ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Програмний додаток для розпізнавання номерних знаків за допомогою методів комп'ютерного зору»

на здобуття освітнього ступеня бакалавр
зі спеціальності 122 Комп'ютерні науки
(код, найменування спеціальності)
освітньо-професійної програми Штучний інтелект
(назва)

Кваліфікаційно робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідні джерела

_____ **Валентин ЗАЦЕРКОВНИЙ**
(підпис)

Виконав: Здобувач вищої освіти група ШІД-42
Валентин ЗАЦЕРКОВНИЙ

Керівник: **Олександр ЗВЕНІГОРОДСЬКИЙ**

Резидент: _____
Ім'я ПРІЗВИЩЕ

Київ 2025

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Штучного інтелекту
Ступінь вищої освіти Бакалавр
Спеціальність 122 Комп'ютерні науки
Освітньо-професійна програма Штучний інтелект

ЗАТВЕРДЖУЮ
Завідувач кафедрою Штучного інтелекту

_____ **Ольга ЗІНЧЕНКО**
«_____» _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Зацерковному Валентину Жановичу
(прізвище, ім'я по батькові здобувача)

1. Тема кваліфікаційної роботи: Програмний додаток для розпізнавання номерних знаків за допомогою методів комп'ютерного зору

керівник кваліфікаційної роботи Звенігородський Олександр к.т.н., доцент,

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «24» лютого 2025р. № 56.

2. Строк подання кваліфікаційної роботи «23» травня 2025р.

3. Вихідні дані до кваліфікаційної роботи: науково-технічна література.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

Огляд проблеми розпізнавання номерних знаків автомобільного транспорту.
Аналіз принципу роботи сучасних технологій глибокого навчання і оптичного розпізнавання символів.
Розробка формальної моделі моніторингу транспортних засобів на підприємстві.
Огляд засобів програмування для створення програмного забезпечення
Розробка програмного додатку розпізнавання номерних знаків.

5. Перелік графічного матеріалу: *презентація*
   1. Мета, об’єкт та предмет дослідження
   2. Завдання дослідження
   3. Літературний огляд проблеми розпізнавання номерних знаків
   4. Використані технології та інструменти
   5. Модель системи моніторингу номерних знаків транспортних засобів
   6. Демонстрація програмного забезпечення

6. Дата видачі завдання «25 лютого» 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Літературний огляд проблеми розпізнавання номерних знаків автомобільного транспорту.	25.02-05.03.2025	Виконано
2	Аналіз алгоритмів глибокого навчання і оптичного розпізнавання символів	06.03-10.03.2025	Виконано
3	Розробка формальної моделі моніторингу транспортних засобів.	11.03-15.03.2025	Виконано
4	Реалізація програмного забезпечення	16.03-10.04.2025	Виконано
5	Тестування програмного забезпечення	11.04-13.04.2025	Виконано
6	Оформлення роботи : вступ, висновки, реферат	14.04-20.05.2025	Виконано
7	Створення демонстраційних матеріалів	21.05-23.05.2025	Виконано

Здобувач вищої освіти _____ Валентин ЗАЦЕРКОВНИЙ
(підпис) *Ім’я ПРІЗВИЩЕ*

Керівник кваліфікаційної роботи _____ Олександр ЗВЕНІГОРОДСЬКИЙ
(підпис) *Ім’я ПРІЗВИЩЕ*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 67 сторінок, 4 табл., 22 рис., 21 джерело.

Мета дослідження – збільшення ефективності автоматизованого виявлення і розпізнавання номерних знаків транспортних засобів для покращення процесів моніторингу потоку машин на контрольно-пропускних пунктах, та автоматизації контролю за транспортом на підприємствах.

Об'єкт дослідження – процес розпізнавання образів і машинного навчання в галузі комп'ютерного зору.

Предмет дослідження – алгоритми розпізнавання об'єктів на фото і у відео послідовностях, алгоритми оптичного розпізнавання символів.

Короткий зміст роботи – У роботі проведено літературний огляд проблеми розпізнавання номерних знаків автомобільного транспорту, проаналізовано алгоритми виявлення об'єктів і символів на зображеннях.

Створено формальну модель моніторингу номерних знаків транспортних засобів. Обрано засоби і технології програмування для створення програмного забезпечення.

Для реалізації системи розпізнавання номерних знаків було використано наступні бібліотеки: OpenCV, Ultralytics YOLO, PaddleOCR, Re, Tkinter, Pandas, Pillow, DateTime. Collections.

Проведено тестування системи, що показало високі показники точності та швидкості обробки даних.

КЛЮЧОВІ СЛОВА: КОМП'ЮТЕРНИЙ ЗІР, МОДЕЛЬ YOLO, ОПТИЧНЕ РОЗПІЗНАВАННЯ СИМВОЛІВ(OCR), СИСТЕМА РОЗПІЗНАВАННЯ, НОМЕРНІ ЗНАКИ АВТОМОБІЛЬНОГО ТРАНСПОРТУ

ABSTRACT

Text part of the master's qualification work: 67 pages, 4 table, 22 pictures, 21 sources.

Purpose of the study: to increase the efficiency of automated detection and recognition of vehicle license plates to improve the processes of monitoring the flow of cars at checkpoints and automate transport control at enterprises.

Object of research: the process of pattern recognition and machine learning in the field of computer vision.

Subject of research: algorithms for recognizing objects in photos and video sequences, algorithms for optical character recognition.

Summary of the work: The paper conducts a literature review of the problem of license plate recognition of motor vehicles, analyzes algorithms for detecting objects and symbols in images.

A formal model of vehicle license plate monitoring is created. Programming tools and technologies for creating software were selected.

The following libraries were used to implement the license plate recognition system: OpenCV, Ultralytics YOLO, PaddleOCR, Re, Tkinter, Pandas, Pillow, DateTime. Collections.

The system was tested, which showed high accuracy and speed of data processing.

KEYWORDS: COMPUTER VISION, YOLO MODEL, OPTICAL CHARACTER RECOGNITION (OCR), RECOGNITION SYSTEM, LICENSE PLATES OF MOTOR VEHICLES

ЗМІСТ

ВСТУП

У сучасному світі кількість транспортних засобів стрімко зростає, контролювати транспортні потоки без втручання людської праці набуває великого значення. Автоматизовані системи для розпізнавання номерних знаків широко застосовуються для різних завдань і в багатьох сферах, а саме – пропускні пункти, системи відеоспостереження на підприємствах, автомобільні стоянки. Завдяки досягненням в галузях комп'ютерного зору та штучного інтелекту відкривається багато можливостей для автоматизації рутинних робочих процесів.

Системи розпізнавання номерних знаків автомобільного транспорту здатні працювати в реальному часі і підтримувати різноманітні умови зйомки, що дозволить мінімізувати помилки які може допустити людина і зробити процес контролю за транспортними потоками швидшим, що дозволить водіям не витрачати багато часу на проїзд через контрольно-пропускні пункти, чи при в'їзді на автомобільні стоянки. Водночас задача розпізнавання може бути ускладнена - низькою якістю зображень, поганими погодними умовами, чутливістю системи розпізнавання до зміни структури номерного знаку.

Останні дослідження демонструють, що сучасні системи розпізнавання номерних знаків створюються за допомоги комбінування різних технологій, основними методами є глибоке навчання та технології оптичного розпізнавання. Щоб досягти хороших показників в розпізнаванні треба ретельно обирати технології і алгоритми реалізації програмного забезпечення. Більшість готових програмних рішень мають обмежений функціонал, деякі не підтримують розпізнавання номерних знаків різних країн, інші вимагають від користувача мати високі апаратні вимоги.

Ціль проекту полягає в покращенні ефективності автоматизованого контролю транспортних засобів на підприємствах і контрольно-пропускних пунктах. Програмне забезпечення для розпізнавання номерів повинне поєднувати в собі простоту у використанні, швидкість обробки кадрів, велику точність. Система забезпечуватиме підтримку різних форматів номерних знаків, виявлення

їх у несприятливих погодних умовах і повинна бути невибагливою до апаратних характеристик.

Для реалізації поставленої мети визначені такі завдання:

- Зробити літературний огляд проблеми розпізнавання номерних знаків автомобільного транспорту.
- Проаналізувати сучасні технології глибокого навчання і оптичного розпізнавання символів.
- Розробити формальну модель системи моніторингу транспортних засобів.
- Вибрати засоби програмування та створити програмний продукт.
- Провести тестування системи, зробити аналіз точності розпізнавання номерних знаків.

1 КОМП'ЮТЕРНИЙ ЗІР ТА ЙОГО ПРАКТИЧНЕ ЗАСТОСУВАННЯ

1.1 Теоретичні основи розпізнання образів

Розпізнавання образів є одним із самих важливих напрямків в технології комп’ютерного зору, що дозволяє системам аналізувати зображення чи відеозаписи, класифікувати та ідентифікувати об’єкти. Розпізнавання образів широко використовується в наш час, зокрема в системах відеоспостереження на підприємствах, автоматичному виявленні машин та їх номерних знаків, розпізнаванні обличь людей, для аналізу медичних знімків пацієнтів у сфері охорони здоров’я. Використання цієї технології дуже полегшує процеси, де раніше була потрібна участь людини, а також забезпечує більш високу ефективність і точність, тому що мінімізує ймовірність людської помилки та автоматизує рутинні робочі процеси. В таблиці 1.1 зображені покрокові досягнення в сфері комп’ютерного зору.

Таблиця 1.1

Історія комп’ютерного зору

Етап розвитку	Рік	Ключова подія	Досягнення
Біологічні основи	1959	Дослідження Хубела і Візеля	Розуміння обробки зображень нейронами
Цифрова обробка	1959	Сканер Кірша	Початок цифрової обробки зображень
Комп’ютерний зір	1962	Дисертація Робертса	Перетворення 2D зображень у 3D
Нейронні мережі	1982	«Неокогнітрон» Фукушіми	Основа згорткових мереж
Набори Даних	2007	«Imagenet» Фей-Фей Лі	Великий набір даних для навчання
Глибоке навчання	2012	Alexnet	Прорив у точності розпізнавання

Першим кроком до розпізнавання образів було відкриття, що нейрони в зоровій корі реагують на специфічні ознаки, такі як краї об’єктів. Це відкриття стало основою для розуміння біологічної обробки візуальної інформації, це все сталося у 1959 році і дало початкову точку для розвитку теми глибокого навчання, яка і досі працює за цім принципом в комп’ютерному розпізнаванні зображень[1]. Потім просування в галузі комп’ютерного зору сталося на початку 1963 року, Лоуренс Робертс написав докторську дисертацію яка називалась «Машинне сприйняття тривимірних тіл», в ній він дослідив перетворення двохвимірних зображень на трьохвимірні, це теж стало відправною точкою для подальших досліджень в області розпізнавання зображень [2]. У той же період Руссел Кірш розробив перший цифровий сканер фотографій, який перетворював зображення в бінарні дані. Ця технологія дозволила відсканувати перше цифрове зображення з роздільною здатністю 176x176 пікселів (30,976 пікселів), заклавши основу для цифрової обробки та аналізу зображень.

У 1980-х роках японський вчений Куніхіко Фукусіма побудував одну із перших нейронних мереж. Ця мережа, названа «Неокогнітрон», складалася з кількох згорткових шарів, рецептивні поля мали вагові вектори, більш відомі як фільтри. Ці фільтри переміщувались по вхідних значеннях (таких як пікселі зображення), виконували обчислення, а потім запускали події, які використовувалися як вхідні дані для наступних шарів. Таким чином, Неокогнітрон можна назвати першою нейронною мережею, яка отримала статус «глибокої», і справедливо розглядається як предок сучасних згорткових нейронних мереж [1]. На рисунку 1.1 можна побачити відмінності простої нейронної мережі та мережі глибокого навчання

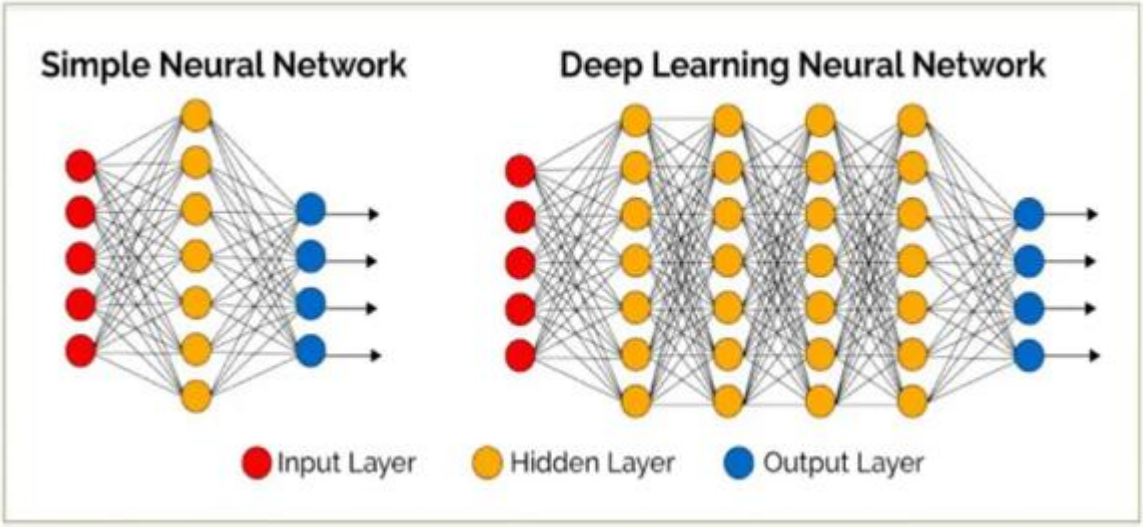

Рис. 1.1 Проста нейронна мережа, та мережа глибокого навчання

Перші масштабні набори даних заснувала Фей-Фей Лі, у 2010 році її база даних під назвою «ImageNet» містила понад 3 мільйони анотованих і сегментованих зображень у більш ніж 5000 типів категорій. Цей набір даних став початком для навчання нейронних мереж, та оцінки алгоритмів розпізнавання образів і щоб привернути до цього увагу інших науковців у 2010 році був створений конкурс Imagenet Large Scale Visual Recognition Challenge (ILSVRC), в цьому конкурсі потрібно було створювати свої програмні продукти та змагатися в класифікації та розпізнаванню об’єктів використовуючи базу даних «ImageNet» [3]. В ньому приймала участь команда Університету Торонто, яка у 2012 році представила всім згорткову нейронну мережу «Alexnet», вона знизила рівень помилок із 25% до 15,3%, ці результати наочно демонструють переваги глибокого навчання. До 2017 року рівень помилок серед нейронних мереж учасників цього конкурсу впав нижче 5%, що свідчить про швидкий прогрес.

Для ефективної і діючої роботи розпізнавання образів важлива коректна обробка зображень. Вона налічує велику кількість технологій і різних варіацій їх використання під кожну потребу, із найбільш ключових є – перетворення Фур’є, цифрова фільтрація, сегментація. Завдяки цим методам система може відокремити основні риси та характеристики зображень, мінімізувати кількість шуму на них і зробити подальшу обробку більш якіснішою.

Перетворення Фур’є – це функції інтегрального перетворення, вони отримують вхідну функцію, а потім видають іншу. Вихідним результатом ми отримаємо комплексну функцію частоти. Перетворення Фур’є може використовуватись для виконання багатьох задач. В розпізнаванні образів це є ключовим методом для аналізу зображень у області частот. Воно дає змогу описати зображення як сукупність складових (синусів та косинусів). Існує декілька варіацій перетворень: інтегральне, ряди Фур’є, дискретні, нижче наведено опис базового інтегрального перетворення.

Інтегральне перетворення – використовується для аналізу вмісту зображення. Перетворення Фур’є розкладає зображення на низькочастотні й високочастотні компоненти. 1D перетворення обробляє функцію однієї змінної, що зображено на формулі (1.1), а 2D перетворення обробляє функцію двох [4].

$$f(x) = \int_{\infty}^{\infty} F(k) e^{2\pi i k x} dk \qquad (1.1)$$

Де f(x) – змінна яку ми одержимо в результаті перетворення;

F(k) – спектральна функція, вона описує розподіл частот сигналів;

k – змінна, яка позначає частоту;

x – змінна, що відповідає просторовій координаті;

e – експонента, яка описує коливання.

Ця формула є основною в аналізі сигналів і обробці зображень, що дозволяє працювати із різними частотами.

Цифрова фільтрація – це обробка цифрових сигналів, її мета полягає в покращені сигналу та знаходженні в ньому інформації. Ця технологія використовує математичні операції до дискретних часових послідовностях. Цифрова фільтрація широко використовується у розпізнаванні образів, вона забезпечує обробку зображень для ідентифікації об’єктів. Вона допомагає підготувати дані для наступних кроків обробки, а саме: регулює шум, виділяє певні ознаки, що робить її важливим інструментом у комп’ютерному зорі. Цифрові фільтри бувають із зворотним зв’язком та без зворотного зв’язку. На

формулі (1.2) показане диференційне рівняння цифрової фільтрації зі зворотнім зв’язком, це основа рекурсивних систем [5].

$$y(n) = \sum_{i=0}^{M} b_i x(n-i) - \sum_{j=1}^{N} a_j y(n-j) \quad (1.2)$$

y(n) – це поточний вихід, що видає система;

bi – прямий зв’язок;

aj – зворотній зв’язок;

x(n – i) – те що ми ввели в систему;

y(n – j) – те що система видала.

Основною перевагою використання цифрових фільтрів з зворотнім зв’язком це їхня пам’ять. Це дозволяє фільтру адаптуватись до змін і обробляти всю інформацію, що сильно може допомогти при фільтрації відеозаписів, обробці сигналів і покращенні якості зображень.

Одним із самих популярних у використанні методів цифрової фільтрації у роботі з підготовкою даних до розпізнавання образів є – лінійна фільтрація. Вона використовує згортку для перетворення піксельних значень, це значно спрощує налаштування зображень, дає можливість налаштувати контури і згладжування. На формулі (1.3) зображена лінійна фільтрація, можна побачити що вона використовує двовимірну згортку x(i, j), з ядром h(m, n) [6].

$$y[i,j] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} h[m,n] * x[i-m, j-n] \quad (1.3)$$

y[i, j] – значення вихідного пікселя зображення після згортки, де i – індекс рядка, а j – стовпця, вони разом показують місцезнаходження пікселя в зображенні;

h[m, n] – ядро згортки (фільтр);

x[i – m, j – n] – значення вхідного зображення включаючи зсунуте ядро.

Таким чином завдяки згортці відбувається фільтрування зображення, враховуються пікселі і задане ядро, що допомагає впоратись з усуненням зайвого шуму, підвищити контраст тощо.

Сегментація – це розподілення цифрового зображення на кілька змістовних частин (сегментів) для виявлення ознак образів. Вона використовується для систем розпізнавання об’єктів, роблячи зображення простішим для аналізу. Основні види сегментації зображень складаються з таких методів:

- Порогова сегментація – цей метод робить зображення бінарним, порівнює яскравість пікселів між собою і єдиним пороговим значенням для цього зображення, однак буває що єдине порогове може давати помилки коли одні області дуже затемнені, чи навпаки освітлені. Адаптивна порогова сегментація впорається з цією проблемою, тому що вона дозволяє працювати при нерівномірних освітленнях. Існує багато методів бінарізації зображень, я хочу виділити декілька, для глобального порогу зазвичай використовується метод Отсу [7], він сам обирає значення яка краще розділяє фон і об’єкти на зображенні, а для адаптивного методи Ніблака чи Сауволи, їх методологія часто використовується для роботи із задачами оптичного розпізнання символів.
- Методи нарощування областей – вони включають в собі аналіз певних зон в яких текстура на зображенні має різкі переходи. Виявлення змін в текстурах знаходиться при порівнянні локальних характеристик зображення у кожній відокремленій зоні. Алгоритм обирає пікселі які знаходяться в точках з різною текстурою та починає розширювати цю область, а закінчується це тоді, коли точки перестають приєднуватися до області. Метод нарощування областей зазвичай використовують при обробці зображень щоб прибрати шуми.
- Морфологічні методи – використовуються для обробки і визначення форм об’єкту. Вони допомагають правильно структурувати контури на зображенні.

- Метод шаблонів – використовується при порівнянні вхідного зображення із заготовленими зразками. Він шукає схожі ознаки в зображеннях. Основні підходи при використанні методу шаблонів : кореляційний аналіз, відповідність ознак на яких визначається схожість з шаблоном, а також пряма відповідність, яка покроково аналізує розпізнавання зображення з шаблоном.

Які є основні підходи до розпізнавання образів?

Статистичне розпізнавання - цей метод більше всього вивчався і використовувався ще до того, як нейромережеві методи розпізнання стали такими популярними як у наші часи. В основі цього методу лежить аналіз статистичних даних для класифікації і розпізнавання образів. Кожен об’єкт повинен описуватись певним набором характеристик, вони формують вектор. Модель навчається розподіляти об’єкти таким чином, щоб вектори інших класів займали різні області, а після навчання система повинна розуміти до якого класу відноситься певний об’єкт, дивлячись на його ознаки по існуючим кластерам.

Синтаксичне розпізнавання - цей підхід розбиває зображення чи дані на дрібні частинки, він аналізує як всі ці частинки поєднуються між собою. Таким чином цей метод допомагає розпізнавати складні зображення і дані. Щоб його використовувати, нам потрібно мати навчальні дані, в яких є правила, вони повинні пояснювати як кожна частинка взаємодіє одна з одною. Синтаксичне розпізнавання образів не просто розпізнає зображення, а ще пояснює з яких частин воно побудоване і як саме це відбувається. Такий підхід дуже зручний для речей із чіткою, послідовною структурою, наприклад його можна використовувати для аналізу хвиль на ЕКГ де я певна послідовність, або для зображень з повторюваними візерунками [8] .

Розпізнавання за допомогою нейронних мереж - цей підхід використовує штучні нейронні мережі, він є самими популярним і ефективним методом для розпізнавання об’єктів. На відміну від попередніх підходів, нейронні мережі здатні автоматично знаходити складні залежності в даних. За допомогою цього вони можуть забезпечити високу точність і швидкість в задачах класифікації і

виявленні об’єктів. Для розпізнання образів зазвичай використовують згорткові нейронні мережі (Convolutional Neural Networks, CNN). Їх архітектура складається з трьох головних елементів: згорткові шари - виявляють локальні ознаки, перетворюють вхідні зображення на вихідні, шари підвибірки – зменшують розмір даних, аналізують і зберігають найважливішу інформацію, повнозв’язні шари – ітерують ознаки, які були отримані з попередніх шарів, здійснюють класифікації. У нейронних мережах є така властивість, що шари завжди з’єднані так, щоб сигнал від входу міг передаватися через всю систему і доходити до виходу з якомога меншою похибкою. Характеристика роботи мережі визначаються функціями її елементів, структурою та способом у який шари зв’язані між собою.

В сьогоденні комп’ютерний зір став не лише звичайним розпізнаванням зображень, чи об’єктів на них, він почав розуміти контекст, вміє прогнозувати поведінки, робить можливим співпрацю машин з людьми, а розпізнавання образів спирається на технології глибокого навчання та згорткових нейронних мереж, за допомогою яких можна досягати високої точності в задачах кваліфікації та виявленні об’єктів. Розвитку комп’ютерного зору стрімко прогресує у сфері глибокого навчання, згорткові нейронні мережі завжди вдосконалюються, з’являються нові архітектури і методи застосування.

1.2 Приклади застосування програм для ідентифікації номерних знаків

Технології розпізнавання номерних знаків автомобільного транспорту використовується у багатьох сферах життя. Машин на дорогах стає все більше і більше, а забезпечувати контроль транспортних засобів стає складніше, тому системи для розпізнавання номерів дуже поширені і використовуються кожен день у різних сферах, щоб автоматизовано здійснювати контроль транспортних засобів без втручання людини. Кращі системи для ідентифікації номерних знаків зазвичай працюють за допомогою комп’ютерного зору і моделей штучного

інтелекту, а для зчитування символів на номерах автомобіля використовують технологію оптичного розпізнавання символів (OCR).

Найбільш поширеними прикладами застосування цієї технології є:

- Моніторинг трафіку транспортних засобів – використовується для збору статистичних даних про транспортний потік на дорогах, це допомагає аналізувати завантаженість доріг, визначати де частіше всього затори. Завдяки цьому можна вдосконалювати транспортні розв'язки, та планувати інфраструктурні проекти.
- Контрольно-пропускні пункти – може бути використана при в'їздах та виїздах з підприємств, промислових ділянок, приватних територій. Встановленні на КПП камери відеоспостереження автоматично фіксують номерний знак транспортного засобу, який хоче заїхати на територію, перевіряють ідентифікований номер у своїй базі даних та приймають рішення, дозволяти приїзд автомобілю чи ні. Таким чином використання системи розпізнавання номерних знаків на контрольно-пропускних пунктах допоможе автоматизувати процеси, а також зменшить час на перевірку номерів транспортних засобів.
- Платний паркінг – у торгових центрах, бізнес-центрах, житлових комплексах часто зустрічаються платні стоянки для автомобільного транспорту. Використовуючи систему ідентифікації номерних знаків процес платного паркування стає дуже зручним для водіїв. Коли машина заїжджає на паркінг камера передає системі дату, час та номер автомобільного транспорту, а при виїзді з платної стоянки транспортний засіб знову сканується системою, визначається тривалість перебування транспорту на паркувальному майданчику і вираховується сума яку повинен сплатити водій. Система дає перевагу для водіїв в зменшенні утворення заторів завдяки швидкій обробці потоку машин, які бажають заїхати на паркінг.
- Пошук викрадених автомобілів – камери які встановлені на вулицях міста, сканують номери транспортних засобів, перевіряють їх зі своєю

базою втрачених автомобілів і миттєво реагують, якщо перевірений номер знаходиться у пошуковій базі.

- Моніторинг порушників швидкісного режиму – в містах встановлюються автоматизовані системи сканування номерних знаків, вони розпізнають швидкість автомобіля, якщо вона перевищує встановлену норму, то водію автоматично надсилається штраф.

1.3 Огляд існуючих програмних рішень

Завдяки великому попиту на використання систем розпізнавання, ця галузь завжди вдосконалюється, тому системи ANPR (Automatic Number Plate Recognition) з кожним роком стають більш точними і швидкими. В цьому розділі представлено самі поширені програмні рішення, їх особливості, переваги і недоліки.

Безкоштовні програми з відкритим кодом.

Використання програм з відкритим кодом це ідеальний вибір для звичайних користувачів, які не планують витрачати кошти, а також програми з відкритим кодом обирають розробники, тому що в них з’являється багато можливостей кастомізувати початкову програму для використання в своїх цілях.

ALPR-Unconstrained [9] – відкрите програмне забезпечення яке призначене для виявлення номерних знаків у складних умовах. Ця програма використовує глибоке навчання для підвищення точності і швидкості результатів розпізнавання.

Переваги:

- безкоштовна;
- висока точність;
- модель навчена на розпізнаванні номерів в складних умовах;
- є можливість кастомізації та інтегрування у свої проекти;
- відкритий код на платформі Github для всіх користувачів [9].

Недоліки:

- обмежена підтримка зі сторони розробників;
- складність розгортання на підприємствах;
- немає конкретного списку регіонів номерних знаків, які система підтримує, тому, можливо, потрібно буде донавчити її даними з номерними знаками потрібної країни.

OpenALPR – це передова система розпізнавання номерних знаків транспорту, є багато версій цієї програми і в кожній із них вона все більше вдосконалювалась, алгоритми розпізнавання змінювались, у ранніх версіях використовували класичні методи комп'ютерного зору, такі як HOG (Histogram of Oriented Gradients) разом з технологією SVM (Support Vector Machine), а сучасні версії працюють на глибокому навчанні та технологіях оптичного розпізнавання символів. Системою OpenALPR користувалось багато компаній, працювало понад 12000 камер у різних країнах світу. За інформацією в джерелі [10] у 2019 році OpenALPR викупила компанія Rekor Systems, вона отримала готовий продукт за допомогою якого змогла запустити в роботу багато нових рішень.

Переваги:

- можливість користуватися безкоштовно, хоча є і комерційна версія;
- розпізнавання в реальному часі;
- великий набір даних на яких була навчена система, вона розпізнає номерні знаки більш ніж 70-ти країн;
- розробники можуть вбудовувати розпізнавання у свої програми за допомогою api;
- є відкритий код на платформі github;
- оновлення і підтримка.

Недоліки:

- обмежене розпізнавання у складних умовах;
- для локального використання потрібне потужне технічне забезпечення.

Комерційні програми

Вони зазвичай використовуються на підприємствах, яким потрібно мати готовий, високоефективний продукт з технічною підтримкою і оновленнями.

Комерційні програми зазвичай пропонують підвищену точність і меншу кількість багів, ніж програми з відкритим кодом. Тому підприємства обирають саме їх, щоб отримати безпеку, надійність та швидке розгортання програмного забезпечення, без зайвих налаштувань і витрати часу.

Axis Communications ANPR [11] – ця система пропонує апаратне і програмне забезпечення, зазвичай використовується у сферах безпеки, контролю доступу. Компанія славиться своїми камерами відеоспостереження з IP підключенням, технологіями розпізнавання об’єктів, транспортних засобів та їх номерів, також в них є програмне забезпечення для моніторингу та охорони місцевості. Axis Communications - це не просто компанія яка створює різні програми для розпізнавання, вона змогла створити цілу екосистему для підприємств, чи охоронної діяльності.

Переваги :

- великий вибір програмного забезпечення;
- інтегровані апаратні системи;
- можливість роботи в реальному часі;
- надійність і безпека;
- регулярні оновлення;
- технічна підтримка.

Недоліки:

- немає можливості адаптувати програми під свої потреби, вони працюють конкретно за даними алгоритмами і сценаріями;
- висока вартість апаратного і програмного забезпечення;
- якщо підключити камеру від іншого виробника, то програмне забезпечення не буде справно працювати.

Kobi Software [12] – українська компанія, заснована в 2004 році, що спеціалізується на технологіях інтелектуального відеоспостереження. Вона пропонує різні системи безпеки, а основними напрямками є розпізнавання обличь та номерних знаків автомобілів. Ключовою особливістю вибору Kobi Software є можливість інтегрувати своє апаратне і програмне забезпечення з проектом

«Безпечне місто». Вони використовують платформу “ZetPro VMS” для своїх систем спостереження, ця платформа підтримує інтеграцію через API з іншими системами. У цій компанії є також два апаратних пристрої – LPR BOX Cyclops і LPR BOX Orphus/Hydra. Перший пристрій призначений для розпізнавання номерних знаків автомобілів на дорогах при великій швидкості руху, а другий пристрій використовується на автостоянках чи контрольно-пропускних пунктах, автоматично контролює доступ автомобілів.

Переваги:

- швидкість і точність роботи;
- підтримка хмарних і локальних середовищ;
- можливість інтегрувати будь-які ip-камери;
- функція управління шлагбаумами, контроль доступу;
- технічна підтримка.

Недоліки:

- велика вартість на апаратне забезпечення;
- залежність від роздільної здатності зображень;
- складно працювати з api;
- обмежена документація.

1.4 Проблеми розпізнавання номерних знаків автомобілів

Обробка фото і відеоматеріалів для розпізнавання номерних знаків автомобілів є дуже вибагливою, тому ця технологія часто має проблеми з точністю та ефективністю. Слід зазначити, що виявлення номерних знаків це складний процес, який складається з певних умов і етапів, кожен з яких вимагає від системи точної і швидкої роботи. Таблиця 1.2. описує типові проблеми які виникають при розпізнанні номерів транспортних засобів за допомогою комп'ютерного зору, та пропонує можливі варіанти вирішення.

Таблиця 1.2

Аналіз проблем та їх вирішення

Проблема	Опис	Варіанти вирішення
Низька роздільна здатність зображень	Через стиск чи погану якість зображення, ми отримуємо мало деталей для розпізнавання. Контури рамок і символи на номерних знаках автомобільного транспорту – розмиті.	Використовувати камери фото і відеофіксації з більшою роздільною здатністю (починаючи з 1080p). Робити попередню обробку зображень, зменшувати шуми, за потреби підвищувати яскравість і контрастність зображень.
Велика швидкість автомобілів	Розмиття руху автомобілів значно послаблює можливості системи при розпізнаванні. Вона отримує нечіткі кадри автомобільного транспорту, що перешкоджає точному розпізнаванню.	Встановлення камер в яких можна регулювати частоту кадрів. Впровадження алгоритмів компенсації руху .
Неправильно встановлені камери	Камера може бути встановлена з не тою висотою чи не під тим кутом, це робить розпізнавання номерних знаків автомобілів неможливим.	Перевстановити камери. Вимірити поля огляду камер перед встановленням.

Продовження таблиці 1.2

Проблема	Опис	Варіанти вирішення
Шрифти, символи	Багато видів шрифтів, схожість деяких символів ускладнює коректне розпізнавання.	Використовувати датасети для навчання моделей з великою кількістю можливих шрифтів, це зменшить кількість хибних розпізнавань в системі. Налаштувати попередню обробку символів. Використовувати технологію оптичного розпізнавання символів, правильно її налаштувати.
Погода, час доби	Туман, вечірній час доби, опади, вони можуть привести до появлення тіні, зайвого шуму, це суттєво погіршить якість зображень та їх обробку.	Встановлення тепловізерних модулів на камери. Аналізувати погодні умови, час доби і розробити адаптивні налаштування під різні умови.
Погана точність моделі для розпізнавання	Через маленьку кількість тренувальних даних, чи неправильне налаштування моделі розпізнавання вона може давати хибні результати.	Донавчання моделі на датасеті з більшою кількістю даних. Аугментація фото. Використання моделі для розпізнавання разом з іншими технологіями, наприклад – OCR.
Захист даних	Розпізнанні номерні знаки автомобільного транспорту можуть бути викрадені з баз даних підприємств, тому вони мають бути добре захищеними.	Шифрування збережених результатів розпізнавання. Використання хмарних сервісів для зберігання. Встановити пароль для бази даних, організувати права доступу.

Продовження таблиці 1.2

Проблема	Опис	Варіанти вирішення
Перешкоди	Деякі об’єкти можуть заважати розпізнаванню номерних знаків, це можуть бути: дерева, кущі, рекламні щити.	Розробка сповіщень, коли детекція виконана не повністю. Встановлення ще однієї камери під іншим кутом. Навчити модель розпізнавати можливі перешкоди.
Розпізнавання номерних кодів різних країн	Колір, розмір, структура номерних знаків в різних країнах може суттєво відрізнятися. Тому деякі вже існуючі системи не можуть працювати з різними номерними знаками, вони розроблені конкретно під свою країну	Використовувати базу даних з різними варіаціями номерних знаків для донавчання нейронної мережі. Розробити алгоритм, який попередньо визначає регіон і визначені параметри для кожного формату номерних знаків.

Висновки до розділу 1

У першому розділі було розглянуто теоретичні засади комп’ютерного зору, особливості його застосування для задач розпізнання номерних знаків транспортних засобів. Детально проаналізовано історію розвитку технологій комп’ютерного зору, методи попередньої обробки зображень для подальшого їх використання у системах розпізнання номерних знаків автомобільного транспорту. Було окреслено практичні сфери використання систем розпізнавання номерних знаків, серед яких: контроль доступу, моніторинг трафіку, платне паркування, виявлення викрадених автомобілів тощо. Також особливу увагу приділено до огляду існуючі програмних рішень, що допомогло оцінити рівень розвитку технологій розпізнавання номерних знаків, виявити їх сильні і слабкі сторони.

2 АНАЛІЗ МЕТОДІВ ТА ТЕХНОЛОГІЙ ДЛЯ РОЗПІЗНАВАННЯ НОМЕРНИХ ЗНАКІВ ТРАНСПОРТНИХ ЗАСОБІВ

2.1 Технології та алгоритми ідентифікації номерних знаків

У галузі комп'ютерного зору існує багато методів і технологій для ідентифікації номерних знаків автомобілів. Основна увага розробників програмного забезпечення приділяється технологіям глибокого навчання та методам оптичного розпізнавання символів. Реалізація такого додатка вимагає від розробника створення правильного алгоритму роботи, це включає в собі виявлення номерного знаку автомобільного транспорту, сегментацію символів, їх розпізнавання за допомогою OCR. На рисунку 2.1 зображений алгоритм поетапного розпізнавання в готовій системі. Він включає в собі: попередню обробку зображення(згладжування шуму, корегування розміру зображення перед відправкою на обробку моделі), детекцію рамки номерного знаку на отриманому зображенні за допомогою попередньо навченої моделі, вона аналізує зображення, прогнозує можливе розташування рамки номерного знаку автомобіля, вирізає із зображення область розташування рамки номерного знаку, сегментує символи перетворюючи у двоколірне зображення, знаходить окремі символи. Після сегментації вирізане зображення з номерним знаком передається на обробку для визначення символів, для цього використовується технологія оптичного розпізнавання (OCR). Вона аналізує кожен символ, перевіряє кут нахилу тексту, після чого ми отримаємо текстовий результат який може виводитись у наш графічний інтерфейс, чи зберігатись у базі даних.

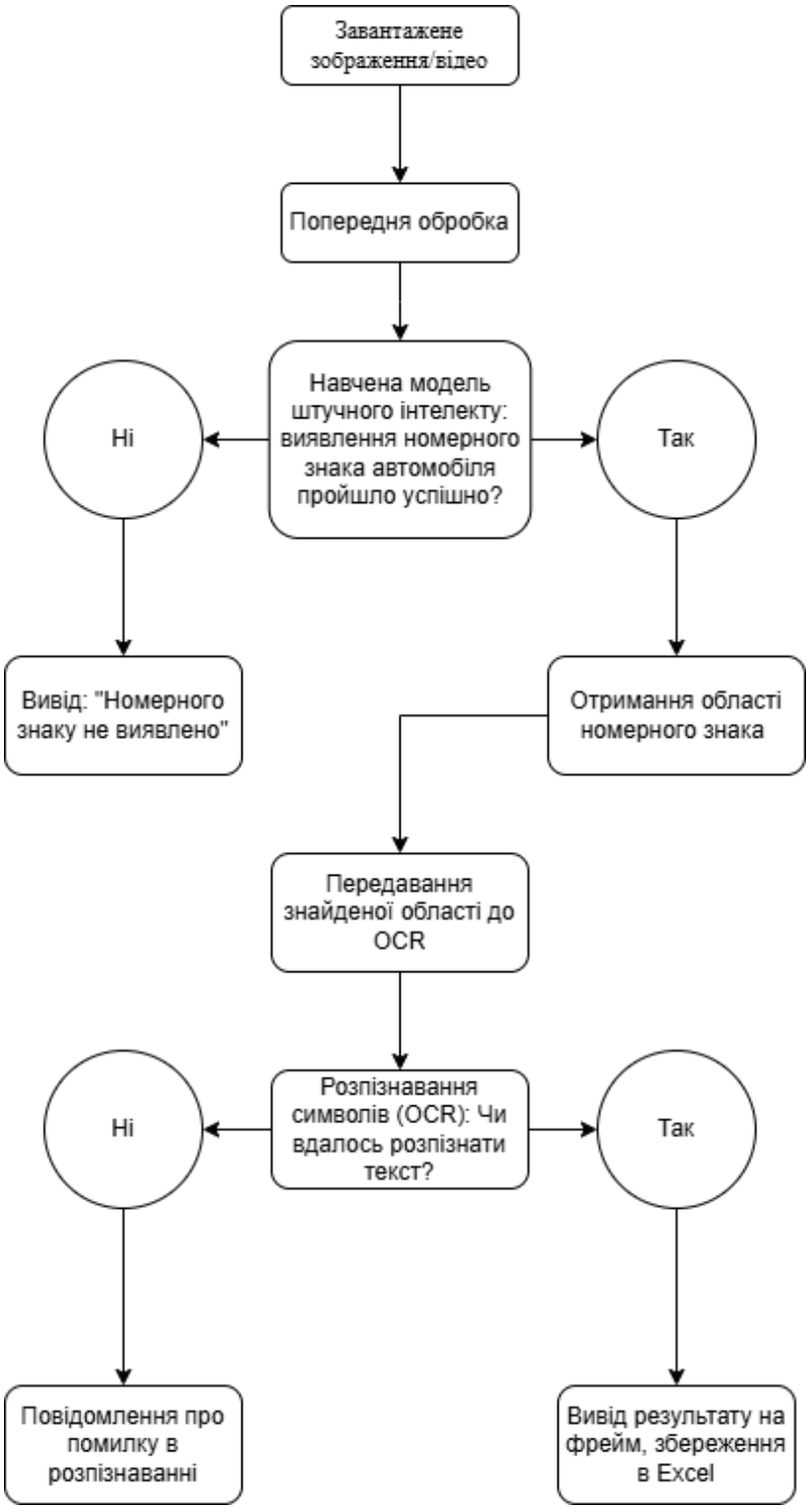

Рис. 2.1 Системна блок-схема етапів розпізнавання номерних знаків

У результаті ми можемо побачити готову систему з добре визначеним алгоритмом роботи, з гарно визначеною структурою та чітко розмежованими функціональними модулями. Кожен із яких відповідає за свій крок і робить подальші етапи роботи зв'язаними між собою, що дозволяє працювати системі в режимі реального часу.

2.2 Моделі штучного інтелекту для виявлення об'єктів

Для знаходження й розпізнавання номерних знаків автомобілів використовуються сучасні моделі глибокого навчання. Вони допомагають визначити місцезнаходження об'єктів на зображеннях або відео, зробити їх класифікацію. Моделі штучного інтелекту засновуються на згорткових нейронних мережах, завдяки ним можна поєднувати точність і швидкість в обробці даних. На сьогоднішній день з'явилась достатня кількість моделей штучного інтелекту, але кожні з них відрізняються способами призначення, швидкістю обробки даних, точністю.

2.2.1 Модель YOLO (You Only Look Once)

Модель штучного інтелекту Yolo – є однією із самих популярних для задач виявлення об'єктів. Вона поєднує в собі багато переваг, що відрізняє її від інших існуючих моделей розпізнавання. Завдяки принципу її роботи, який використовує обробку даних лише один раз, за крок [13], під час якого виконуються всі обчислення для об'єктів на зображенні, YOLO забезпечує високу швидкість, дозволяє працювати і обробляти зображення й відеозаписи з мінімальною затримкою, а інші моделі можуть вимагати від користувача декількох етапів обробки, що зробить роботу системи повільнішою. Це робить YOLO самим ефективним вибором для програмного забезпечення, метою якого є автоматичне виявлення об'єктів у реальному часі. Ця модель пройшла велику кількість оновлень за свої роки існування. На рисунку 2.2 можна побачити як прогресувала швидкість роботи YOLO з кожним оновленням. Самою потужною на даний час є версія – v11.

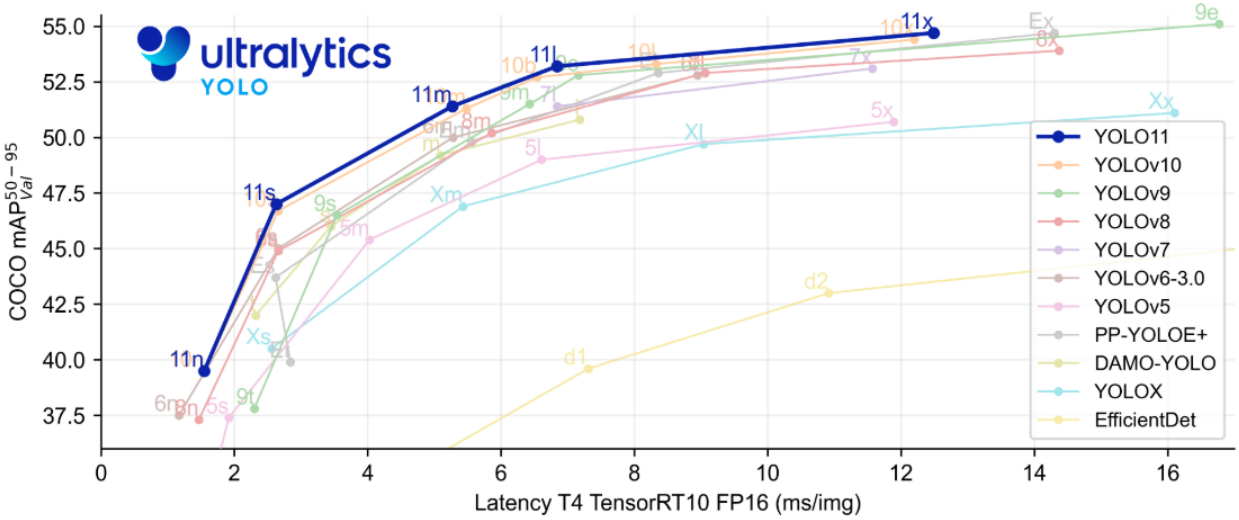

Рис. 2.2 Швидкість роботи моделей YOLO

На рисунку 2.3 продемонстрована архітектура YOLOv11, на ній можна побачити як доцільно працює ця модель. Вона складається з трьох основних компонентів – згорткова нейронна мережа, проміжна структура, вихідна частина. Модель отримає зображення, поділяє його на окремі області у вигляді клітинок і кожна з них використовується для виявлення об'єктів.

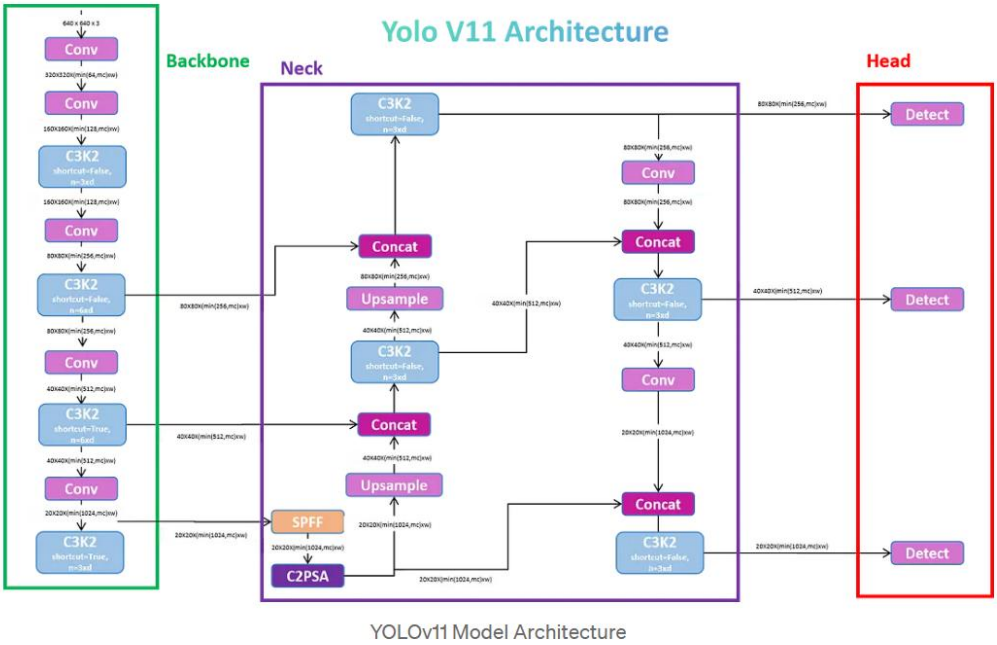

Рис. 2.3 Архітектура YOLOv11

Щоб використовувати модель YOLO у своїх проектах, її потрібно попередньо навчити на потрібному наборі даних. Як зазначено в джерелі [16] треба створити датасет з анотованими зображеннями. Для навчання моделі кожна фотографія повинна мати текстовий файл з анотаціями, які підтримують формат YOLO. Збирати датасет своїми руками і анотувати кожне зображення - дуже трудомістке заняття, яке потребує багато часу, тому існує велика кількість якісних, вже готових датасетів для використання. Одним із таких наборів даних є Car Dataset [17], він містить в собі 10 000 зображень автомобільного транспорту з анотаціями до кожного з них.

Основні характеристики та переваги цього набору даних:

- Кількість зображень: 10000 різноманітних зображень, які розділені на тренувальну, валідаційну та тестувальну вибірки.
- Різноманітність зображень: містить кадри під різними кутами нахилу, з різними формами номерних знаків, а також включає в собі номерні знаки різних країн.
- Формати завантаження: підтримує всі поширені формати – YOLO, COCO, JSON, XML.
- Сумісність з фреймворками комп'ютерного зору: YOLOv11, Detectron2, PyTorch, Tensorflow, тощо.

2.2.2 Аналіз інших моделей порівняно з YOLO

Вибір моделі YOLO для системи розпізнавання номерних знаків транспортних засобів засновується на високій швидкості обробки зображень, що допомагає працювати системі у реальному часі без перебоїв, достатня точність виявлення рамки номерів автомобілів, мінімальне витрачання ресурсів комп'ютера при навчанні і використанні моделі. У таблиці 2.1 наведено порівняння YOLO з іншими популярними моделями для розпізнавання об'єктів.

Таблиця 2.1

Порівняння популярних моделей для виявлення об'єктів

	YOLOv11	RetinaNet	Faster R-CNN	SSD	EfficientDet
Точність	Висока	Висока	Дуже Висока	Середня	Висока
Швидкість	Дуже висока	Середня	Низька	Висока	Низька
Складність в навчанні і реалізації	Низька	Середня	Висока	Низька	Висока
Виявлення маленьких об'єктів	Висока	Середня	Висока	Середня	Середня
Апаратні вимоги	Середні	Середні	Високі	Середні	Високі

Проаналізувавши результати, можна побачити що оптимальним вибором є модель YOLO, в неї самі збалансовані показники і найвища швидкість роботи, це потрібно для виявлення номерних знаків в реальному часі. Хоча Faster R-CNN славиться своєї точністю і він краще виявляє дрібні деталі на зображеннях, але для задач розпізнавання в реальному часі він не підходить, тому що ця модель залежить від алгоритму вибіркового пошуку, що значно сповільнює процес знаходження об'єкта.

2.3 Технологія оптичного розпізнавання символів

Технологія оптичного розпізнавання символів (OCR) широко використовується в різних задачах – зчитує текст з документів, фотографій, у нашому випадку з рамок номерних знаків автомобільного транспорту.

Застосування цієї технології у програмному забезпеченні для розпізнавання використовується для ідентифікації символів на вирізаному зображені рамки номера, що зробила навчена модель штучного інтелекту і передала її на обробку до OCR.

Огляд етапів роботи технології оптичного розпізнавання символів:

- Обробка отриманого зображення – отримане зображення з камери повинно обробитись для підвищення чіткості символів і зменшенню шумів. Розмиття символів усуваються за допомогою цифрових фільтрів і вирівнюється контрастність зображення щоб освітлення було рівномірним. Зображення на рисунку 2.4 перетворюється в чорно-білий формат через бінаризацію, це допомагає знайти область розташування номерного знаку [18].

Рис 2.4 OpenCV та бінаризація Otsu

Також можуть бути використані морфологічні операції, щоб темні літери були білими, а задній фон чорним, це полегшить роботу розпізнавання, такий метод обробки представлений на рисунку 2.5

Рис 2.5 Використання морфологічних операцій до зображення

- Виявлення області – далі система повинна визначити рамку номера автомобільного транспорту за допомогою аналізу зображення і виділення регіону що містить символ. Моделі глибокого навчання знаходять координати цієї області і передають до OCR.
- Розбиття виявленого тексту на окремі символи – OCR розбиває текст на символи.
- Перевірка текстових областей – деякі системи налаштовують додаткову перевірку правильності виділення області з текстом, завдяки цьому можна мінімізувати кількість помилок при розпізнаванні.
- Класифікація символів – кожен знайдений символ аналізується і перетворюється на машинозчитуваний текст.

- Корекція отриманого тексту – після розпізнавання текст може перевірятись за якимось заданим форматом, наприклад за довжиною номерного знаку автомобільного транспорту, ця дія допоможе уникнути зайвих символів і зробити роботу системи точнішою.
- Отримання вихідних даних – на цьому етапі ми отримаємо розпізнаний текст, який перетворений у цифровий формат і записаний в базу даних, або в Excel файл, також його можна відобразити в інтерфейсі програмного забезпечення.

2.3.1 Бібліотека PaddleOCR

У технології оптичного розпізнавання символів існує багато фреймворків, таких як EasyOCR, PaddleOCR, Tesseract, FineReader, тощо. Вони активно використовуються в задачах розпізнавання номерних знаків автомобільного транспорту. PaddleOCR – це сучасна технологія яка підтримує понад 80 мов різних країн і має гнучкі параметри налаштування, що підходить для кожного користувача який прагне створити свій програмний продукт [19]. Інтегрування у свої проекти може здійснюватися за допомогою звичайної команди «pip install», це значно спрощує його використання у своїх програмних проектах.

У системі розпізнавання номерних знаків автомобільного транспорту архітектура PaddleOCR повинна бути правильною та структурованою, це буде забезпечувати точне зчитування символів на зображеннях. Є три основні компоненти з який вона складається:

- Виявлення тексту – перший етап роботи OCR, алгоритм Differentiable Binarization визначає яка область символів на зображенні номерного знаку.
- Класифікація кута нахилу – другий етап визначає під яким кутом знаходиться виділена область з символами на номері автомобіля
- Розпізнавання символів – третій етап перетворює розпізнанні символи у текстовий формат для виводу результатів, це виконується за допомогою нейронних мереж розпізнавання CNN і RNN.

На рисунку 2.6 представлена діаграма роботи PaddleOCR в системі розпізнавання номерних знаків, з її ключовими компонентами.

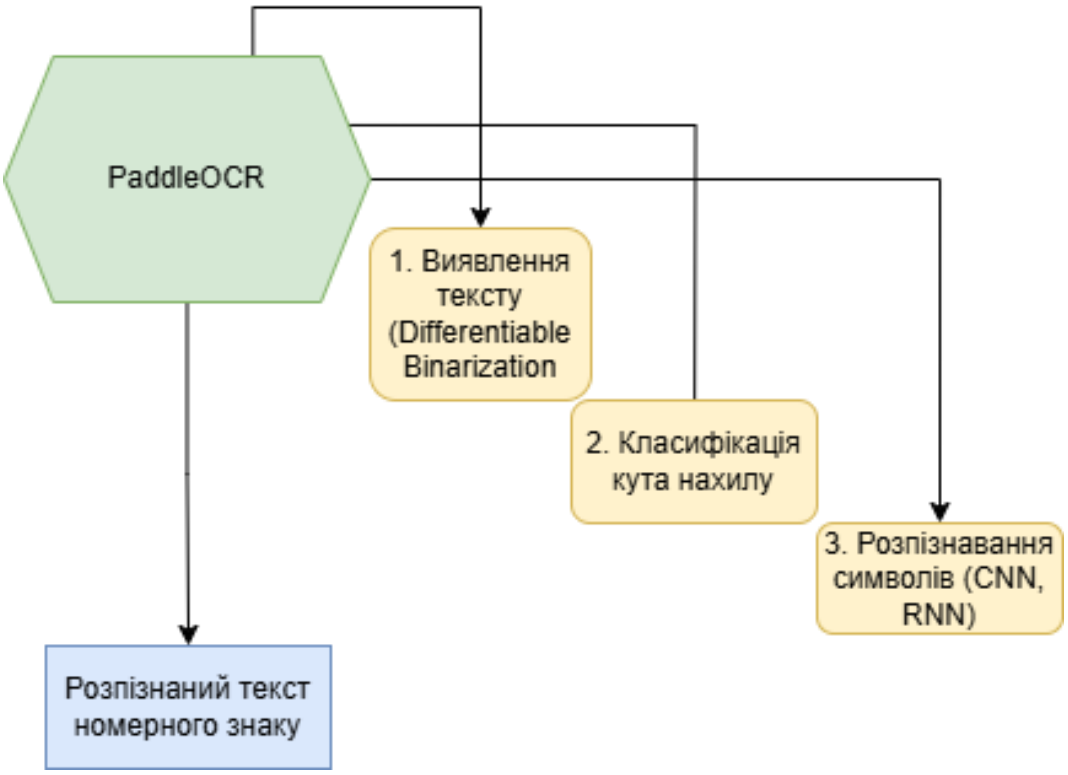

Рис 2.6 Архітектура PaddleOCR у системі розпізнавання номерних знаків автомобільного транспорту

2.3.2 Порівняння з іншими OCR

PaddleOCR є ефективним інструментом для автоматичного моніторингу номерних знаків автомобільного транспорту. Але існує ще ряд інших варіантів, кожен з який теж активно використовується в галузі ідентифікації номерів. На розробника випадає важливий вибір, яку саме технологію для реалізації програмного забезпечення обрати, цей вибір може засновуватися на швидкості роботи, точності, легкому інтегруванні у свої системи. В таблиці 2.2 описані ключові переваги і недоліки популярних OCR рішень.

Таблиця 2.2

Огляд характеристик інструментів оптичного розпізнавання символів.

	PaddleOCR	Tesseract	FineReader	EasyOCR
Підтримка розпізнавання різних мов	80 і більше	100 і більше	190 і більше	80 і більше
Точність	Висока	Висока але потрібно багато налаштувань	Висока	Середня
Швидкість обробки	Висока	Середня	Низька	Висока
Стійкість до нахилів	Висока	Низька	Висока	Середня
Переваги	Має дуже високу точність, підтримку роботи як з CPU так і з GPU, легко налаштовується й інтегрується в проекти, має можливість донавчання.	Підтримує багато мов, гнучке налаштування	Висока точність обробки, підтримує більше всього мов.	Легкий у використанні, має велику швидкість обробки.
Недоліки	Велика вага моделі, може бути потрібно потужна техніка для донавчання.	Мала точність розпізнавання якщо не має ретельно налаштованої обробки зображення, також складний в налаштуванні.	Висока вартість, низька швидкість, складна інтеграція у свої проекти.	Погана адаптація до шрифтів, низька точність при поганих умовах.

Найбільш доцільним вибором для розпізнавання номерних знаків в реальному часі є PaddleOCR. Він поєднує в собі швидкість, точність і легку інтеграцію у свої проекти.

2.4 Розробка моделі моніторингу транспортних засобів на підприємствах

Автоматизація процесу ідентифікації номерних знаків транспортних засобів на підприємствах грає важливу роль. З кожним роком з'являється все більше автомобільного транспорту і контролювати їх рух через контрольно-пропускні пункти стає складним завданням для звичайної людини. Впровадження системи автоматизованого розпізнання номерних знаків на підприємствах допоможе зробити робочі процеси більш ефективними, система зменшить час очікування автомобільного транспорту на контрольно-пропускних пунктах, усилить точність ведення обліку транспортних засобів які відвідують підприємство. Ці аспекти мають значно підвищити продуктивність роботи.

Для реалізації такої системи на підприємствах, потрібне не тільки програмне забезпечення, а ще й технічне, це забезпечить повноцінну роботу моніторингу номерних знаків транспортних засобів. На рисунку 2.7 продемонстрована робота системи моніторингу з інтеграцією роботи камери в програмне забезпечення.

Рис 2.7 Детекція номерних знаків на контрольно-пропускних пунктах

Камера фіксує номерний знак автомобільного транспорту. Він передається в програмне забезпечення з модулем розпізнавання, проходить обробку, після чого активується механізм шлагбаума, який надає доступ для проїзду транспортному засобу. Враховуючи ці етапи і результати аналізу технологій з розділів 2.2.2 і 2.3.2, була розроблена власна архітектурна модель системи розпізнавання номерних знаків автомобільного транспорту, вона містить всі користувальницькі функції які можуть бути реалізовані в робочих процесах моніторингу транспортних засобів на підприємствах, це зображено на рисунку 2.8.

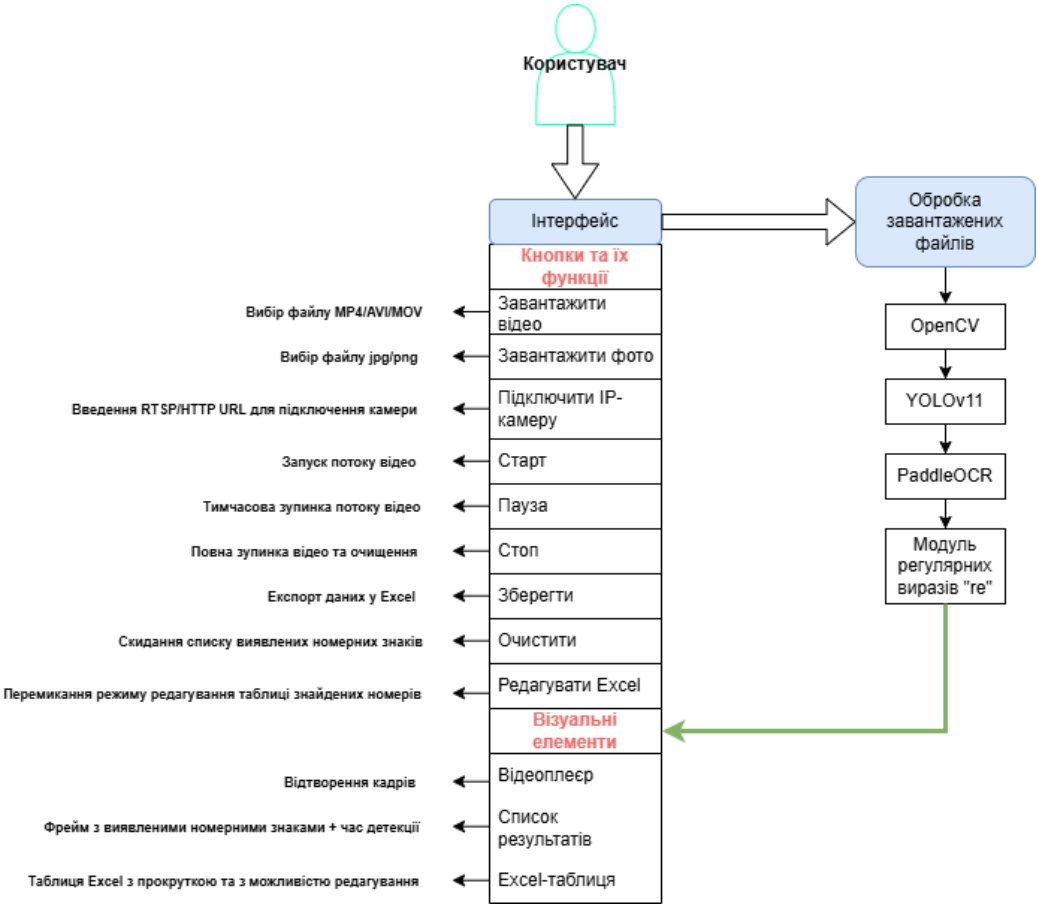

Рис 2.8 Модель моніторингу транспортних засобів

Взаємодія із системою розпочинається з користувача, який здійснює контроль затранспортними засобами на підприємстві. В системі передбачений інтуїтивно зрозумілий графічний інтерфейс, за допомогою якого можна обирати потрібні дії:

- Звантажувати відео
- Завантажувати фото
- Підключати IP-камеру

- Розпочинати або зупиняти обробку
- Переглядати і редагувати виявлені результати

Після завантаження потрібних файлів або підключення IP-камери, система переходить до етапу розпізнавання. За допомогою бібліотеки OpenCV здійснюється захоплення кадрів, їх підготовка до аналізу, потім кожен кадр передається у модель YOLO, виконується детекція області розташування номерного знаку автомобільного транспорту. Виявлена область передається в модуль PaddleOCR, для розпізнавання символів. Після отримання тексту відбувається очищення від зайвих символів за допомогою регулярних виразів. Знайдені номерні знаки зберігаються у внутрішній структурі програмного забезпечення і виводяться на графічний інтерфейс з зафіксованою датою та часом розпізнавання. Знайдені системою номерні знаки можна зберегти у форматі excel. За потреби користувач може відкрити вбудовану в програму панель редагування знайдених номерних знаків автомобільного транспорту для фіксації і подальшого внесення змін.

Висновки до розділу 2

Було проведено ретельний аналіз методів і алгоритмів, які використовуються для розпізнання номерних знаків транспортних засобів. Основи сучасних систем розпізнавання засновуються на глибокому навчанні та оптичному розпізнаванні символів. Особливу увагу було приділено моделі YOLO та PaddleOCR, було порівняно їх з іншими технологіями для детекції об'єктів і зчитування тексту, вони відрізнились своєю високою швидкістю і точністю в розпізнаванні, що підтвердило їх переваги і оптимальність використання в системах розпізнавання. На основі проаналізованих технологій було розроблено архітектурну модель моніторингу транспортних засобів на підприємствах, яка містить в собі оптимальні технології комп'ютерного зору і забезпечує автоматизацію виявлення номерних знаків.

3 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Вибір мови програмування та середовища розробки

У сфері машинного навчання і розпізнавання образів можуть бути використані різні мови програмування і треба зробити правильний вибір, щоб швидкість і якість роботи створеного програмного забезпечення була високою. Для реалізації проекту я використовував мову програмування Python, яка є основною при роботі з комп'ютерним зором. Вона підтримує всі сучасні інструменти розробки штучного інтелекту, може інтегруватись у програмні забезпечення, які написані на інших мовах програмування, налічує велику кількість посібників і ресурсів по використанню, що полегшує роботу при написанні програмного забезпечення різної складності.

Вибір середовища розробки залежить від різних факторів, це можуть бути потреби системи, підтримка інтеграцій потрібних фреймворків, потужність технічного обладнання тощо. У процесі розробки програмного забезпечення я використовував Google Colaboratory і Pycharm.

Google Colaboratory це хмарне середовище розробки, воно використовувалось для швидкого навчання моделі YOLO і зручної інтеграції набору даних для навчання. Google Colaboratory дозволяє зберігати сам проект і всі файли в ньому в хмарі. Також компанія Google надає безкоштовний доступ до використання графічного процесору при виконанні своїх проектів, це зображено на рисунку 3.1

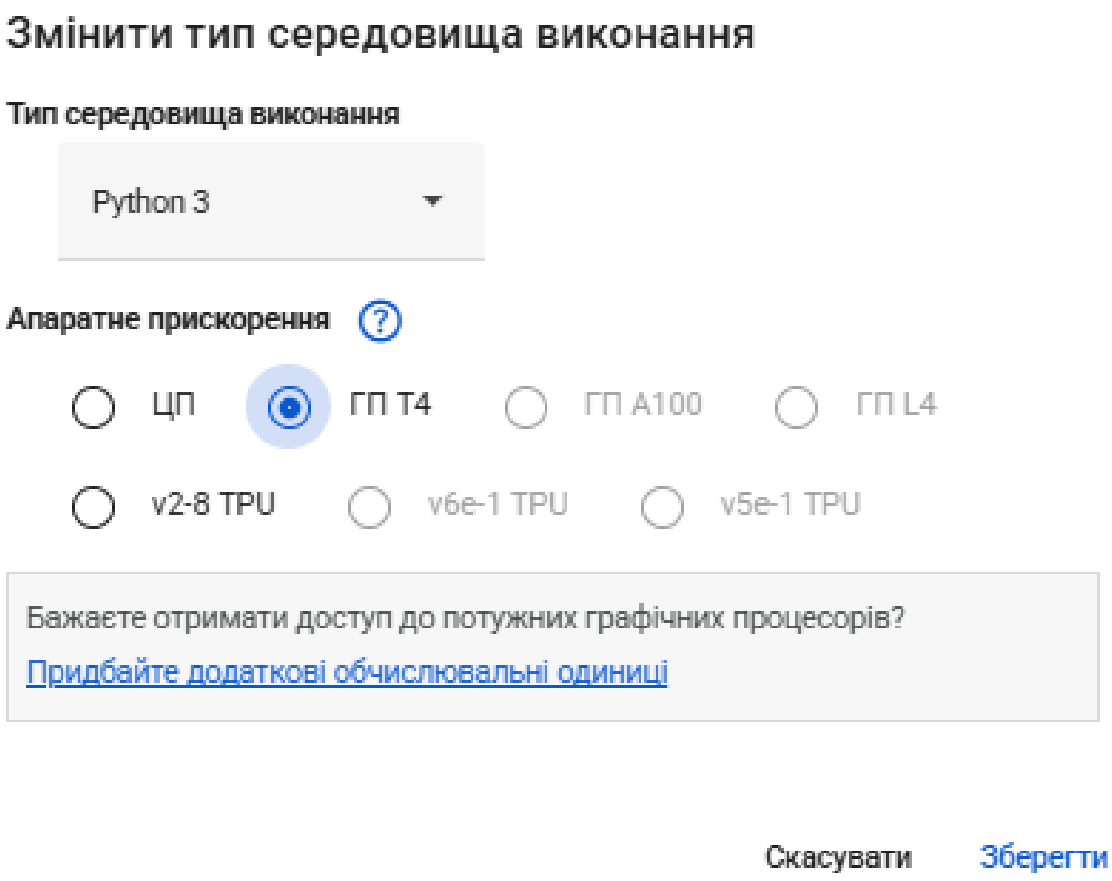

Рис. 3.1 Тип середовища виконання Google Colaboratory

Ця функція допоможе прискорити навчання нейронних мереж без використання апаратних ресурсів, які при задачах з великими обсягами даних мають велике навантаження. Для навчання моделі YOLO я використовував графічний процесор від Google Colaboratory, що дало змогу скоротити час обробки даних і підвищити якість тренування моделі не маючи потужного технічного обладнання.

Для реалізації прикладної частини програмного забезпечення я використовував PyCharm. В нього вбудована велика кількість різних можливостей, серед основних переваг PyCharm можна виділити:

- Вбудоване автодоповнення коду, що зменшує кількість помилок при написанні програмного забезпеченні і підвищує швидкість.
- Інструменти тестування, вони допомагають аналізувати помилки в роботі програми і усувати їх.

- Можливість інтегрувати всі популярні фреймворки, які були використані при розробці проекту
- Віртуальне середовище, за допомогою якого можна мати багато проектів, бібліотеки, які використовуються в них, не будуть конфліктувати між собою

Отже вибір мови програмування і середовищ розробки обумовлюється ефективністю при машинному навчанні та зручністю роботи при написанні основного коду для системи розпізнавання номерних знаків автомобільного транспорту.

3.2 Огляд використаних бібліотек

Для якісного функціонування програмного забезпечення було використано сучасні бібліотеки, вони забезпечують обробку зображень, роботу з відео і IP-камерами, ідентифікацію області номерного знаку автомобільного транспорту та визначення символів на них. На лістингу 3.1 продемонстровані бібліотеки які використовувались у проекті.

Лістинг 3.1 Імпорт потрібних бібліотек

```
import tkinter as tk
from tkinter import filedialog, Label, Canvas, Frame, Scrollbar, ttk, messagebox
from ultralytics import YOLO
from PIL import Image, ImageTk
from paddleocr import PaddleOCR
import re
from collections import defaultdict
import cv2
import pandas as pd
from datetime import datetime
```

- OpenCV – використовується для попередньої обробки зображень, зчитування потоку на відео і IP-камерах.
- Ultralytics YOLO – знаходить область розташування рамки номерного знаку автомобільного транспорту, може працювати в реальному часі.
- PaddleOCR – бібліотека оптичного розпізнавання символів, вона отримує знайдену область і зчитує на ній символи, працює навість при складних умовах, таких як: дощ, туман, нахил зображення.
- Re – це стандартна бібліотека регулярних виразів, вона використовується в проекті для очищення зайвих символів, які не характерні для номерних знаків, це значно підвищує точність при розпізнаванні.
- Tkinter – з її допомогою створювався графічний інтерфейс програмного забезпечення.
- Pandas – це бібліотека для роботи з даними. У програмі вона використовується для створення таблиць знайдених номерних знаків і збереженні їх у форматі Excel.
- Pillow – використовується при роботі з форматом зображень. Отримує зображення через OpenCV і перетворює в сумісний формат для відображення їх на графічному інтерфейсі програми.
- DateTime – дозволяє фіксувати точний час і дату розпізнавання кожного номерного знаку автомобільного транспорту, він відображається у списку результатів поряд з виявленим номером.
- Collections – дозволяє зберігати знайдені номерні знаки у контейнерах. Вони зберігаються, проводиться ініціалізація кожного значення, що дозволяє вести підрахунок кількості виявлених номерних знаків, робити фільтрування за кількістю повторів знайдених номерів, а також сортувати їх у кінцевому спинку.

Завдяки такому набору технологій було реалізовано програмне забезпечення для розпізнавання номерних знаків транспортного засобів.

3.3 Тренування моделі YOLO

Для виявлення номерних знаків автомобільного транспорту використовувалась модель YOLOv11, вона навчалась на наборі даних, детально описаному в розділі 2.2.1, він містить 10 000 анотованих зображень автомобільного транспорту з різноманітним освітленням, кутами зйомки. На рисунку 3.2 зображено виконані кроки для тренування моделі YOLOv11 з використанням інтеграції набору даних з платформи Roboflow [17] в середовище розробки Google Colaboratory.

Рис 3.2 інтеграція Roboflow та навчання моделі YOLO

- Встановлення необхідних бібліотек, що дозволяють працювати з Roboflow і YOLO
- Використання API Roboflow з ключем доступу, що дозволяє використовувати обраний набір даних для тренування
- Завантаження набору даних Car Dataset з робочого простору платформи Roboflow
- Інтеграція з Google Drive, для збереження результатів тренування моделі.

- Та запуск процесу навчання моделі YOLO, він має такі параметри : 100 епох тренування, фіксований розмір зображення 640x640 пікселів і 8 потоків обробки. На рисунку 3.3 продемонстровано початок тренування моделі

```
Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
1/100        0G     0.9672      1.914      1.167         25        640: 100% 441/441 [1:39:08<00:00, 13.49s/it]
           Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 64/64 [11:30<00:00, 10.78s/it]
             all       2032       2218      0.948      0.916      0.945      0.681

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
2/100        0G      0.988      1.033      1.152         22        640: 100% 441/441 [1:41:04<00:00, 13.75s/it]
           Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 64/64 [11:35<00:00, 10.87s/it]
             all       2032       2218       0.92      0.846      0.891      0.631

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
3/100        0G      1.065     0.9622      1.199         20        640: 100% 441/441 [1:40:31<00:00, 13.68s/it]
           Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 64/64 [11:07<00:00, 10.43s/it]
             all       2032       2218      0.895      0.787      0.857      0.601

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
4/100        0G      1.103     0.9386      1.233         27        640:  69% 303/441 [1:09:03<32:24, 14.09s/it]
```

Рис 3.3 Тренування моделі YOLO

Після завершення тренування, модель збережеться в папці у хмарі, в файлі best.pt, який інтегровано в основну програму для подальшого використання. Також в папці з моделлю будуть її метрики, за допомогою яких було оцінено якість тренування. На рисунку 3.4 зображені метрики для оцінки моделі. Основна метрика, що демонструє показник якості виявлення номерних знаків досягає відмітки 0.98 на етапі тренування і валідації, що говорить про дуже низьку можливість помилки при розпізнаванні, а також під час тренування можна побачити як зменшуються функції витрат і зростають метрики, це свідчить про ефективний процес навчання моделі.

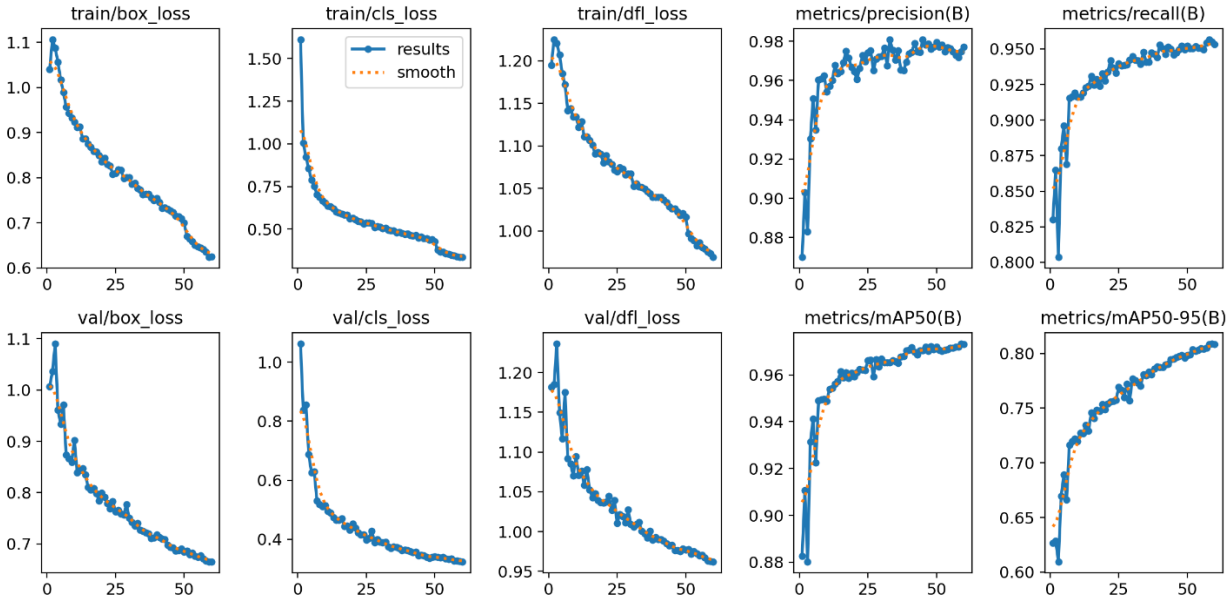

Рис 3.4 Метрики після тренування моделі YOLO

3.4 Розробка графічного інтерфейсу

Розробка графічного інтерфейсу проводилась з урахуванням інтуїтивності і простоти у використанні. Для забезпечення зручної взаємодії користувача з готовою системою для розпізнавання номерних знаків автомобільного транспорту, було розроблено попередню модель системи із визначеними функціями кожної з кнопок керування, що зазначені на рисунку 2.8. Ця модель стала основою для впровадження функціонального інтерфейсу у програмне забезпечення.

На рисунку 3.5 зображений графічний інтерфейс системи розпізнавання номерних знаків. Для забезпечення стабільної роботи інтерфейсу було використано фіксовані розміри головного вікна (1200x700 пікселів).

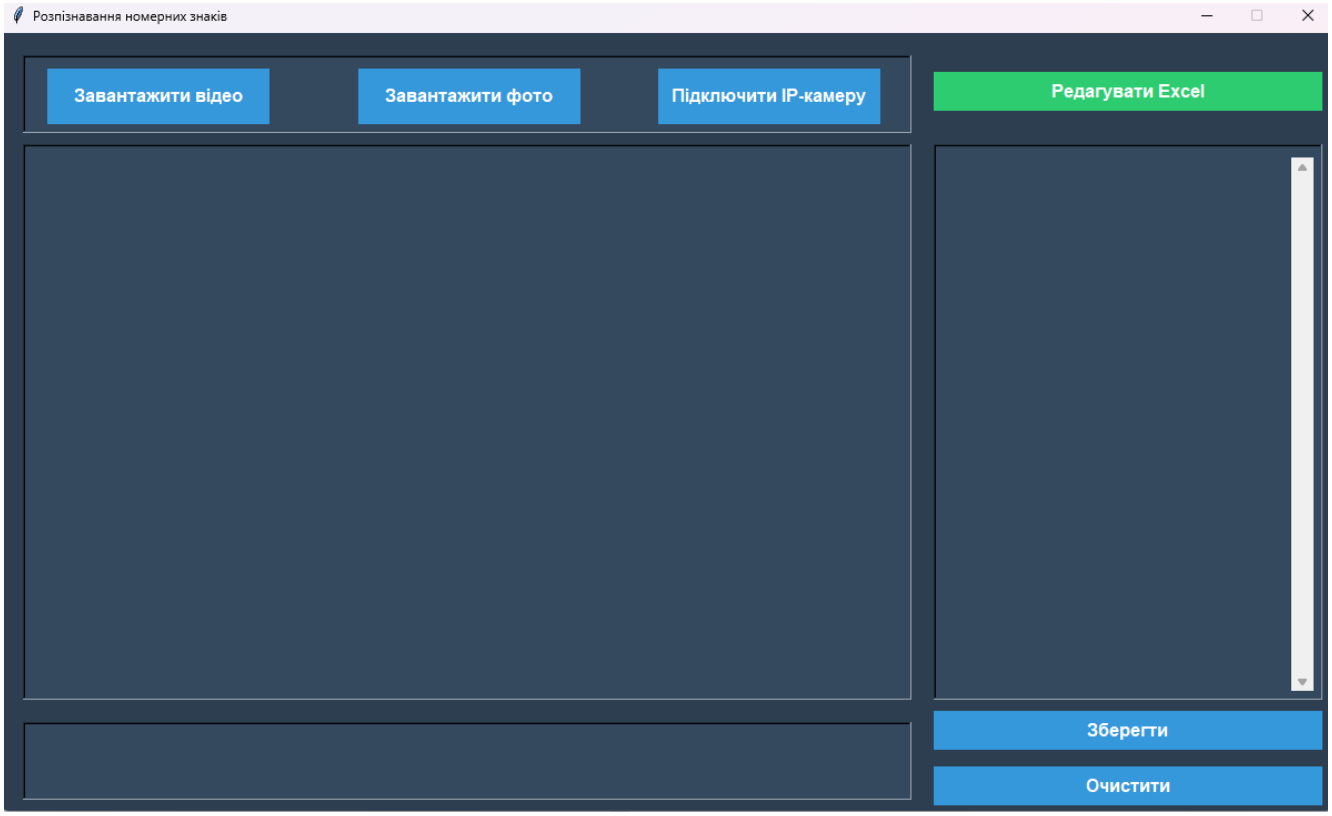

Рис 3.5 Графічний інтерфейс програмного забезпечення

Інтерфейс реалізовано за допомогою класу LicensePlateRecognitionApp, він містить всю логіку програми, а використання бібліотеки Tkinter дозволило використовувати у проекті графічні компоненти, такі як Frame, Label, Canvas, Button, Scrollbar та Treeview. Щоб зробити графічний інтерфейс більш цікавим і сучасним було використано стилі ttk.Style, які налаштовують шрифти, кольори і поведінку кнопок при взаємодії з користувачем. Графічний інтерфейс розподілений на декілька зон, в який знаходяться графічні компоненти, кожна зона має свою роль, що дозволяє користувачу легко орієнтуватися в системі. Наприклад, на рисунку 3.5 знизу залишилася пуста зона, але коли користувач завантажить відео файл чи підключить IP-камеру з'являться кнопки керування (Старт, Пауза, Стоп), таке рішення запобігає випадковим діям користувача, що робить інтерфейс більш зрозумілим.

Основні фрейми програми:

- Фрейм для завантаження файлів – він розташований у верхній частині вікна програми, він містить в собі три кнопки, а саме: завантаження зображень,

відео, та підключення IP-камери. Кожна розташована кнопка викликає відповідну функцію для вибору файлу через діалогове вікно. Ця зона служить початковим етапом в користуванні системи. На лістингу 3.2 наведений код побудови фрейму з кнопками.

Лістинг 3.2 Ініціалізація верхнього фрейму для вибору джерела даних

```
self.style_frame = {"bg": "#34495E", "bd": 2, "relief": "sunken"}
self.frame_top = Frame(root, **self.style_frame)
    self.frame_top.place(x=20, y=20, width=800, height=70)
    self.btn_upload_video = ttk.Button(
        self.frame_top,
        text="Завантажити відео",
        style="Custom.TButton",
        command=self.upload_video
    )
    self.btn_upload_video.place(x=20, y=10, width=200, height=50)
    self.btn_upload_photo = ttk.Button(
        self.frame_top,
        text="Завантажити фото",
        style="Custom.TButton",
        command=self.upload_photo
    )
    self.btn_upload_photo.place(x=300, y=10, width=200, height=50)
    self.btn_connect_camera = ttk.Button(
        self.frame_top,
        text="Підключити IP-камеру",
        style="Custom.TButton",
        command=self.connect_ip_camera
    )
    self.btn_connect_camera.place(x=570, y=10, width=200, height=50)
```

- Фрейм для відображення завантажених файлів – призначений для відображення поточного зображення або відеопотока з позначеними на них рамками розпізнаних номерних знаків автомобільного транспорту. В центральній частині цього фрейму розміщена мітка Label, вона є основною для відображення процесу розпізнавання у реальному часі. На рисунку 3.6 продемонстровано завантажене зображення і відтворення його на фреймі.

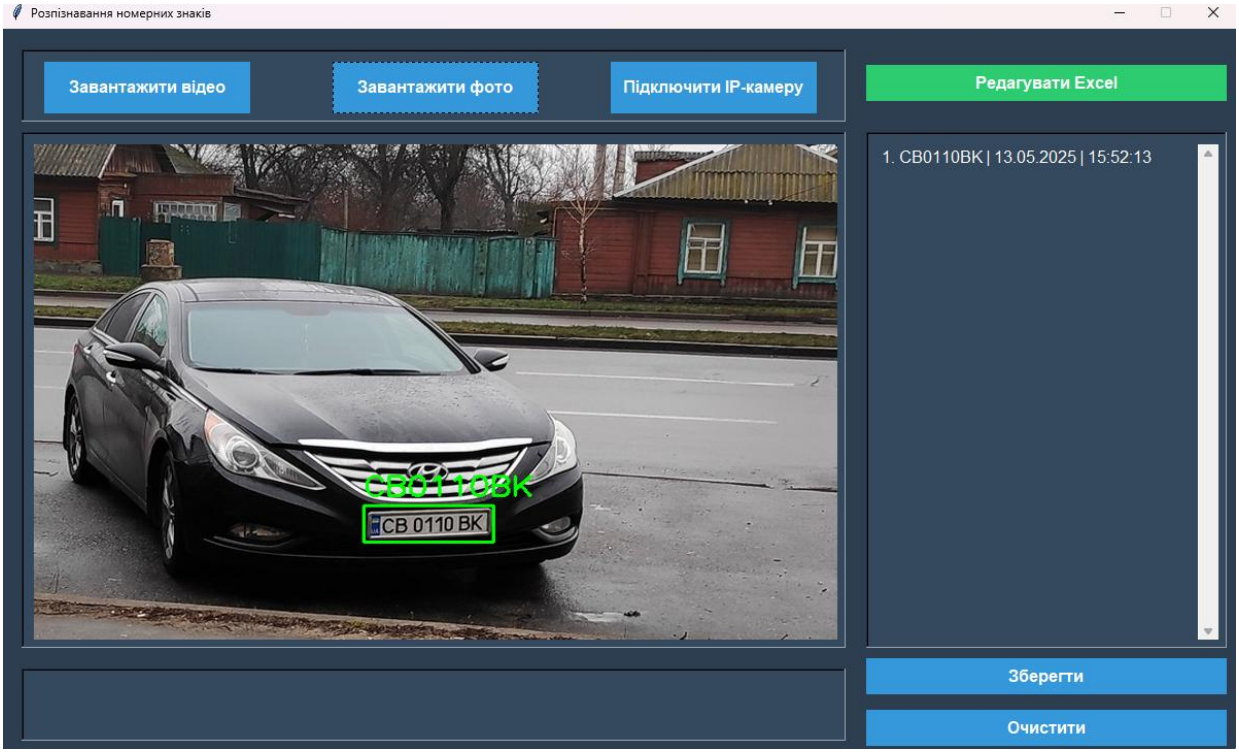

Рис 3.6 Відображення поточного зображення

На лістингу 3.3 наведений код побудови фрейму для відображення завантажених файлів

Лістинг 3.3 Центральний фрейм для відображення завантажених файлів

```
self.frame_video = Frame(root, **self.style_frame)
self.frame_video.place(x=20, y=100, width=800, height=500)
self.video_label = Label(self.frame_video, **self.style_label)
self.video_label.place(x=10, y=10, width=780, height=480)
```

- Фрейм для відображення результатів розпізнавання – це панель яка розташована праворуч, від центрального фрейму, вона призначена для відображення списку розпізнаних номерних знаків із датою та часом їх виявлення, це зображено на рисунку 3.7 . Вона також має прокрутку, що дозволяє переглядати велику кількість знайдених номерних знаків, що дуже потрібно при моніторингу з підключенням IP-камер.

Рис 3.7 Фрейм відображення знайдених результатів

На лістингу 3.4 наведений код побудови фрейму для відображення результатів розпізнавання.

Лістинг 3.4 Права панель для відображення результатів

```
self.frame_results = Frame(root, **self.style_frame)
self.frame_results.place(x=840, y=100, width=350, height=500)
self.best_canvas = Canvas(self.frame_results, bg="#34495E",
highlightthickness=0)
```

```
self.best_scrollbar = Scrollbar(self.frame_results, orient="vertical",
command=self.best_canvas.yview)
self.best_canvas.configure(yscrollcommand=self.best_scrollbar.set)
self.best_scroll_frame = Frame(self.best_canvas, bg="#34495E")
self.best_canvas.create_window((0, 0), window=self.best_scroll_frame,
anchor="nw")
self.best_scroll_frame.bind(
    "<Configure>",
    lambda e: self.__update_scrollregion()
)
self.best_canvas.bind_all("<MouseWheel>", self._on_mousewheel)
```

- Фрейм для кнопок керування потоками відео – він активується після завантаження відеозаписів або підключення камери, що зображено на рисунку 3.8. З’являються три кнопки: “Старт , “Пауза” і “Стоп”, вони мають різні інтуїтивно зрозумілі кольори (зелений, помаранчевий та червоний). Їх призначення - контроль процесу обробки даних. У тій самій області розміщені кнопки “Зберегти” та “Очистити”, які забезпечують збереження знайдених результатів і очищення даних.

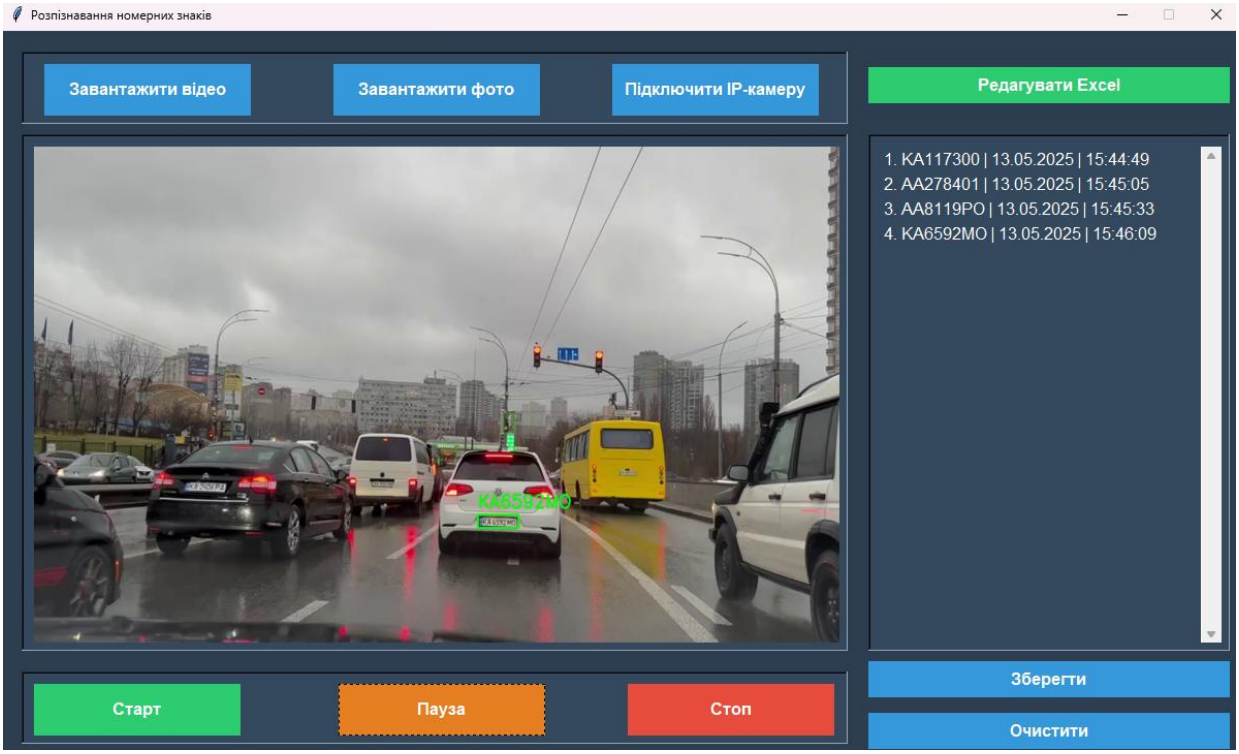

Рис 3.8 Кнопки керування відеозаписом

- Фрейм редагування збережених номерів автомобільного транспорту у форматі Excel – цей фрейм є прихованим, він активується після натискання кнопки “Редагувати Excel” і завантаження обраного файлу. Кнопка змінює колір на червоний, змінюється її назва, що підкреслює активність режиму редагування і відкривається фрейм з можливістю редагувати обрані колонки, це зображено на рисунку 3.9.

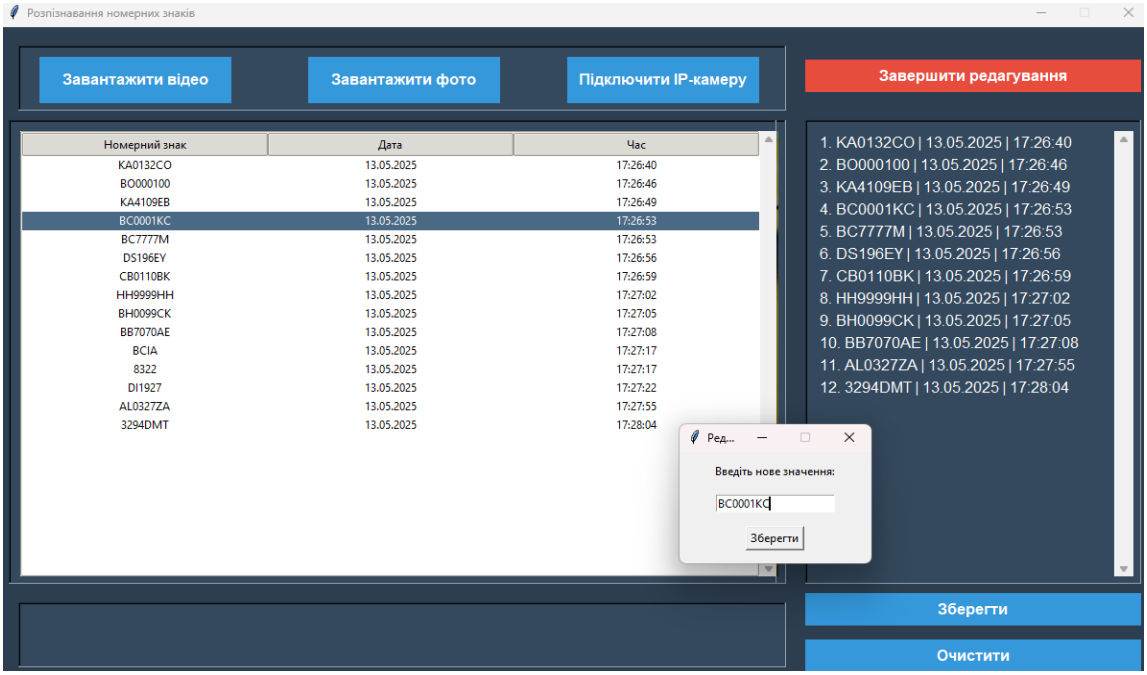

Рис 3.9 Редагування Excel

На лістингу 3.5 можна побачити використання компоненту Treeview для відображення та редагування даних із обраного користувачем Excel файлу. Використання цієї функції у системі дозволяє аналізувати розпізнані номерні знаки, редагувати їх і вносити потрібні нотатки до кожного із знайдених номерів.

Лістинг 3.5 Відображення та реагування даних Excel

```
self.frame_excel = Frame(root, bg="#34495E", bd=2, relief="sunken")
self.tree = ttk.Treeview(self.frame_excel, show="headings")
 self.tree_scroll = Scrollbar(self.frame_excel, orient="vertical",
command=self.tree.yview)
self.tree.configure(yscrollcommand=self.tree_scroll.set)
self.tree.place(x=10, y=10, width=770, height=480)
self.tree_scroll.place(x=780, y=10, width=20, height=480)
self.tree.bind("<Double-1>", self.enable_cell_editing)
self.excel_visible = False
```

Фрейм спочатку прихований і відображається лише в режимі редагування. Treeview у режимі show = “headings” відображає лише дані без зайвих колонок, потім прив'язується Scrollbar. Подія «Double-1» дозоляє редагувати обрані колонки через метод enable_cell_editing.

3.5 Налаштування відеопотоку

Для додавання універсальності системі розпізнавання номерних знаків автомобільного транспорту було розроблено підтримку локальних зображень, відеофайлів, а також підтримку потоків з IP-камер. Основним механізмом для організації читання кадрів в системі є – cv2.VideoCapture, цей клас допомагає отримувати відеопотік із завантаженого локального файлу і з IP-камери через одну й ту саму функцію бібліотеки OpenCV, що дуже спрощує роботу та підтримку системи. Лістинг 3.6 показує реалізацію локального завантаження відеофайлу та можливість підключення до IP-камери за допомогою cv2.VideoCapture

Лістинг 3.6 Ініціалізація відеофайлу і IP-камери для подальшої обробки

```
# Завантаження відеофайлу
def upload_video(self):
    self.current_media = "video"
    self.detected_plates.clear()
    self.video_path = filedialog.askopenfilename(filetypes=[("Video Files", "*.mp4;*.avi;*.mov")])
    if self.video_path:
        self.cap = cv2.VideoCapture(self.video_path)
        ret, frame = self.cap.read()
        if ret:
            self.display_frame(frame)

# Підключення IP-камери
def connect_ip_camera(self):
    self.current_media = "camera"
    self.detected_plates.clear()
    camera_popup = tk.Toplevel(self.root)
```

```
    tk.Label(camera_popup, text="Введіть URL:").pack()
    entry_url = tk.Entry(camera_popup)
    entry_url.insert(0, "RTSP/HTTP")
    def try_connect():
        self.cap = cv2.VideoCapture(entry_url.get())
        if self.cap.isOpened():
            ret, frame = self.cap.read()
            if ret:
                self.display_frame(frame)
                camera_popup.destroy()
    tk.Button(camera_popup, text="Підключити", command=try_connect).pack()
 def display_frame(self, frame):
    frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    img_resized = Image.fromarray(frame_rgb).resize((780, 480))
    img_tk = ImageTk.PhotoImage(img_resized)
    self.video_label.config(image=img_tk)
    self.video_label.image = img_tk
```

За допомогою метода upload_video локальний файл обробляється і робить ініціалізацію в cv2.VideoCapture для наступної обробки. Метод connect_ip_camera створює вікно для введення URL потоку (RTSP/HTTP) і також використовує cv2.VideoCapture для підключення. Обидва методи викликають display_frame, який конвертує кадр у RGB, змінює його розмір до розмірів фрейма де відображаються відеозаписи (video_label). Такий підхід допомагає урегульовувати відображення кадрів незалежно від його джерела.

Щоб отримувати відеопотік у реальному часі, потрібно приєднатись до IP-камери за допомогою протоколу RTSP. У моєму проекті реалізований метод “connect_ip_camera”, що показаний на лістингу 3.6. Після натискання кнопки «Підключити IP-камеру» відкривається спливаюче вікно, де користувач повинен ввести URL потоку, це зображено на рисунку 3.10.

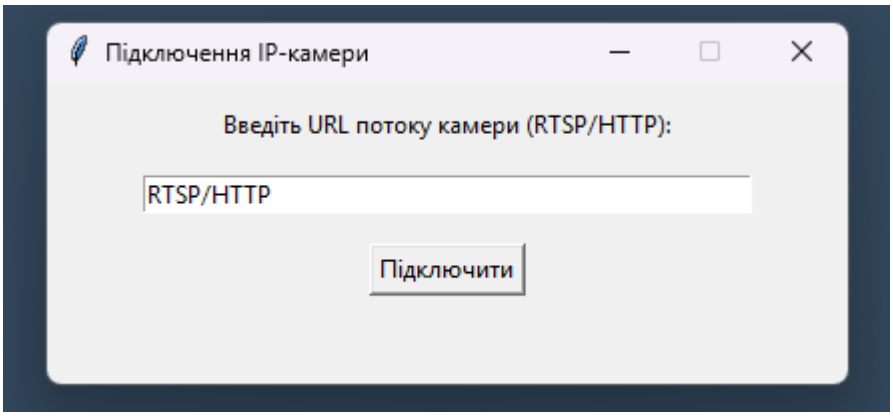

Рис 3.10 Вікно підключення камери

Для коректної роботи системи з підключеною IP-камерою, користувач повинен знати IP-адресу камери, її логін і пароль. Камера повинна бути підключена до однієї мережі бути з пристроєм на якому встановлене програмне забезпечення. Після врахування цих моментів і успішного підключення, потік з відеокамери буде оброблятися в реальному часі за допомогою метода “update_video”, цей метод зчитує кадри кожні 30 мс, кадри аналізуються і передаються далі, вже для розпізнавання номерних знаків автомобільного транспорту. Також після підключення відеокамери з’являться кнопки керування, які до цього моменту були приховані.

3.6 Сумісне використання моделі YOLO та PaddleOCR

Сумісне використання набору цих технологій у системі розпізнавання номерних знаків автомобільного транспорту грає ключову роль, завдяки ним було досягнуто високої точності і швидкості розпізнавання у реальному часі. Такий комбінований підхід розкриває всі переваги обох технологій. Модель YOLOv11 яка була використана у розробці програмного забезпечення, аналізує вхідні кадри, це можуть бути звичайні зображення або відеопотоки і з великою швидкістю визначає координати рамки номера. Після визначення рамок, вони вирізаються і передаються до PaddleOCR для оптичного розпізнавання символів. На лістингу 3.7 показана ініціалізація моделей YOLO та PaddleOCR у проекті.

Лістинг 3.7 Ініціалізація YOLO і PaddleOCR

```
self.yolo_model = YOLO("train/weights/best.pt")
self.ocr = PaddleOCR(lang='en')
self.cap = None
self.video_path = None
self.running = False
self.detected_plates = defaultdict(int)
```

Після ініціалізації моделей ми переходимо до етапу обробки вхідного зображення чи відео файлу за допомогою моделі YOLOv11. Отримані координати рамки номерного знаку автомобільного транспорту повертаються у форматі (x1, y1, x2, y2), там де індекс один – визначається верхній лівий кут рамки, там де індекс два – нижній правий. Но основі цього за допомоги NumPy вирізається потрібна нам область. Така реалізація точно вирізає область номерного знаку з кадру, незважаючи на його положення і розмір. Нижче наведений лістинг 3.8, він демонструє використання виділення області і передавання на розпізнавання символів OCR.

Лістинг 3.8 Вирізання області номерного знаку і передавання на обробку в PaddleOCR

```
def detect_plate(self, image):
    results = self.yolo_model(image, conf=0.5)
    detected_texts = []
    for result in results:
        for box in result.boxes.data:
            x1, y1, x2, y2 = map(int, box[:4])
            cv2.rectangle(image, (x1, y1), (x2, y2), (0, 255, 0), 2)
            plate_roi = image[y1:y2, x1:x2]
            plate_roi_gray = cv2.cvtColor(plate_roi, cv2.COLOR_BGR2GRAY)
            plate_roi_rgb = cv2.cvtColor(plate_roi_gray, cv2.COLOR_GRAY2BGR)
            ocr_results = self.ocr.ocr(plate_roi_rgb, cls=True)
```

На наведеному коді видно як знайдена область готується до передачі в OCR, вона переводиться у градації сірого, щоб контрастність була кращою, а потім повертає формат BGR, який необхідний для роботи PaddleOCR.

Після успішної передачі вирізаної області до OCR, вона починає виявляти символи на завантажених рамках номерних знаків але поки не показує результату. Текст номерного знаку починає проходити фільтрацію і перевірку на довжину номерного знаку (4-10 допустима кількість символів) і фільтрує зайві символи, це відбувається за допомогою бібліотеки регулярних виразів. Також система веде підрахунок кількості розпізнавань для кожного номерного знаку, що дозволяє знизити ризик повторних ідентифікацій одного й того самого номера автомобіля і хибних спрацювань. Код наведений на лістингу 3.9 демонструє процес фільтрації і рахування кількості розпізнавань кожного номерного знаку(detected_plates).

Лістинг 3.9 Фільтрація виявлених символів і підрахунок кількості розпізнавань

```
 for line in ocr_results:
    if line:
        for res in line:
            if res and len(res) > 1 and res[1]:
                text = re.sub(r'[^A-Z0-9]', '', res[1][0])
                if 4 <= len(text) <= 10:
                    detected_texts.append(text)
                    if self.detected_plates[text] < self.MIN_DETECTIONS:
                        self.detected_plates[text] += 1
                    cv2.putText(
                        image, text, (x1, y1 - 10),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 255, 0), 2
                    )
return image, detected_texts
```

Таким чином, система обробки тексту після роботи OCR дозволяє підвищити точність розпізнавання і зробити кількість помилок мінімальним за рахунок багатоетапної перевірки кожного розпізнаного номерного знаку.

В залежності від вибору користувача (завантажити зображення, відео чи підключити IP-камеру) змінюється мінімальна кількість детекцій, впровадження такого рішення ґрунтується на особливостях обробки різних типів вхідних даних. Для статичного зображення достатньо тільки одного зчитування номера. А коли йде потік кадрів з відео чи IP-камери треба більше зчитувань. В кадрах постійно відбувається рух, велика швидкість автомобільного транспорту, об’єкти можуть перекриватись, це все впливає на розпізнавання, тому було вирішено додати багаторазове підтвердження одного й того самого номерного знаку на відео, що збільшить стабільність розпізнавання в динамічних умовах і зменшить хибні результати. На лістингу 3.10 показана динамічна зміна порогу детекцій для відео.

Лістинг 3.10 Зміна порогу детекцій

```
def upload_video(self):
    self.MIN_DETECTIONS = 5
    self.current_media = "video"
    self.detected_plates.clear()
```

Атрибут «self.MIN_DETECTIONS» визначає, скільки разів номерний знак маю бути виявлений, щоб вважатися достовірним і відобразитись в графічному інтерфейсі.

3.7 Тестування системи розпізнавання номерних знаків автомобільного транспорту

Система розпізнавання номерних знаків була протестована в різних умовах для забезпечення впевненості у роботі всіх процесів і точності розпізнавання. Для цього було проведено кілька етапів аналізу – статистичні зображення, обробка відеозаписів знятих у поганих умовах.

Першим етапом було перевірено розпізнавання на локально завантажених зображеннях. Вони мають різну якість і номери автомобільного транспорту розташовані під різними кутами, результати тестування статичних зображень можна побачити на рисунках 3.11 і 3.12.

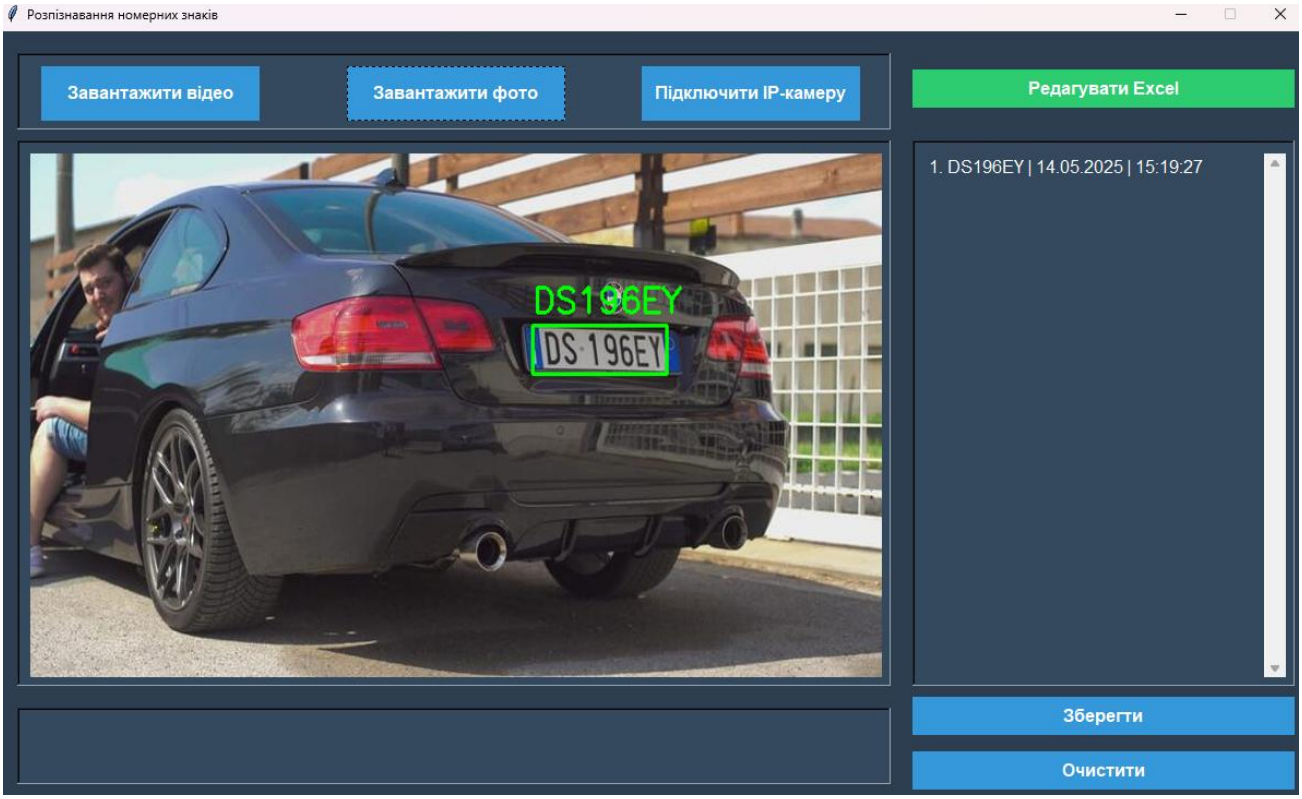

Рис 3.11 Завантаження зображення високої якості

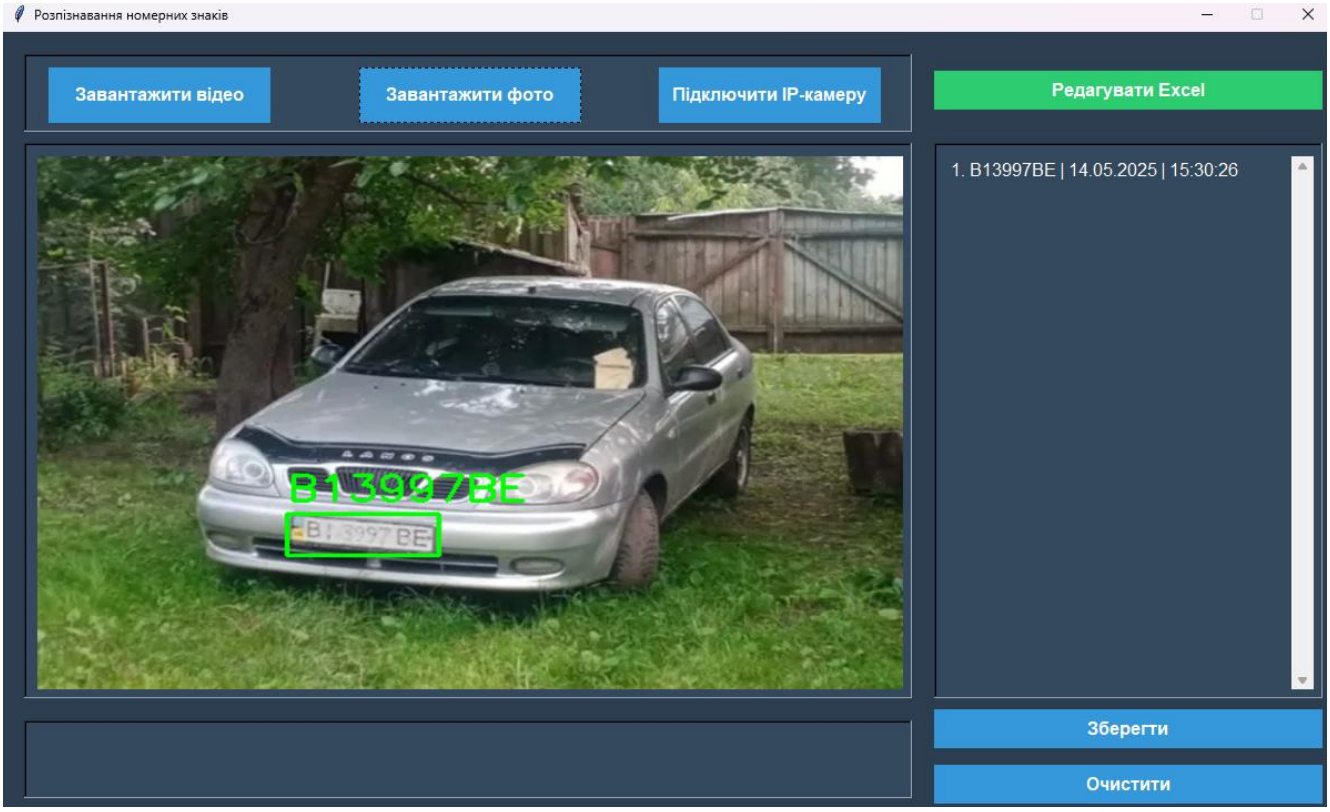

Рис 3.12 Завантаження зображення поганої якості зі стертим номерним знаком

Другий етап тестування проводився на відеофайлах, цей процес покаже як обробляються локальні відеозаписи, що дасть нам змогу зрозуміти як оброблятимуться IP-камери. Для тестування було обрано відеозапис у форматі MP4 із частотою 30 кадрів/с. Відео було записане з вікна автомобільного транспорту, що рухався з великою швидкістю, за умов поганої видимості через дощ, що допоможе оцінити точність розпізнавання в складних умовах. На рисунку 3.13 зображені результати розпізнавання номерних знаків з відеозапису.

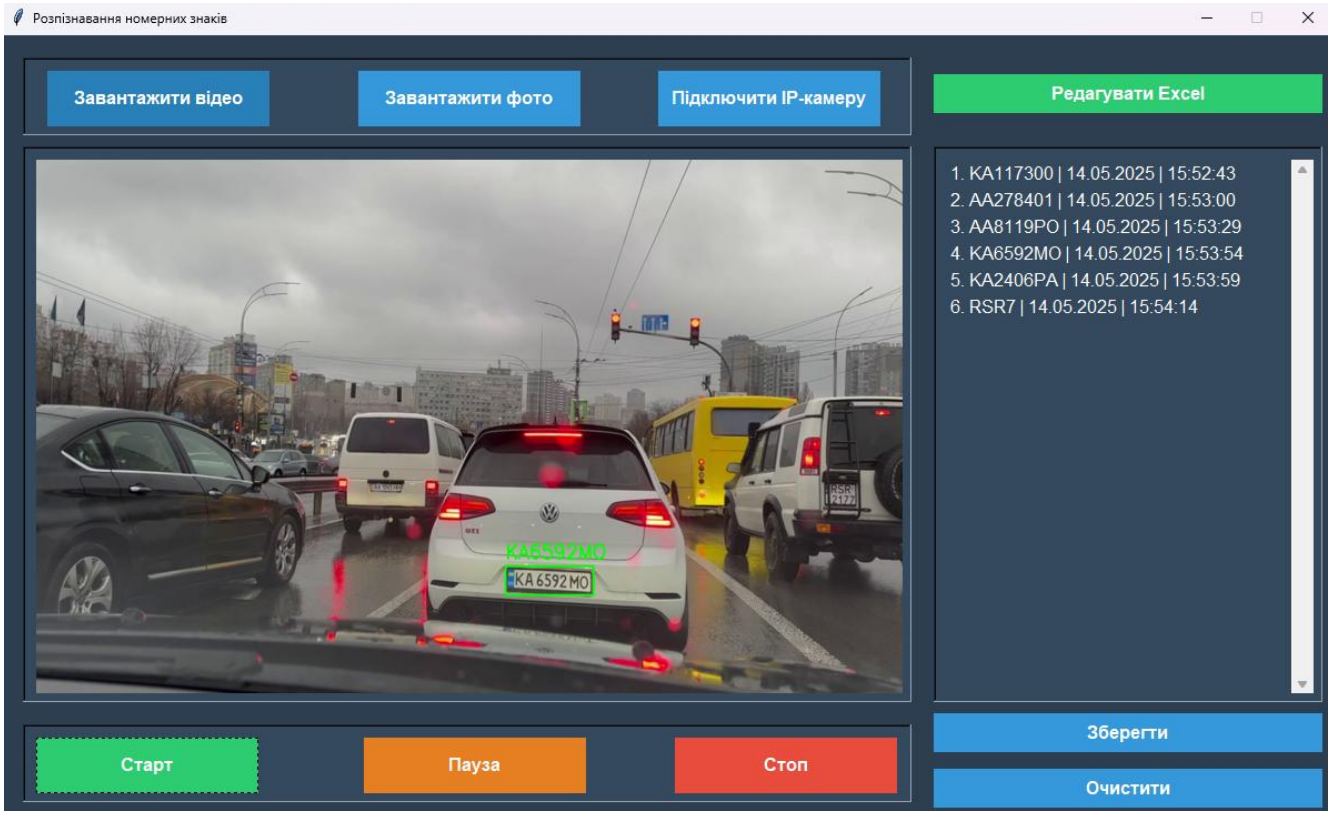

Рис 3.13 Завантаження відеозапису зі складними умовами

Система успішно виявила всі номерні знаки на зображеннях різної якості, а також на відеозапису, незважаючи на складні умови. На відеозаписі, знятому за умов дощу та розмиття через високу швидкість, точність розпізнавання склала 85%, що свідчить про високу точність і швидкість обробки кадрів. Проведені тестування демонструють високу надійність системи розпізнавання номерних знаків автомобільного транспорту, що дозволяє з впевненістю використовувати її на підприємствах в різних умовах експлуатації.

Висновки до розділу 3

У третьому розділі показана практична реалізація ключових програмних компонентів і алгоритмів для розпізнавання знаків автомобільного транспорту. Було описано використану мову програмування, середовища для розробки Google Colaboratory і Pycharm, а також бібліотеки які використовувались при написанні проекту: OpenCV, YOLOv11, PaddleOCR, Tkinter.

Модель штучного інтелекту YOLOv11 була навчена на наборі даних який містить 10 000 зображень, проаналізовано її точність за допомогою метрик. Сумісне використання моделі YOLO та PaddleOCR дозволило досягти високої точності та швидкості системи. Створено графічний інтерфейс, який забезпечує зручну взаємодію користувача і підтримку роботи не тільки з локальними файлами, а і з IP-камерами.

Тестування розробленої системи розпізнавання номерних знаків автомобільного транспорту показало хорошу точність враховуючи різні сценарії роботи, що підтверджує надійність розробленого програмного забезпечення.

ВИСНОВКИ

Основною метою мого дослідження було розробити систему розпізнавання номерних знаків автомобільного транспорту для автоматизації моніторингу на підприємствах. У ході виконання дослідження розглянуто теоретичні, аналітичні та практичні аспекти, що дозволило обрати правильні технології та алгоритми для реалізації функціонального програмного забезпечення для розпізнавання номерних знаків. Впровадження такої системи на підприємство значно полегшить рутинні завдання співробітникам та забезпечить більш надійний та автоматизований контроль на контрольно-пропускних пунктах.

Було проаналізовано основи і теоретичні аспекти комп'ютерного зору. Історичний огляд показав зародження технологій виявлення об'єктів і великі прориви у глибокому навчанні. Також було розглянуто практичні приклади моніторингу трафіку, контролю доступу, вони показують важливість розробки і вдосконалення систем автоматизованого контролю номерних знаків. Огляд програмних рішень виявив їхні слабкі та сильні сторони, після аналізу цих проблем було розроблено кроки їх вирішення, що допомогло мені при створені свого програмного забезпечення.

Проаналізовано моделі глибокого навчання та оптичного розпізнавання символів – це основні компоненти сучасних систем моніторингу. Модель YOLOv11 була обрана завдяки її швидкості і детального аналізу порівняння з іншими моделями штучного інтелекту. PaddleOCR також показав більш надійний і широкий вибір функцій ніж його аналоги.

Розроблено модель системи розпізнавання номерних знаків транспортних засобів на підприємствах, яка функціонує завдяки правильно налаштованим алгоритмам і поетапного налаштування.

В результаті проведеного дослідження було створено систему розпізнавання номерних знаків автомобільного транспорту в реальному часі. Проведене тестування системи, показало що вона може бути використана на підприємствах, контрольно-пропускних пунктах та забезпечить надійність і точність.

ПЕРЕЛІК ПОСИЛАНЬ

1. Image recognition from the early days of technology to endless business applications today. Trendskout. URL: https://trendskout.com/en/solutions/image-recognition-technology/
2. Roberts, L. G. *Machine Perception of Three-Dimensional Solids*. Doctoral dissertation, Massachusetts Institute of Technology. Cambridge, 1963. С. 15-30.
3. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. ImageNet: a large-scale hierarchical image database. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009. С. 1-8.
4. Fourier Transform. MathWorld. URL: https://mathworld.wolfram.com/FourierTransform.html .
5. Difference Equation. DSP RELATED. URL: https://www.dsprelated.com/freebooks/filters/Difference_Equation_I.html.
6. 2D Convultion in Image Processing. All About Cirtuits. URL: https://www.allaboutcircuits.com/technical-articles/two-dimensional-convolution-in-image-processing/ .
7. Otsu, N. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 1979. № 1. С.62-66.
8. Flasinski, P. Syntactic pattern recognition of ECG for diagnostic justification. *Machine Graphics and Vision*, 2014. №3/4. С. 43-55. URL: https://mgv.sggw.edu.pl/article/view/2989 .
9. ALPR-Unconstrained. GitHub. URL: https://github.com/sergiomsilva/alpr-unconstrained .
10. Rekor Systems Selected by Department of Defense to Provide Automatic License Plate Recognition Software. Rekor Systems. URL: https://www.rekor.ai/post/rekor-systems-selected-by-department-of-defense-to-provide-automatic-license-plate-recognition-software .

11. AXIS License Plate Verifier. Axis Communications. URL: https://www.axis.com/products/axis-license-plate-verifier.
12. Розпізнавання автомобільних номерів. Kobi Software. URL: https://kobi.ua/technological-solutions/plate-recognition/.
13. Redmon, J.,Divvala, S., Girshick, R., Farhandi, A. You Only Look Once: Unified, Real-Time Object Detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. С.779-788.
14. Ultralytics YOLO11. Ultralytics Docs. URL: https://docs.ultralytics.com/ru/models/yolo11/ .
15. YOLO: Algorithm for Object Detection Explained. V7labs. URL: https://www.v7labs.com/blog/yolo-object-detection.
16. How to Train YOLOv8 Object Detection on a Custom Dataset. Roboflow. URL: https://blog.roboflow.com/how-to-train-yolov8-on-a-custom-dataset/.
17. CAR_DATASET Computer Vision Project. Roboflow. URL: https://universe.roboflow.com/team59car/car_dataset-ddzft.
18. OpenCV: Automatic License/Number Plate Recognition (ANPR) with Python. Pyimagesearch. URL: https://pyimagesearch.com/2020/09/21/opencv-automatic-license-number-plate-recognition-anpr-with-python/.
19. Multi-langue model. PaddleOCR Documentation. URL: https://paddlepaddle.github.io/PaddleOCR/main/en/ppocr/blog/multi_languages.html.
20. Автоматичне розпізнавання автономерів з відеоаналітики. Verna. URL: https://www.verna.ua/videoanalytics/license-plate-recognition.
21. Nohai, M., & Dzhaka, R. Preprocessing of images for Optical Character Recognition Using Neural Networks. *Journal of Pattern Recognition Research*, 2011. С.1-11.

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Програмний додаток для розпізнавання номерних знаків за допомогою методів комп'ютерного зору»

Виконав: здобувач вищої освіти гр. ШІД-42
Валентин ЗАЦЕРКОВНИЙ
Керівник: кандидат технічних наук, доцент
Олександр ЗВЕНІГОРОДСЬКИЙ

2025р.

МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- **Мета роботи:** збільшення ефективності автоматизованого виявлення і розпізнавання номерних знаків транспортних засобів для покращення процесів моніторингу потоку машин на контрольно-пропускних пунктах, та автоматизації контролю за транспортом на підприємствах.
- **Об'єкт дослідження:** процес розпізнавання образів і машинного навчання в галузі комп'ютерного зору.
- **Предмет дослідження:** алгоритми розпізнавання об'єктів на фото і у відео послідовностях, алгоритми оптичного розпізнавання символів.

Завдання дослідження

1. Зробити літературний огляд проблеми розпізнавання номерних знаків автомобільного транспорту.
2. Проаналізувати сучасні технології глибокого навчання і оптичного розпізнавання символів.
3. Розробити формальну модель системи моніторингу транспортних засобів.
4. Вибрати засоби програмування та створити програмний продукт.
5. Провести тестування системи, зробити аналіз точності розпізнавання номерних знаків.

Літературний огляд проблеми розпізнавання номерних знаків

Проблеми розпізнавання номерних знаків автомобілів:

- Низька роздільна здатність зображень: розмиті контури та символи ускладнюють ідентифікацію.
- Велика швидкість автомобілів: рух спричиняє розмиття, що знижує точність розпізнавання.
- Неправильне розташування камер: невідповідний кут або висота ускладнює детекцію.
- Різноманітність шрифтів і символів: схожість символів може призводити до помилок.
- Погодні умови та час доби: дощ, туман, тіні знижують якість зображень.
- Перешкоди: дерева, рекламні щити можуть закривати номерні знаки.
- Різні формати номерів: відмінності в кольорах, розмірах і структурі між країнами.

Варіанти вирішення:

- Використання камер високої роздільної здатності (від 1080p).
- Алгоритми компенсації руху та адаптивні налаштування для погодних умов.
- Донавчання моделей на різноманітних датасетах для підтримки різних форматів.

Існуючі програмні рішення

➢ **ALPR-Unconstrained**
• Відкрите ПЗ з використанням глибокого навчання
• Висока точність
• Відкрите для модифікацій
- Обмежена підтримка, складність розгортання

➢ **OpenALPR**
• Безкоштовна та комерційна версія
• Підтримка понад 70 країн
• Інтеграція через API
- Вимогливе до обладнання

➢ **Axis Communications ANPR**
• Інтеграція апаратного і програмного забезпечення
• Працює в реальному часі
- Висока ціна, складність адаптації під інші камери

➢ **Kobi Software (Україна)**
• Інтеграція з «Безпечним містом»
• Підтримка IP-камер
• Висока точність
- Дорогі пристрої, обмежена документація

ВИКОРИСТАНІ ТЕХНОЛОГІЇ ТА ІНСТРУМЕНТИ

Середовища розробки:

Google Colab – тренування моделей у хмарному середовищі з використанням GPU

PyCharm - основне середовище для написання коду

Використані бібліотеки:

- ✓ Roboflow
- ✓ Ultralytics
- ✓ OpenCV
- ✓ YOLOv11
- ✓ PaddleOCR
- ✓ Pandas
- ✓ Tkinter
- ✓ Re
- ✓ Pillow
- ✓ DateTime

МОДЕЛЬ СИСТЕМИ МОНІТОРИНГУ НОМЕРНИХ ЗНАКІВ ТРАНСПОРТНИХ ЗАСОБІВ

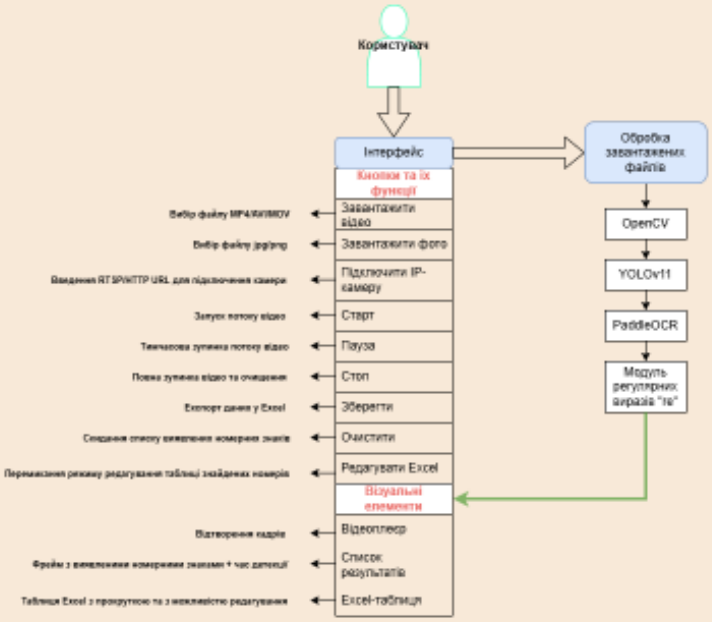

Рис. 1. Схема роботи системи моніторингу

Демонстрація програмного забезпечення

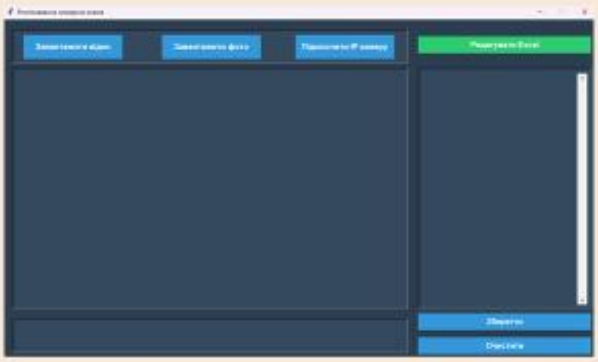

Рис. 2. Головне вікно програми

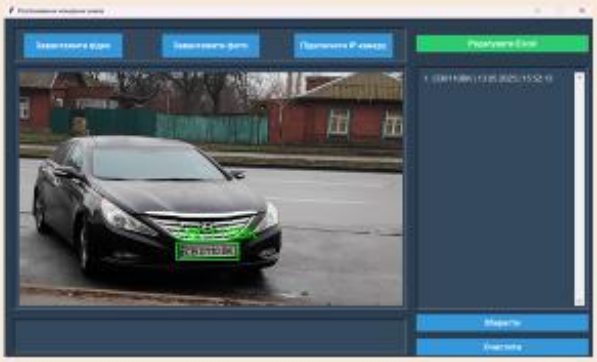

Рис. 3. Відображення завантаженого зображення

Демонстрація програмного забезпечення

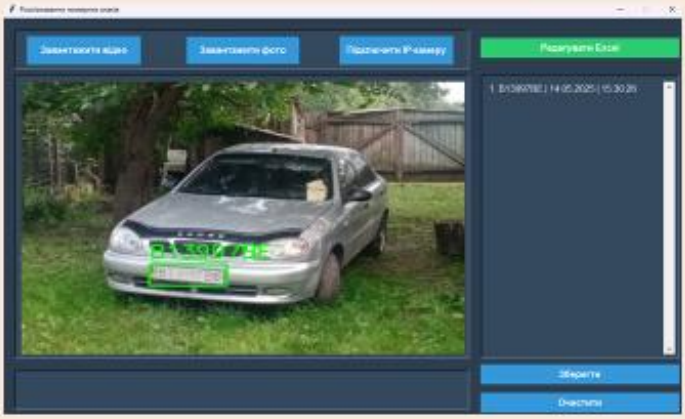

Рис. 4. Завантаження зображення поганої якості зі стертим номерним знаком

Демонстрація програмного забезпечення

Рис. 5. Завантаження відеозапису з поганими погодними умовами

Демонстрація програмного забезпечення

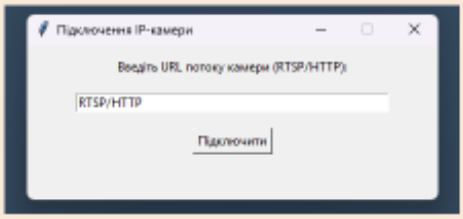

Рис. 6. Підключення IP-камери

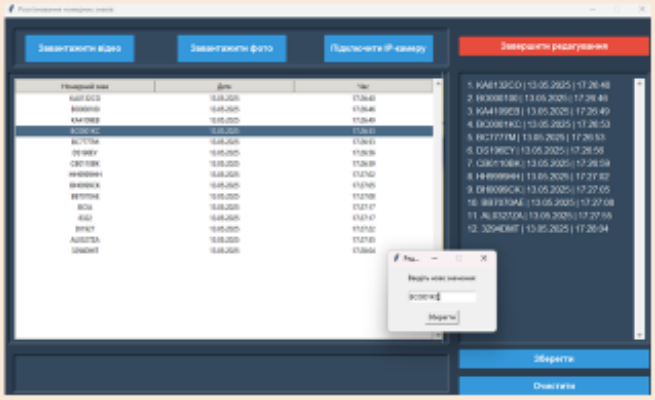

Рис. 7. Редагування знайдених номерних знаків в Excel

ВИСНОВКИ

➢ Проведений аналіз:
- Проведено літературний огляд проблеми розпізнавання номерних знаків транспортних засобів
- Виявлено недоліки існуючих рішень і шляхи їх усунення
- Оцінено існуючі технології глибокого навчання та оптичного розпізнавання символів

➢ Технологічна реалізація:
- Обрано оптимальні інструменти: YOLOv11 та PaddleOCR
- Створена модель моніторингу номерних знаків транспортних засобів
- Реалізовано підтримку завантаження локальних фото/відео, підключення IP-камери
- Створено інтерфейс для керування та збереження результатів

Розроблений програмний додаток забезпечить надійний контроль транспорту на підприємствах, покращить процеси контролю потоків машин на КПП, автоматизує рутинні дії.