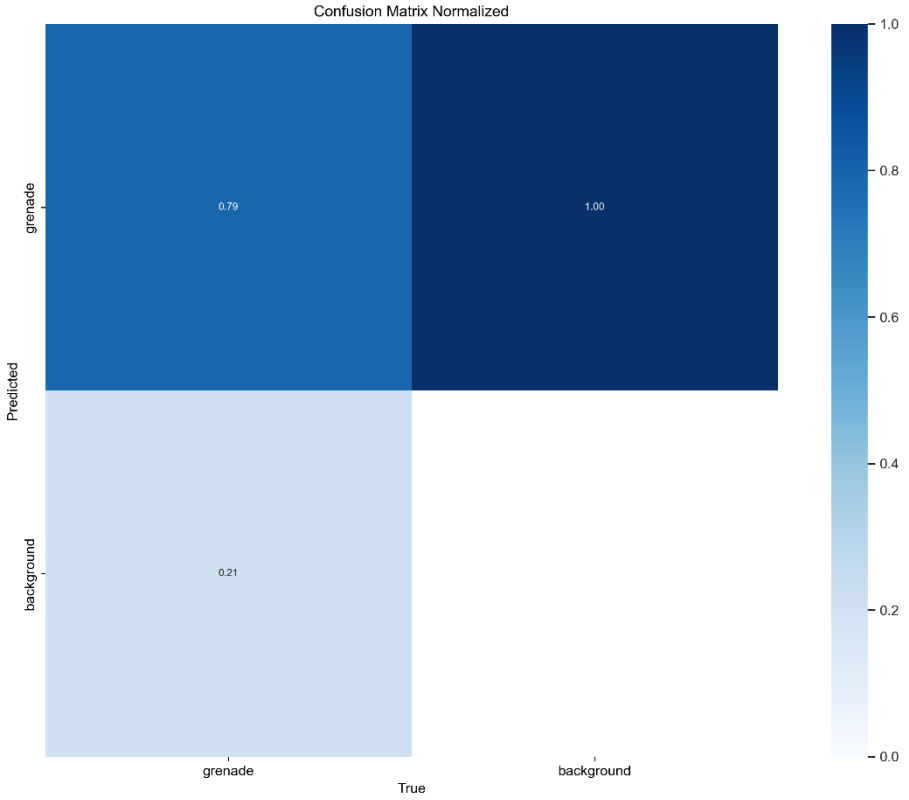


Рисунок 3.9 – Результати тренування у вигляді матриці

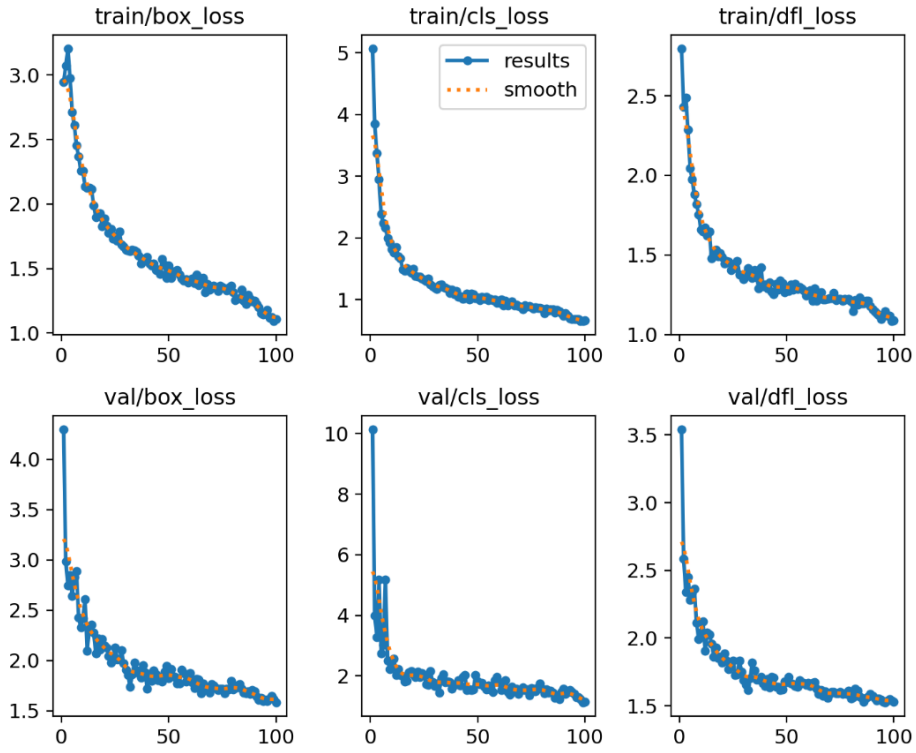


Рисунок 3.10 – Результати тренування у вигляді графіків

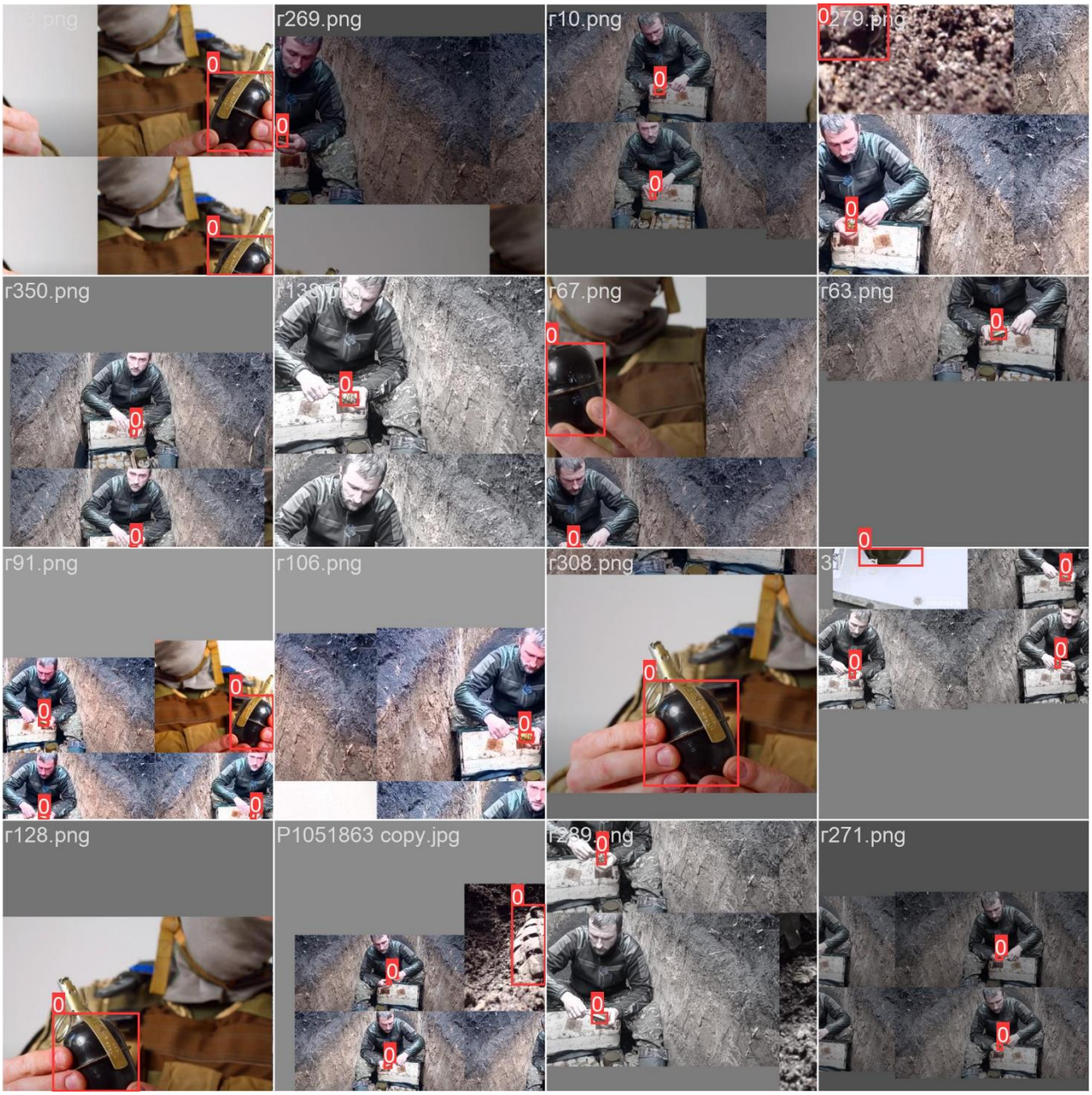


Рисунок 3.11 – Результати тренування першої групи зображень

Рисунок 3.12 – Результати тренування другою групи зображень

Рисунок 3.13 – Результати тренування третьої групи зображень

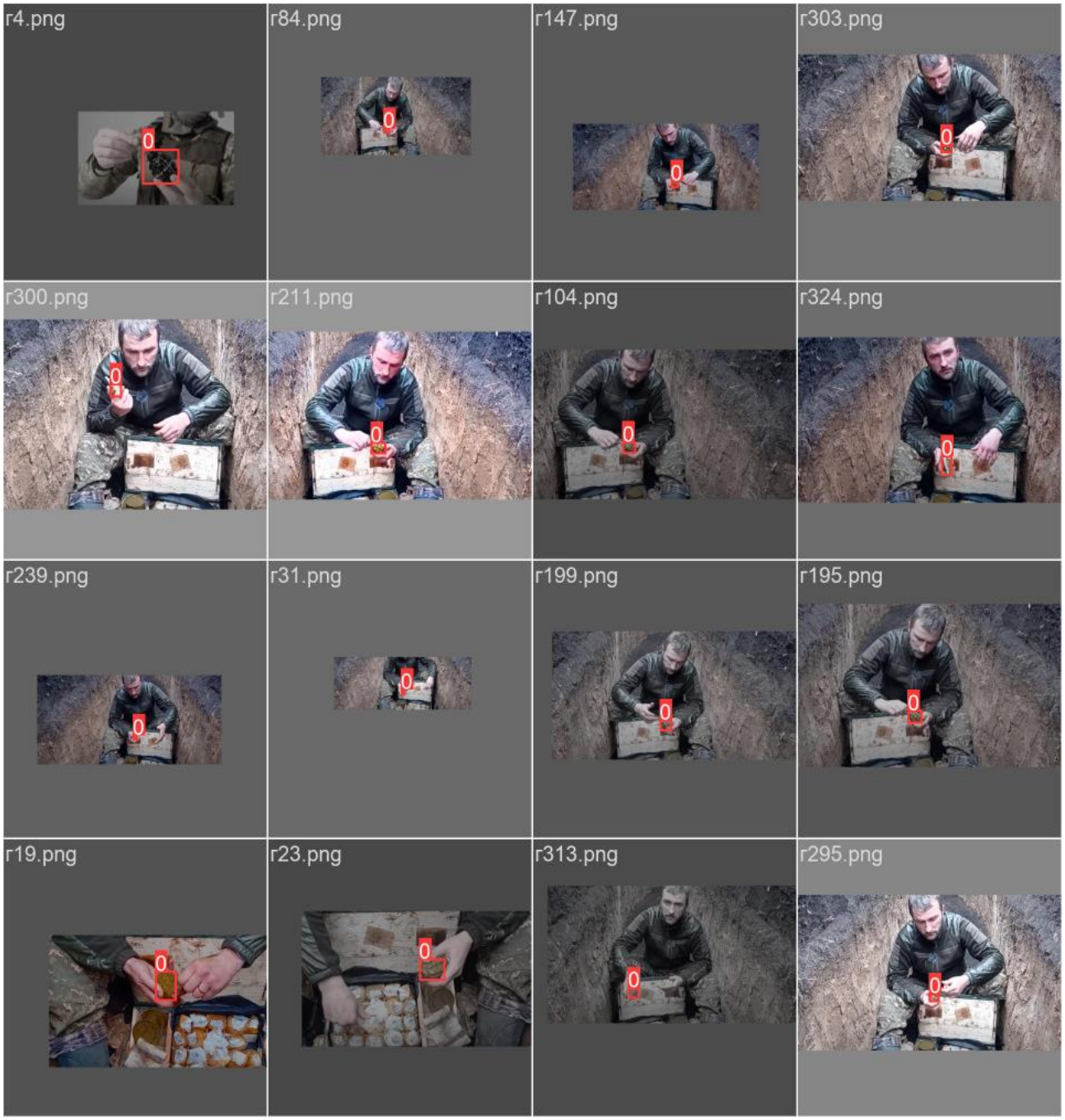


Рисунок 3.14 – Результати тренування четвертої групи зображень

Рисунок 3.15 – Результати тренування п’ятої групи зображень

Наступним кроком потрібно створити файл «val.py» за допомогою якого проведемо валідацію навченої мережі. В цьому файлі треба обрати модель, тобто файл «best.pt», вказати валідацію за параметрами файлу «data.yaml» і запустити виконання програми. Результати валідації нейронної мережі зображено на рисунку 3.16.

Рисунок 3.16 – Результати валідації нейронної мережі

В процесі навчання після валідації, нейронна мережа, самотужки передбачає вибухонебезпечні предмети на зображеннях для валідації. Результати передбачення в процесі валідації зображено на рисунку 3.17.

Рисунок 3.17 – Результати передбачення в процесі валідації

Тепер можна використати навчену нейронну мережу для передбачення і дізнатися чи зможе вона знайти вибухонебезпечний предмет на зображені самотужки. Для цього треба використати зображення, яких не було у нейронної мережі в наборі зображень для навчання чи валідаціїї. Результати передбачення нейронної мережі зображено на рисунку 3.18.

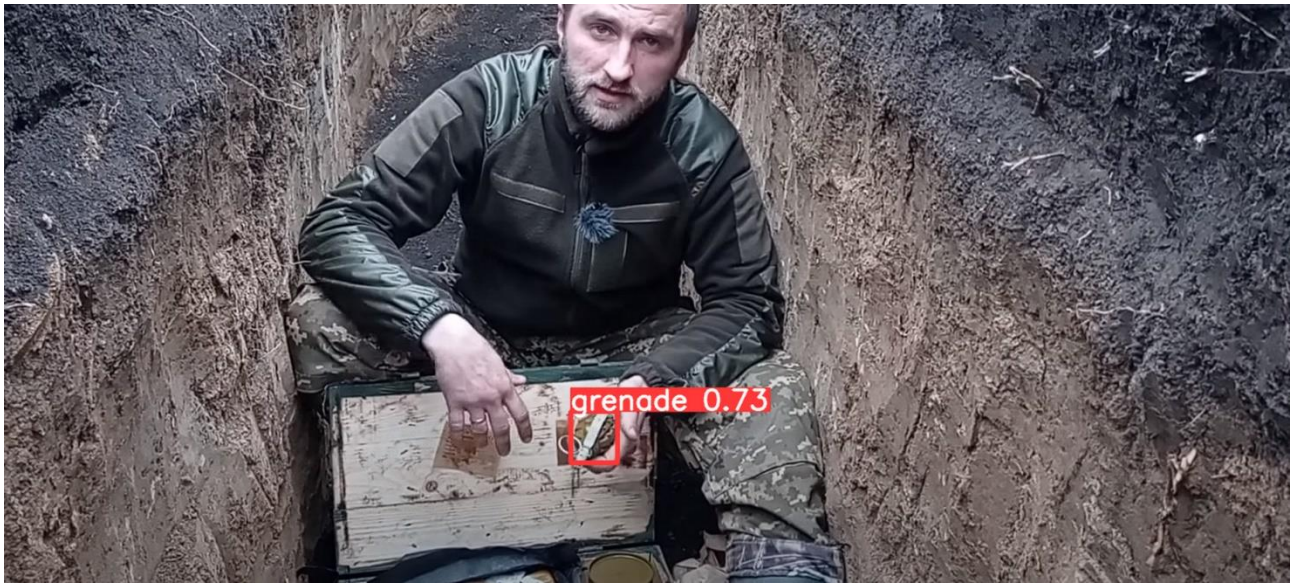


Рисунок 3.18 – Результати передбачення навченої нейронної мережі

Як видно з представленого зображення нейронна мережа навчилася виявляти вибухонебезпечні предмети. Для навчання цієї моделі було використано 335 зображень для тренування та 70 зображень для валідації. Підвищення впевненості та покращення виявлення можна досягти збільшивши набір даних для навчання.

3.3 Висновки до розділу

Цей розділ кваліфікаційної роботи було присвячено навчанню нейронної мережі YOLOv8 для виявлення вибухонебезпечних об’єктів на власному наборі даних. Було використано глибоке навчання, що надало змогу навчати нейронну мережу автоматично без втручання людини.

Головне в навчанні нейронної мережі – це набір даних для навчання та анотація зображень. CVAT дозволяє в зручній формі проводити анотацію зображень та імпортувати ці дані для багатьох моделей нейронних мереж.

PyCharm зручне середовище розробки, яке дозволяє створити власну файлову структуру, переглядати процес навчання нейронної мережі, а також надає більш детальну інформацію у вигляді графів.

4 ОХОРОНА ПРАЦІ

4.1 Охорона праці

Приміщення, де планується встановлення та подальша експлуатація комп'ютера, повинні відповідати проектній документації будинку, яка отримала затвердження від уповноважених державних органів. Крім того, роботодавець повинен дотримуватися діючих санітарних норм щодо освітлення, параметрів мікроклімату (температури, відносної вологості), рівня і сили вібрації, звукового тиску і вогнестійкості приміщення, а також характеристик електромагнітного, ультрафіолетового та інфрачервоного випромінювання.

Наприклад, роботодавцю заборонено розміщувати комп'ютери у приміщеннях, які розташовані у підвалах будинків. З метою уникнення можливих аварій та коротких замикань, також не допускається виконання робіт, які передбачають використання технологічних процесів з надмірною вологою, поруч із приміщеннями, де здійснюється робота з комп'ютером (над чи під ними).

Відповідне приміщення має бути оснащене системами центрального чи індивідуального опалення, кондиціювання та вентиляції повітря. Однак при встановленні таких систем важливо переконатись, що елементи, такі як батареї опалення, водопровідні труби та вентиляційні кабелі, надійно приховані під захисними щитками, які убезпечують від можливого контакту працівника з електричною напругою. В кожній кімнаті, де облаштовуються робочі місця для співробітників, які працюють на комп'ютерах, повинні бути наявні елементи природного та штучного освітлення. Зокрема, на вікнах рекомендується встановлювати легко регульовані жалюзі чи штори, що дозволяють коригувати рівень освітлення в приміщенні. Важливо також розташувати комп'ютери в кімнаті таким чином, щоб світло надходило на екрани моніторів з півдня чи північного сходу.

З метою забезпечення максимального рівня безпеки та охорони праці при

роботі з комп'ютером, виробничі приміщення слід обладнати аптечками першої медичної допомоги, системами автоматичної пожежної сигналізації та вогнегасниками. У приміщенні, де одночасно працюють 5 чи більше комп'ютерів, розміщується службовий вимикач на видному місці, який в разі потреби дозволить повністю відключити електричне живлення кімнати [35].

Під час використання комп'ютерної техніки користувачі можуть стикатися з різними видами небезпек, такими як:

– ураження електричним струмом;

– енергетична небезпека, що виникає внаслідок короткого замикання: опіки, електрична дуга, викид розплавленого металу;

– небезпека загоряння;

– термонебезпека внаслідок воздушно-теплового впливу високих температур через нагрівання конструктивних елементів;

– механічна небезпека, яка може призвести до травм через падіння, контакт з рухомими частинами та порізи від гострих елементів конструкції;

– небезпека випромінювання, включаючи звукове, високочастотне, інфрачервоне, ультрафіолетове та іонізуюче випромінювання, а також видиме світло когерентної високої інтенсивності (лазерне випромінювання);

– хімічна небезпека, яка може виникнути внаслідок контакту з хімікатами, використовуваними для обслуговування обладнання, або від вдихання їх парів [36].

ВИСНОВКИ

В ході виконання кваліфікаційної роботи були проаналізовані нейронні мережі, різницю машинного та глибокого навчання, типи глибоких нейронних мереж, завдання комп'ютерного зору, алгоритми виявлення об'єктів, а саме R-CNN, Fast R-CNN, Faster R-CNN та YOLO.

Було розроблено алгоритм роботи підсистеми виявлення вибухонебезпечних предметів. Завдяки використанню CVAT було проведено анотацію зображень для подальшого навчання нейронної мережі. Проведено навчання нейронної мережі моделі YOLOv8 на власному наборі даних. Набір даних складався з 335 зображень для тренування та 70 зображень для валідації. Навчання відбувалося протягом 5 годин, результати навчання представлені у звіті у вигляді зображень та графіків.

Було проведено використання навченої нейронної мережі для виявлення вибухонебезпечних предметів на зображеннях, де було вірно виявлено вибухонебезпечний предмет з впевненістю 0,73.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Методічні вказівки з підготовки та захисту кваліфікаційної роботи [Електронний ресурс] – режим доступу: https://tapr.nure.ua/wp-content/uploads/2022/04/mv__mag_151.pdf.

2. ДСТУ 3008-15. Документація. Звіти у сфері науки та техніки. структура та правила оформлення. Введ. 2015-06-22. К. Держстандарт України, 2017. 29 с.

3. Бузніков В.Р. Використання технології комп'ютерного зору для виявлення вибухонебезпечних предметів // Збірник студентських наукових статей «ADED» 1 трав. 2023 р. – Харків: ХНУРЕ, 2023. – С. 257–262.

4. What Is a Neural Network? [Електронний ресурс] – режим доступу: https://www.mathworks.com/discovery/neural-network.html

5. The Basics of Neural Networks [Електронний ресурс] – режим доступу: https://towardsdatascience.com/the-basics-of-neural-networks-neural-network-series-part-1-4419e343b2b

6. Neurons in Neural Networks [Електронний ресурс] – режим доступу: https://www.baeldung.com/cs/neural-networks-neurons

7. What is a Neural Network? [Електронний ресурс] – режим доступу: https://www.investopedia.com/terms/n/neuralnetwork.asp

8. Precision and Recall in Machine Learning [Електронний ресурс] – режим доступу: https://www.javatpoint.com/precision-and-recall-in-machine-learning

9. Accuracy, Precision, and Recall in Deep Learning [Електронний ресурс] – режим доступу: https://blog.paperspace.com/deep-learning-metrics-precision-recall-accuracy/

10. Computer Vision Model Performance Evaluation [Електронний ресурс] – режим доступу: https://viso.ai/computer-vision/model-performance/

11. Precision vs. recall vs. accuracy in neural networks [Електронний ресурс] – режим доступу: https://www.educative.io/answers/precision-vs-recall-vs-accuracy-in-neural-networks

12. Мордик О. О., Цимбал О. М. Обчислення середньої точності знаходження об’єктів за допомогою засобів комп'ютерного зору // Матеріали XV міжнародної науково-практичної конференції 20-21 жовт. 2022 р. – Одеса, Видавництво ОНТУ, 2022 р. – С. 192–195.

13. What is AI (Artificial Intelligence)? [Електронний ресурс] – режим доступу: https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-ai

14. Difference Between Machine Learning and Deep Learning [Електронний ресурс] – режим доступу: https://www.geeksforgeeks.org/difference-between-machine-learning-and-deep-learning/

15. Deep Learning vs. Machine Learning – What`s The Difference? [Електронний ресурс] – режим доступу: https://levity.ai/blog/difference-machine-learning-deep-learning

16. What`s the difference between Machine Learning and Deep Learning? [Електронний ресурс] – режим доступу: https://viso.ai/deep-learning/deep-learning-vs-machine-learning/

17. Deep Neural Network: The 3 Popular Types (MLP, CNN and RNN) [Електронний ресурс] – режим доступу: https://viso.ai/deep-learning/deep-neural-network-three-popular-types/

18. Introduction to Convolution Neural Network [Електронний ресурс] – режим доступу: https://www.geeksforgeeks.org/introduction-convolution-neural-network/

19. What Is Computer Vision? [Електронний ресурс] – режим доступу: https://www.v7labs.com/blog/what-is-computer-vision

20. What is Computer Vision? IBM [Електронний ресурс] – режим доступу: https://www.ibm.com/topics/computer-vision

21. R-CNN vs Fast R-CNN vs Faster R-CNN – A Comparative Guide [Електронний ресурс] – режим доступу: https://analyticsindiamag.com/r-cnn-vs-fast-r-cnn-vs-faster-r-cnn-a-comparative-guide/

22. R-CNN vs Fast R-CNN vs Faster R-CNN ML – GeeksforGeeks

[Електронний ресурс] – режим доступу: https://www.geeksforgeeks.org/r-cnn-vs-fast-r-cnn-vs-faster-r-cnn-ml/

23. R-CNN, Fast R-CNN, Faster R-CNN, YOLO – Object Detection Algorithms [Електронний ресурс] – режим доступу: https://towardsdatas-cience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e

24. R-CNN, Fast R-CNN, Faster R-CNN, YOLO – DATA SCIENCE [Електронний ресурс] – режим доступу: https://datascience.eu/computer-vision/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms/

25. Faster R-CNN Explained for Object Detection Tasks [Електронний ресурс] – режим доступу: https://blog.paperspace.com/faster-r-cnn-explained-object-detection/

26. Мікрокомп'ютер Rasberry Pi 3 Model B+ [Електронний ресурс] – режим доступу: https://evo.net.ua/raspberry-pi-3-model-b/

27. Камера Rasberry Pi Camera Board [Електронний ресурс] – режим доступу: https://evo.net.ua/raspberry-pi-camera-board/

28. DC Grearbox Motor TT Motor [Електронний ресурс] – режим доступу: https://www.adafruit.com/product/3777

29. Зовнішній акумулятор Anker Power Bank PowerCore 10000 mAh with QC3.0 V3 [Електронний ресурс] – режим доступу: https://ti.ua/ua/anker-power-bank-powercore-10000-mah-with-qc3-0-v3-black.html

30. Як розрахувати час роботи обладнання від безперебійного блоку живлення [Електронний ресурс] – режим доступу: https://www.bezpeka-shop.com/ua/blog/poleznye-sovety/kak-rasschitat-vremya-raboty-oborudovaniya-ot-bespereboynogo-bloka-pitaniya/

31. Що таке пітон? – визначення з технопедії [Електронний ресурс] – режим доступу: https://uk.theastrologypage.com/python

32. Help – Ultralytics YOLOv8 Docs [Електронний ресурс] – режим доступу: https://docs.ultralytics.com/help/

33. PyCharm: the Python IDE for Professional Developers [Електронний ресурс] – режим доступу: https://www.jetbrains.com/pycharm/

34. How to Use the CVAT Annotation Tool [Електронний ресурс] – режим доступу: https://blog.roboflow.com/cvat/

35. Охорона праці при роботі з комп’ютером: роз’яснює Держпраці [Електронний ресурс] – режим доступу: https://news.dtkt.ua/labor/labor-relations/39447-oxorona-praci-pri-roboti-z-kompiuterom-roziasniuje-derzpraci

36. Аналіз шкідливих і небезпечних виробничих факторів при роботі на комп’ютері [Електронний ресурс] – режим доступу: https://vuzlit.com/1008307/analiz_shkidlivih_nebezpechnih_virobnichih_faktoriv_roboti_kompyuteri