МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНВЕРСИТЕТ
ІМЕНА ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра інтелектуальних технологій

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

НА ТЕМУ

Нейромережний застосунок ідентифікації номерних знаків транспортних засобів, що перебувають у розшуку

Галузь знань: **12 «Інформаційні технології»**
Спеціальність: **122 «Комп'ютерні науки»**
Освітня програма: **«Аналітика даних»**
Освітній рівень: бакалавр

Виконав студент 4-го курсу групи АнД-41

Білуха Назар Романович _____
підпис

Керівник: к.т.н., доцент, Іларіонов О.Є.

_____
Підпис

Випускна кваліфікаційна робота бакалавра допущена до захисту рішенням кафедри *інтелектуальних технологій*
Протокол № 13 від 05.06.2023 р.

Завідувач кафедри _____ доц. Іларіонов О.Є.

Київ – 2023

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій
Кафедра інтелектуальних технологій
Спеціальність 122 «Комп'ютерні науки»

ЗАТВЕРДЖУЮ
Завідувач кафедри інтелектуальних технологій
Іларіонов О.Є.

"___"_____2022 р.

ЗАВДАННЯ
НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

_____Білусі Назару Романовичу_____
(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) «Нейромережний застосунок ідентифікації номерних знаків транспортних засобів, що перебувають у розшуку», затверджена протоколом засідання кафедри від «11» листопада 2022 р. № 4
2. Термін здачі студентом закінченого проекту (роботи) _____ травня 2023 року
3. Вихідні дані до проекту (роботи) _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)

   Дослідження та аналіз предметної області, що пов'язана із методами генерацій зображень на основі текстових описів, програмна реалізація модулю, тестування розробленого застосунку_____
5. Перелік презентаційного матеріалу (з точним зазначенням обов'язкових презентацій)

   Мета дослідження дипломного проекту (1 слайд), актуальність завдання (1 слайд), контекстна діаграма програмного модулю розробки (1 слайд), опис предметної області (2 слайди), постановка задачі (1 слайд), функціональний аналіз (1 слайд), опис навчального набору даних (1 слайд), алгоритм вирішення задачі (1 слайд), схема архітектури програмного модулю та опис інструментів програмної реалізації (1 слайд), оцінка результатів та приклади роботи застосунку (2 слайди), висновки (1 слайд).

6. Консультанти з випускної кваліфікаційної роботи із зазначенням розділів випускної кваліфікаційної роботи, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 15 лютого 2023 року

Керівник _____ / Іларіонов О.Є. /
(підпис) (ПІБ)

Завдання прийняв до виконання _____ / Білуха Н.Р. /
(підпис) (ПІБ)

КАЛЕНДАРНИЙ ПЛАН

Пор. №	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів випускної кваліфікаційної роботи	Примітка
1.	Обговорення з керівником постановки завдання, підбір літератури, огляд існуючих рішень	15.02.2023 – 01.03.2023	
2.	Аналіз постановки задачі, формалізація задачі, аналіз літературних джерел, написання розділу 1	02.03.2023 – 20.03.2023	
3.	Проектно-технологічна реалізація неромережного застосунку ідентифікації номерних знаків транспортних засобів, що перебувають у розшуку.	21.03.2023 – 11.04.2023	
4.	Розробка та тестування застосунку ідентифікації номерних знаків транспортних засобів	12.04.2023 – 15.05.2023	
5.	Робота над оформленням пояснювальної записки та презентаційних матеріалів	16.05.2023 – 29.05.2023	

Студент-дипломник _____ / Білуха Н.Р. /
(підпис) (ПІБ)

Керівник випускної кваліфікаційної роботи _____ / Іларіонов О.Є. /
(підпис) (ПІБ)

Анотація

Білуха Назар Романович виконав випускну кваліфікаційну роботу на тему “Нейромережний застосунок ідентифікації номерних знаків транспортних засобів, що перебувають у розшуку” за спеціальністю 122 – “Комп’ютерні науки”, освітньою програмою “Аналітика даних”.

У випускній кваліфікаційній роботі досліджено питання ідентифікації та класифікації номерних знаків; розроблено програмний застосунок для виявлення знаків та перевірки їх до належності правопорушень. Методи, що розглядались для реалізації: згорткові нейронні мережі, алгоритм Канні, оператор Собеля, градієнт.

Ключові слова: згорткова нейронна мережа, нейромережа, класифікація, ідентифікація, номерний знак, транспортний засіб.

Abstract

The bachelor’s project “Neural network application for identifying license plates of stolen vehicles” has been completed by Nazar Bilukha specialty 122 – “Computer science”, educational program “Data analysis”.

The issue of identification and classification of license plates was investigated; a software application was developed to detect license plates and check them for offenses. Methods considered for implementation: convolutional neural networks, Canny algorithm, Sobel operator, gradient.

Keywords: convolutional neural network, neural network, classification, identification, number plate, vehicle.

Зміст

Умовні позначення та скорочення

КЗ – Комп’ютерний зір

ДТП – дорожньо транспортна пригода

ТЗ – транспортний засіб

VIN (Vehicle Identification Number) – унікальний номер автомобіля

CV – computer vision

API – application programming interface

AWS – amazon web service

Int, Float, String, Date – типи даних в реляційних базах даних

UML – unified modeling language

ВСТУП

Розпізнавання номерного знаку – один із найпростіших способів ідентифікації автомобіля в повсякденному житті. Сучасна міжнародна система формування номерів дозволяє класифікувати транспортний засіб за країною, регіоном, типом двигуна та низкою інших факторів відповідно до законів країни реєстрації.

Щодня десятки тисяч автомобілів проходять контрольно пропускні пункти, перевірка на яких займає до декількох хвилин, що призводить до затримки дорожнього руху. З цієї причини можуть бути прорахунки з логістикою продовольчої продукції, ліків, критично важливої сировини як для великих компаній так і для середнього та малого бізнесу.

Першочергове завдання ідентифікації транспортних засобів полягає в забезпеченні перевірки автомобіля в базі даних, отриманні усієї необхідної інформації в найкоротші терміни. Це дозволить автоматизувати роботу установ, які відповідають за безпеку пропускних пунктів, покращити швидкість уточнення даних по транспорту та власнику. Розроблена система дозволяє уникати помилок внаслідок “людського фактору” та запобіганню шахрайства зі сторони державних органів.

Основною метою створення програмного модулю є розпізнавання автомобільного знаку за допомоги методів машинного навчання та технологій комп’ютерного зору.

В ході створення модулю буде проведено ряд досліджень над даними, проаналізовано різні підходи до вирішення поставленої задачі. Об’єктом дослідження є діяльність правоохоронних органів, предметом досліджень є модель нейронної мережі, застосована для дослідження об’єкту.

Результат даної роботи буде корисний для державних установ, підприємств, яким важливо контролювати потік транспортних засобів на визначених ділянках дороги.

РОЗДІЛ 1. АНАЛІТИЧНИЙ ОГЛЯД МЕТОДІВ ІДЕНТИФІКАЦІЇ НОМЕРНИХ ЗНАКІВ ТРАНСПОРТНИХ ЗАСОБІВ

1.1. Система формування та поняття номерного знаку

Державний номерний знак – це унікальний ідентифікатор автомобіля. За значеннями, вказаними на номерному знакові можливо дізнатися інформацію про авто, його власника та історію експлуатації транспортного засобу. Комбінація із літер та цифр є унікальною та мають прив'язку до регіонів України.

Рисунок 1.1. Шаблон державного номеру автомобільного транспорту

З 2004 року введено формат знаків, який дійсний і сьогодні. Літери на знаках українською мовою, але вони обов'язково мають збігатися із латиницею (приклад: А, І, К – можливо, Ф, Я, Л – не можливо). Регіони України розділено в алфавітному порядку, починаючи з Києва – АА, далі Вінницька область – АВ, Волинська – АС і так далі.

З 2013 року ухвалено закон про зміни до Державних Стандартів України та замінили перші літери регіональних кодів на К, Н та І. Перші дві літери – коди регіонів, комбінація із чотирьох цифр – порядковий номер, останні дві літери – серія номеру, вона також формується із літер, які мають збігатися із латиницею.

З 2020 року, із початком реєстрації електромобілів, з'явились нові державні номери, які мають латинські літери Z та Y у правій частині комбінацій знаків (наприклад.: ZA, ZB, YA, YB тощо).

Рисунок 1.2. Карта регіонів місця реєстрації автомобільних транспортів

Ідентифікація номерних знаків – важливий процес перевірки автомобіля на можливу причетність до правопорушень або інших дій, передбачених чинним законодавством України.

На сьогодні, існують програми, які використовуються правоохоронними органами для надання інформації по автомобілю. Модуль також може використовуватися сайтами з продажу автомобілів, торговими брокерами, сервісами прокату авто та простими громадянами, яким необхідні дані про транспортних засіб.

1.2. Способи та методи ідентифікації транспортного засобу

Стрімкий розвиток автобудування протягом останнього століття зумовлює потребу в створенні системи класифікації автомобілів. Виробництво здійснюється під різними марками, моделями, серіями, проте зовнішні характеристики можуть бути ідентичними. Для того, щоб вирішити цю проблему було введено VIN код (унікальний номер транспортного засобу). Даний номер зберігає повну інформацію по експлуатації авто, причетність до ДТП та історію власників. Окрім VIN коду в автомобіля є також номер шасі, який містить інформацію про виробництво та характеристики ТЗ. Додатково

власники автівок змушені оформляти страховий поліс, техпаспорт, відповідно до чинного законодавства України. Володіючи інформацією хоча б про один із вище вказаних документів, можливо отримати ключові характеристики про автомобіль.

На сьогодні існує чимало сервісів, які допомагають в перевірці ТЗ. Проте більшість з них мають власну базу даних, яка не завжди відповідає тій, що міститься в офіційних реєстрах. Більше того, чимало сервісів здатні створювати фішингові посилання, збирати особисту інформацію про користувачів та вчиняти протиправні дії. Програмний модуль не передбачає авторизації користувачів, а лише повертає запит із офіційних статистичних джерел. Реалізація системи виконується шляхом розпізнавання об'єктів на фотографії та має назву комп'ютерний зір (Computer Vision).

Комп'ютерний зір (КЗ) – це одна із підгалузей штучного інтелекту, яка передбачає створення цифрових систем, які здатні обробляти, аналізувати, розпізнавати зображення та відеоматеріали. В основі КЗ лежать алгоритми машинного навчання. Саме вони дозволяють знаходити закономірності між даними, відрізняти одні об'єкти від інших, визначати ключові паттерни та залежності.

Найбільш поширені задачі КЗ:

- Класифікація об'єктів – система аналізує візуальний контент та класифікує предмет на фото або відео до попередньо визначеної категорії.
- Ідентифікація об'єктів – система аналізує візуальний контент та визначає предмет на фото або відео.
- Відстежування об'єктів – система ідентифікує об'єкт на відео, який відповідає певним критеріям та відстежує його рух.

Програмний модуль передбачає поєднання задачі ідентифікації та класифікації, що дозволяє більш детально дослідити принципи роботи КЗ.

1.3. Згорткові нейронні мережі

Одними із методі КЗ є згорткові нейронні мережі. Це один із типів нейронних мереж, тривіальні алгоритми в галузі глибинного навчання.

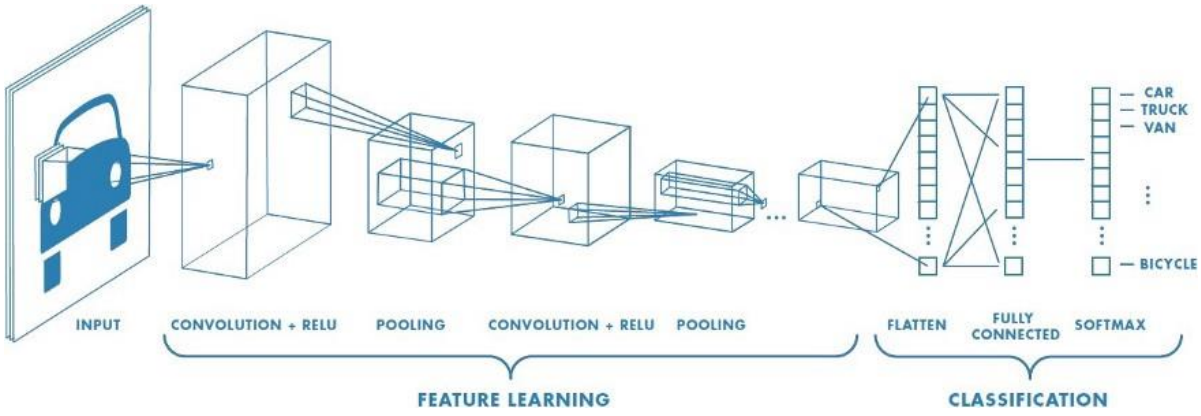

Рисунок 1.3. Принцип роботи класичної згорткової нейромережі

На вхід нейронна мережа отримує шаблони фотографій, які необхідно класифікувати. Кожне фото представлене тривимірним масивом, де кожен піксель представлений у форматі RGB, тобто масивом з трьох значень та розміщенням відносно осі X та Y.

Початковий шар згортки – ядро. Основне завдання початкового шару – проаналізувати усі можливі характеристики зображення (краї, колір, орієнтація градієнта, тощо) з розумінням подальших шаблонів.

Згорткові нейромережі в процесі обчислення передбачають зменшення просторового розміру об’єкта. Це виконується для обмеження обчислювальної потужності, необхідної для обробки даних. Крім того, дана практика дозволяє виділяти ключові домінуючі характеристики, тим самим підтримуючи ефективність процесу навчання моделі.

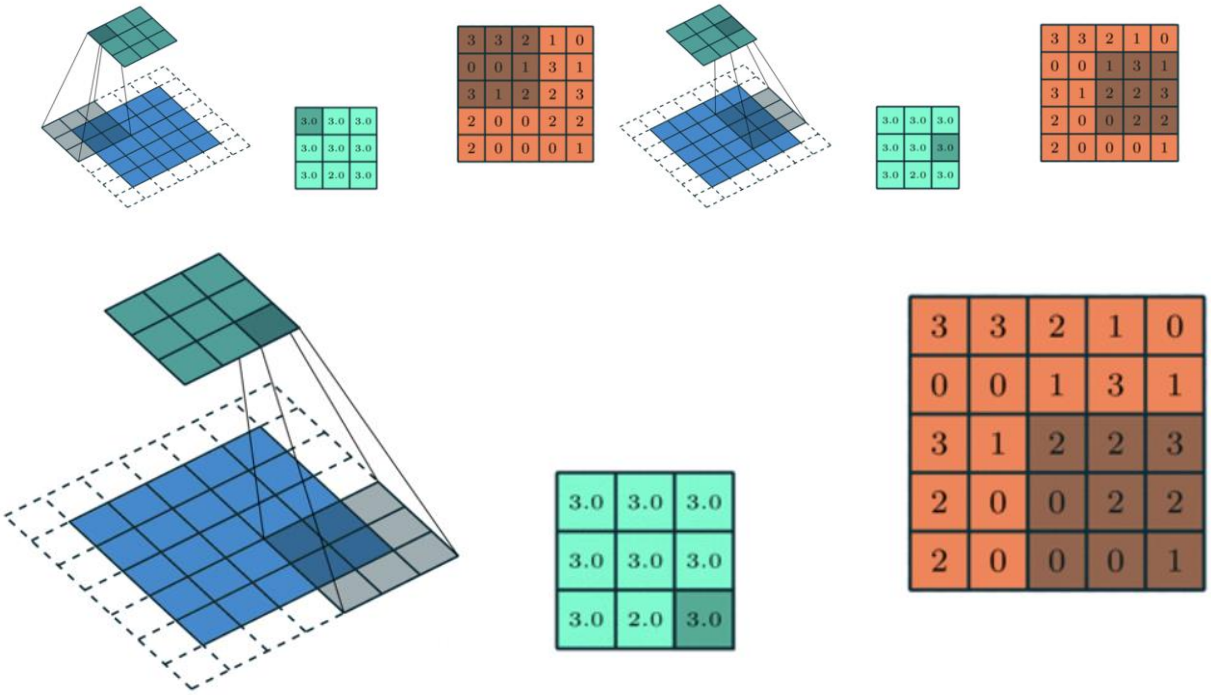

Рисунок 1.4. Технічний приклад роботи згортки

Зменшення розмірності відбувається шляхом множення матриць ключових інваріант з певним кроком. Для виділення основних частин зображення та перетворення його в подальший шар використовують операцію об’єднання – процес поділу інваріант на частини за ключовими показниками (об’єднання за середніми значень, об’єднання за максимальними значеннями).

1.4. Передумови для створення системи

Згідно з офіційними даними Національної поліції України, у 2021 році викрали 3546, за неофіційними даними – 7092 автомобілів. На жаль, викрадення автомобілів є досить поширеним явищем. В той час, як показник відсотку розкриття таких злочинів та повернення викраденого майна законним власникам – низький. Це змушує автомобілістів ретельніше обирати системи захисту для свого транспорту. Сучасні технології дозволяють зловмисникам використовувати нові способи крадіжок, за допомоги спеціальних пристроїв. Найвідоміші сучасні способи викрадення автомобіля:

- Використання кодграббера – пристрій, який використовується грабіжниками для перехоплення радіосигналу сигналізації та відтворює його. Такий спосіб є досить зручним, оскільки дозволяє заволодіти автомобілем без зайвого шуму та додаткових клопотів. Проте даний спосіб не є ефективним для нових автомобілів з покращеною системою захисту.
- Використання підробки ключа SmarkKey – щоб виркасти автомобілі з системою SmartKey (безключовий доступ) необхідно використовувати подовжувач штатного ключа. Коли власник знаходиться поза авто, пристрій підсилює сигнал від картки-ключа та відмикає центральний замок транспортного засобу. Використовується лише для автомобілів з безключовим доступом.
- Використання пристрою приглушення сигналу – для злому таким способом використовується спеціальний девайс, який приглушує сигнал автосигналізації. Власник транспортного закриває автівку на ключ, тим самим залишаючи автомобіль без цифрового захисту.
- Викрадення звичним способом – викрадення автомобіля шляхом застосування грубої сили, виманювання водія з-за керма, викрадення при продажу, фізичного пошкодження дверей автомобіля, тощо.

Досить часто процес розкривання подібних злочинів затягується на тривалий час, а в подеяких випадках взагалі відкладається. Це зумовлено складністю виявляти та відстежувати автомобіль в межах цілого міста. Розроблена система ідентифікує автівку з будь-яких цифрових об’єктів, таких як камери відеоспостереження, фотофіксація, GPS-навігатори, камера заднього/переднього виду в автомобілі. Це допомагає значно швидше виявляти порушників та отримувати інформацію про викрадений автомобіль по факту виявлення транспорту.

1.5. Вимоги до інтелектуального програмного модулю

Для якісної розробки програмного модулю необхідно сформулювати ряд функціональних та нефункціональних вимог, визначити ключові цілі системи розпізнавання номерних знаків.

1.5.1. Опис ключових компонент дослідження

Міністерство цифрової трансформації розробило єдиний портал відкритих даних – центр компетенцій, що має на меті підвищити рівень знань про відкриті дані, їхній вплив та користь для кожного. Держава збирає та консолідує інформацію про різні сфери повсякденного життя. Більшість з них – доступна та відкрита для бізнесу, проте портал містить також обмежені набори даних, які використовуються державними органами України.

Об’єктом дослідження є розроблена нейронна мережа розпізнавання номерних знаків. Уся зібрана інформація записується в єдиний офіційний реєстр, за яким здійснюється перевірка транспортного засобу. Ключова ознака об’єкту дослідження – номерний знак. Саме за його значенням реалізовується виконуваний запит.

Предметом дослідження є методи ідентифікації, сегментації, розпізнавання зображень. Результатом предмету дослідження є власне інформація про транспортний засіб.

1.5.2. Високий поріг метрик точності та повноти моделі

Оскільки завдання розробленої системи полягає у ідентифікації номерного знаку, а саме кожного з його елементів, важливо щоб модель якісно розпізнавала усі об’єкти з якомога більшою ймовірністю. Точність показує долю правильно розпізнаних елементів, повнота показує долю реальних розпізнаних компонентів, які відносяться до компонентів номерного знаку. В разі отримання хибного результату, буде створюватись некоректний запит до

бази даних, тим самим повертаючи неякісний результат. Це робить модель неефективною для використання.

1.5.3. Швидкодія та ефективність розробленої системи

Швидкість отримання інформації з бази даних є важливим фактором, оскільки система дозволятиме отримувати дані про автомобіль в найкоротші терміни. Це дозволить пришвидшити існуючу структуру інформування правопорушень в державних органах. Доцільно також розглянути можливість використання API запитів з хмарними технологіями (Amazon Web Services, Microsoft Azure, Google Cloud Platform) для моніторингу ситуації в режимі реального часу.

Рисунок 1.5. Приклад Real-Time розпізнавання

1.5.4. Велика кількість якісних даних для тренування нейромережі

Сегментація елементів номерного знаку створює масив окремих об’єктів, які в подальшому піддаються розпізнаванню. Важливо, щоб система обробила якомога більше даних, для уникання конфліктів між символами. Такі ситуації можливі, коли система намагатиметься класифікувати елементи, схожі за формою, проте різні за значенням. Наприклад цифра 1 схожа за формою на літеру I, цифру 0 модель помилково може віднести до класу літер

О. Важливо навчити мережу відрізняти подібні символи, шляхом збільшення тренувальної вибірки. Варто врахувати також, що більша кількість навчальних даних призводить до довшого часу тренування моделі, проте підвищує точність та якість створеної мережі.

1.5.5. Простота використання та мінімальні системні вимоги

Розроблена система може масово використовуватись у різних сферах. Важливо створити простий функціонал без додаткових системних конфігурацій. Варто передбачити як можливість повного циклу ідентифікації транспортного засобу, шляхом фото/відео фіксації автівки, так і ручного коригування у випадку невідповідності результату моделі з реальними даними.

1.6. Постановка задачі для розробки програмного модулю.

Аналізуючи ключові вимоги та труднощі, пов’язані з практичною реалізацією модулю, виявлено, що не усі номерні знаки піддаються загальному алгоритму ідентифікації та класифікації. Прикладом таких є транзитні (червоний колір фону) та маршрутні (жовтий колір фону) номерні знаки.

Рисунок 1.6. Приклади різних типів номерних знаків

Для урахування потреб, модель може бути адаптована конкретно під замовника, в нашому випадку розглянуто загальний алгоритм процесу. Програмний модуль також вимагає великої кількості “ручної” підготовки

даних, пошук автомобілів на сайтах з продажу, оренди автівок для візуалізації результату моделі.

Дані для дослідження використано з єдиного порталу відкритих даних Міністерства цифрової трансформації, набору даних для розпізнавання цифр та літер MNIST та основних сайтів з продажу автомобілів (AutoRia, RST, Olx).

В ході вибору методів для реалізації даного питання було вирішено створити модель, яка працюватиме на основі підходу навчання з учителем, розпізнаватиме номерний знак автомобіля та створюватиме запит до бази даних з отриманням результату.

Висновки за розділом І

Підсумовуючи, можемо стверджувати, що модуль розпізнавання номерних знаків може значно пришвидшити виявлення автомобілів, які перебувають у розшуку. Це допоможе отримувати інформацію про транспортний засіб в найкоротші терміни та інформувати державні установи про час, місце перебування та стан автомобіля. Розроблена система повинна бути ефективною та аналізувати зібрану інформацію, щоденний трафік автомобілів дозволить їй навчатися на отриманих даних та покращувати ключові якісні метрики моделі.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ЗАСТОСУНКУ З ІДЕНТИФІКАЦІЇ НОМЕРНИХ ЗНАКІВ ТРАНСПОРТНИХ ЗАСОБІВ

Для кращого розуміння принципу роботи розробленої системи, необхідно більш детально дослідити усі компоненти розробленого застосунку. В цьому розділі приділено увагу архітектурі, особливостям роботи програмного модулю.

2.1. Опис архітектури програмного модулю

Система ідентифікації номерних знаків передбачає перевірку транспортного засобу в офіційних державних реєстрах. Безпосередньо сам модуль являє собою розроблений застосунок, який дозволяє самостійно виконувати перевірку інформації.

Передбачена архітектура є базовим представленням повноцінної системи (рисунок 2.1.).

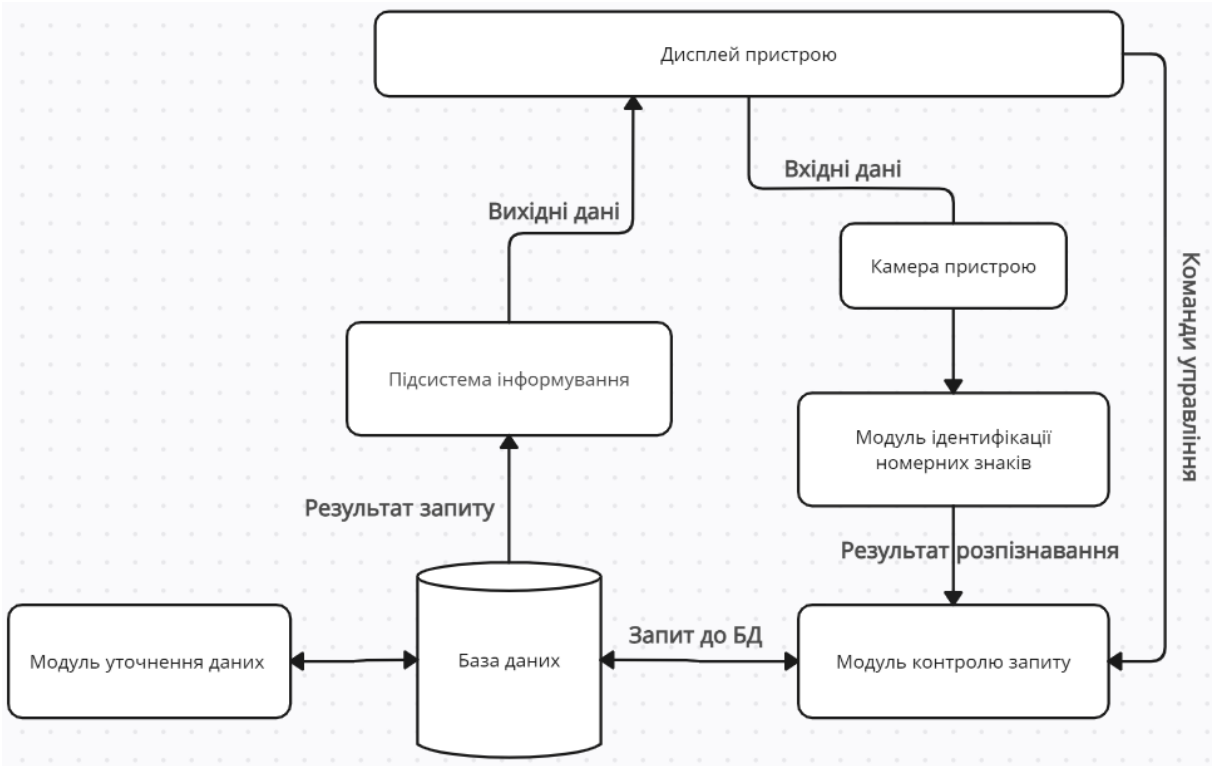

Рисунок 2.1. Архітектура програмного модулю ідентифікації ТЗ

Застосунок дозволяє доповнювати структуру додатковими мікросервісами, модулями відповідно до потреба замовника. Важливо розуміти що даний шаблон може також бути адаптований під різні інформаційні системи, сайти, компоненти програмного забезпечення, окремі елементами повноцінного продукту. В нашому випадку розглянуто подання програми в якості мобільного застосунку.

Як і більшість повноцінних систем, модуль представлений набором компонентів які є пов'язаними між собою.

Основними елементами архітектури є власне Модуль ідентифікації, модуль контролю запиту, база даних та модуль уточнення даних. Ключовим компонентом є блок "Модуль ідентифікації номерних знаків", оскільки дана робота присвячена саме цьому.

Послідовність дій користувача при використанні застосунку є наступною. На дисплеї пристрою вмикається камера, яка повинна зафіксувати фотографію номерного знаку. Після здійснення фотознімку, програма автоматично запускає модуль ідентифікації та здійснює розпізнавання елементів. Після відпрацювання модулю, здійснюється перевірка користувачем щодо коректності розпізнання даних моделлю. У випадку невідповідності, користувач може вручну уточнити інформацію та підтвердити правильність даних. Наступним кроком є створення запиту до бази даних. Запит повертає результат, після чого відпрацьовує модуль уточнення даних. У випадку виявлення автомобіля як викраденого, модуль уточнення повертає інформацію про стан розгляду справи, повідомляє про знайдення, або ж підтверджує викрадення. Підсистема інформування створює повідомлення користувачу про результат перевірки транспортного засобу та виводить сповіщення на дисплей пристрою.

Перевагою даного підходу до архітектури є швидкодія та ефективність, проте виникає окрема необхідність в постійному оновленні бази даних.

2.2. Підготовка даних для програмного модулю

Важливою складовою для якісної розробки модулю є збір, обробка інформації для навчальної та тренувальної вибірки. Якісна підготовка даних перед навчанням нейронної мережі допомагає досягти кращих результатів, поліпшує швидкість збіжності мережі та робить її більш стійкою до шумів та некоректних даних. Це має значний вплив на її ефективність, тому репрезентативність навчальних даних можуть визначати, наскільки добре програма зможе виконувати поставлену задачу.

Програмний модуль передбачає 3 набори даних:

- Інформація, що призначена для пошуку та ідентифікації ТЗ
- Фотографії літер та цифр для навчання нейронної мережі
- Фотографії ТЗ для тестування нейронної мережі

Перший датасет сформований на основі 4-х ключових таблиць, які містять інформацію про транспортний засіб (car), запис в базу даних (entity), відповідального за запис в базу (policeman) та відділок, який закріплений за транспортним засобом (police_department). Набір описує ключові характеристики ТЗ, ключові показники, за якими можна його ідентифікувати, час запису та відповідальний структурний підрозділ. Це дає змогу оцінити наскільки ефективно працює конкретна правова одиниця та визначити “шаблон” правопорушника для впровадження певний дій, які будуть націлені на відновлення правопорядку.

Структура бази даних (рисунок 2.2., рисунок 2.3.) для першого набору виглядає наступним чином.

Другий та третій датасет включає в себе фотографії, на основі яких здійснюватиметься навчання та тренування нейронної мережі. Тренування відбуватиметься на образах літер та цифр різних шрифтів, форматів, ширини, висоти символу. Тестування в свою чергу містить реальні фото транспортного засобу. Усі зображення представляються у вигляді трьохвимірного масиву, при чому масив, що відповідає за кольорове забарвлення (координати колірної

моделі RGB), піддається нормалізації. Це здійснюється для того, щоб привести дані до стандартного формату та діапазону значень. Нормалізація дозволяє збільшити стабільність, збіжність нейромережі та застерегти від викидів аномалій.

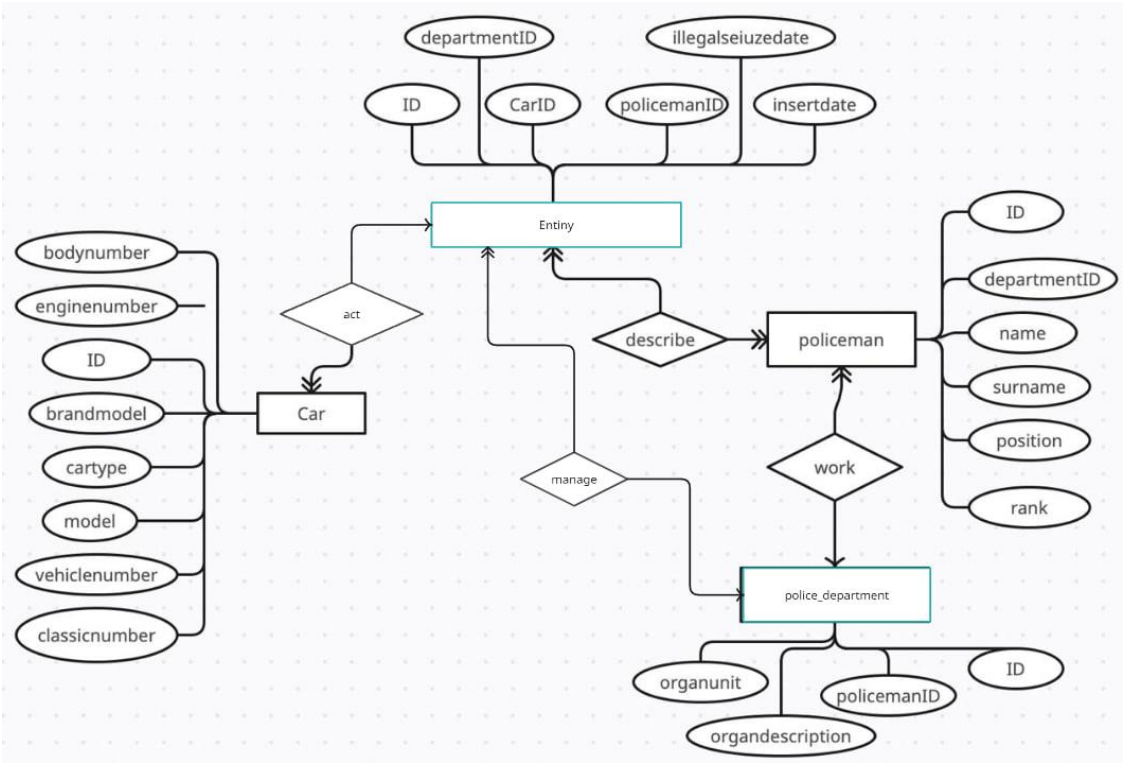

Рисунок 2.2. Концептуальна модель бази даних застосунку

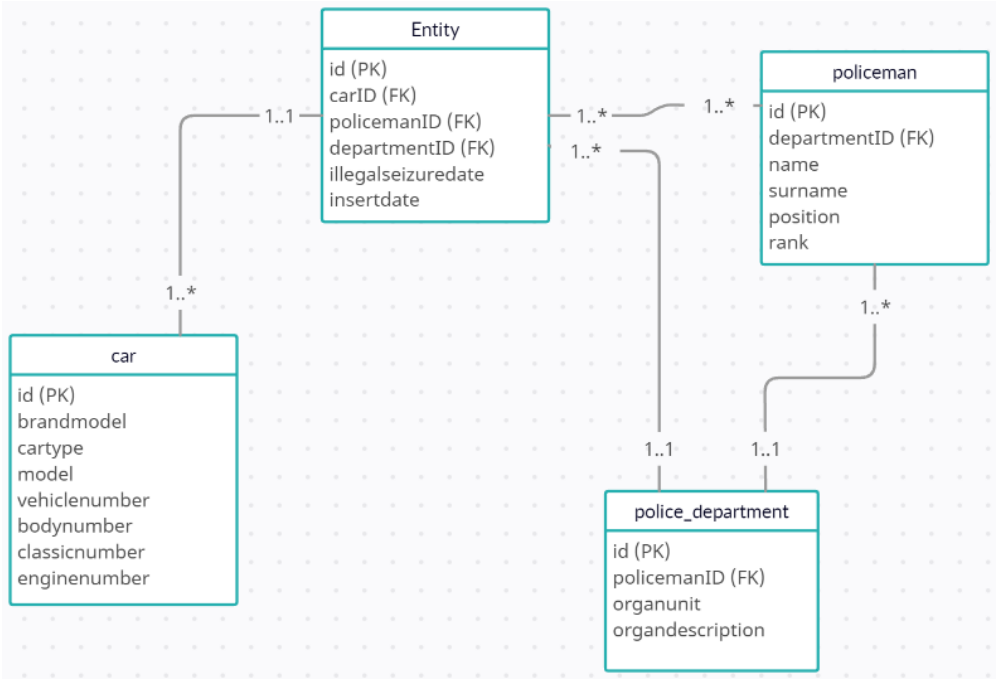

Рисунок 2.3. Реляційна модель бази даних застосунку

Таблиця car пов’язана з таблицею Entity за полем ID (car) та carID (Entity) та має зв’язок багато до одного. Тобто будь-яка кількість автомобілів може бути задіяна в ідентифікації транспортного засобу. Таблиця Entity пов’язана з таблицею policeman за полем ID (policeman) та policeman (Entity) та має зв’язок багато до багатьох. Це означає, що більше 1 поліцейського може брати участь в ідентифікації більше 1 транспортного засобу. Таблиця Entity пов’язана за таблицею police_department за полем departmentID (Entity) та id (police_department). Тобто до ідентифікації транспортних засобів може бути залучений лише 1 відділок поліції. Таблиця policeman та police_department пов’язані між собою за полями ID (police_department) та departmentID (policeman). До ідентифікації транспортних засобів можуть бути залучені поліцейські лише з 1 структурного підрозділу.

Параметричний опис структури першого набору даних представлено в таблиці 2.1.

Таблиця 2.1. Структура набору даних для ідентифікації ТЗ

Назва поля	Тип	Призначеня
id	Bigint	Унікальний ідентифікатор запису
cartype	String	Тип транспортного засобу
color	String	Колір транспортного засобу
vehiclenumber	String	Номерний знак транспортного засобу
bodynumber	String	Номер шасі транспортного засобу
classicnumber	String	VIN код транспортного засобу
enginenumber	String	Номер двигуна транспортного засобу
illegalseizuredate	Datetime	Дата незаконного заволодіння
organunit	String	Відповідальний підрозділ
insertdate	Datetime	Дата обліку інформації

2.3. Основні елементи програмного модулю

Розглянемо детальніше основні елементи архітектури, які використовуються для проектування.

2.3.1. Зчитування зображення

На вхід система отримує інформацію у вигляді зображення, яке подаватиметься на подальше опрацювання. В даному випадку програма зчитує фото із зовнішнього пристрою, проте в комерційних цілях таке рішення не буде досить ефективним, оскільки сучасні продукти використовують інші підходи для реалізації системного продукту.

Отримання інформації за допомоги HTTP-запитів до веб-служби. Модуль може використовувати дані запити, або ж певну його частину в якості передавача зображень на сервер та отримувати результат розпізнавання. Використання API стороннього сервісу. Веб-сервісна проекція програмного модулю може використовувати API для передачі зображення на сервер. Прикладами такими сервісів можуть бути AWS (S3, lambda, Step Function), Microsoft Azure, Google Cloud Platform.

Бувають випадки, коли можемо використовувати готові шаблони зображень для генерації псевдовипадкових картинок. Це може передаватися в програмний модуль через буфер пам’яті, наприклад, у вигляді масиву пікселів або матриці зображення. Варто звертатися до такого рішення тоді, коли необхідно створити велику кількість зображень схожих, за ключовими параметрами. Наприклад образ однієї літери з різними кутами нахилу, зашумленням або штучною деформацією.

2.3.2. Обробка зображення

Обробка є ключовим елементом системи який відповідає за якість та коректність розпізнавання. Даний крок можна умовно поділити на декілька етапів: препроцесинг, виділення областей для роботи, сегментація, виявлення ознак.

На етапі препроцесингу відбувається обробка інформації з метою оптимізації його для подальшого аналізу. Отримане зображення, піддається масштабуванню. Воно може бути збільшене або зменшене з метою забезпечення однакового розміру об’єктів на зображенні або пристосування зображення до конкретних вимог алгоритму. В більшості випадках фотографія робиться під певним кутом, це вимагає додатково використати процес коригування геометричної перспективи. За допомогою методів, таких як геометрична трансформація або використання відомих маркерів, зображення може бути вирівняне, щоб забезпечити пряме положення номерного знаку. Видалення шумів дозволяє підвищити точність моделі та якість даних для навчання.

Рисунок 2.4. Ключові складові обробки зображень

Шум може бути спричинений різними факторами, такими як дефект камери, неефективні методи стиснення зображення або неякісне освітлення. Застосування фільтрів, які можуть бути лінійними або не лінійними, допомагає покращити якість зображення і зробити його відповідним для подальшої обробки.

Виділення областей для роботи з зображенням включає алгоритми виявлення об’єктів за відповідними образами. Допускається можливість використання морфологічних операцій, які дозволяють виокремити області з певними геометричними характеристиками, подібними до форми номерного знаку. Використання глибинних мереж нейронних мереж дає можливість автоматично навчати модель на великій кількості зображень для виявлення номерних знаків. В даному випадку модуль визначає контури за допомоги алгоритму Канні.

Рисунок 2.5. Препроцесинг фотографій

Алгоритм Канні – один із базових алгоритмів в області комп’ютерного зору і обробки зображень. Алгоритм складається з кількох етапів. Першим кроком є згладжування зображення, для якого застосовується фільтр, наближений до гаусівського. Це допомагає знизити шум та покращити якість зображення перед подальшою обробкою.

$$B = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} * A \quad (1)$$

Фільтр Канні

$$G = \sqrt{G_x^2 + G_y^2} \quad (2)$$

$$\theta = arctg\left(\frac{G_y}{G_x}\right) \quad (3)$$

Кут нахилу градієнту

Наступним кроком використовується оператор Собеля для виявлення градієнту яскравості на зображенні. Даний оператор вимірює зміни яскравості пікселів в горизонтальному і вертикальному напрямках. Це допомагає виявити різкі зміни, що відповідають границям об’єктів.

Після виявлення градієнтів застосовується порогова обробка, яка відсікає слабкі градієнти, що можуть не представляти справжні межі об’єктів. Встановлюється два пороги – нижній та верхній, які допомагають визначити сильні і слабкі градієнти. Крайнім кроком є пошук контурів з використанням алгоритму «трасування по контурах». Цей алгоритм дозволяє з’єднати сильні градієнти в межах контурів та сформувати окремі границі об’єктів. В результаті отримуємо зображення, в якому виявлені межі об’єктів з високою точністю та мінімальним впливом шуму.

Виділені об’єкти на зображенні подаються на подальшу сегментацію, яка може здійснюватися за допомоги порогової обробки, де використовується певне порогове значення, щоб розділити зображення на дві категорії: пікселі, що відповідають коректному класу, і пікселі фону. Пікселі, які перевищують поріг, відносяться до якісних, інші ж – належать до тих, які потрібно відсікти.

Інший підхід – це сегментація за пікселями, де використовується алгоритм, який аналізує кожен піксель зображення і визначає, чи належить він до знаку чи фону. Цей підхід може базуватись на властивостях пікселів, таких як колір, текстура або яскравість, або використовувати вже існуючі класифікаційні моделі.

Другий спосіб вимагає значних ресурсів для перебору кожного пікселю зображення, тому його використання не є ефективним в нашому випадку. Програма передбачає наступну умову: якщо піксель становить більше 128 біт, відносимо його до елементу знаку (присвоюємо йому чорний колір), інакше ігноруємо (присвоюємо білий колір).

2.4. Архітектура нейронної мережі

Якісна архітектура нейронної мережі повинна містити низку згорткових, повнозв'язних шарів, велику кількість фільтрів для виявлення складних залежностей в даних. Проте глибокі мережі можуть вимагати значних ресурсів для тренування моделі, тому важливо раціонально оцінити ресурси, що використовуються.

Розмірність пулінгових шарів, підбір оптимальної функції активації допоможуть оптимізувати складну нейромережу. Пулінгові шари використовуються для зменшення розмірності карт ознак. Саме тому правильних вибір його розмірів впливає на здатність мережі до збереження важливих ознак та локалізації об'єктів у вхідних даних.

Для уникнення перенавчання, доцільно використати регуляризацію. У випадку, коли мережа вивчає тренувальні дані, дана методика спеціально погіршує результат навчання для кращого розпізнавання тестових випадків. Існує два види регуляризації L1 та L2. L1 регуляризація (Lasso) використовує суму абсолютних значень ваг в моделі, в той час, коли L2 регуляризація (Ridge) використовує суму квадратів ваг в моделі. Обидва методи регуляризації використовуються з додатковим гіперпараметром, який контролює силу регуляризації. Збільшення значення гіперпараметра призводить до більшої регуляризації і зменшення ваг моделі.

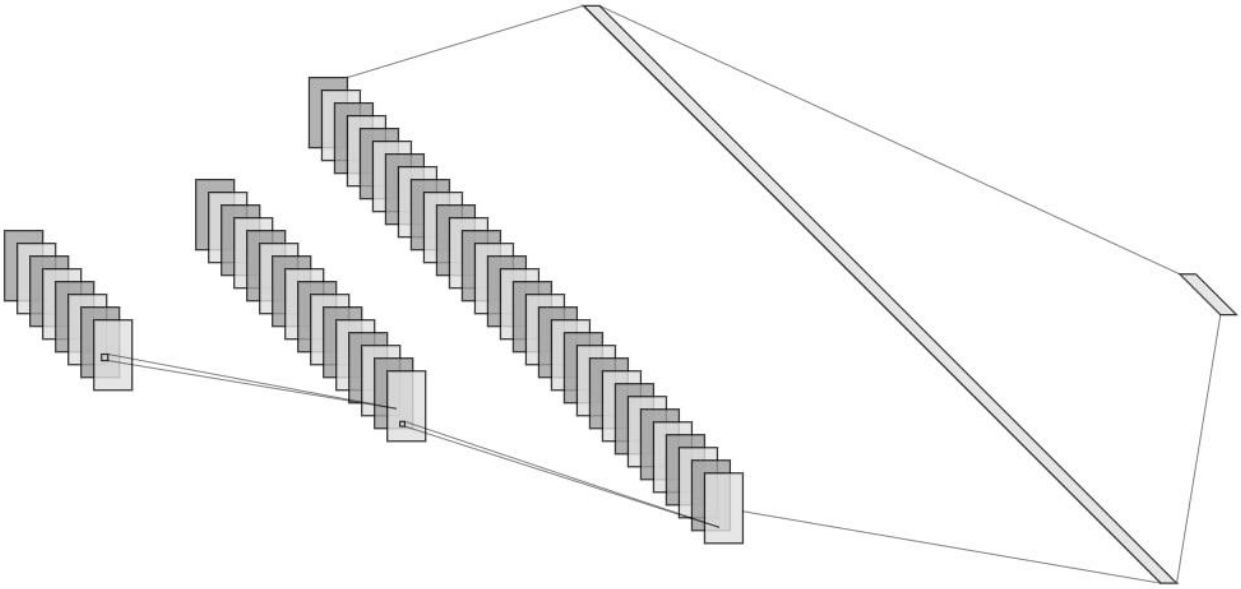

Рисунок 2.6. Архітектура згорткової нейронної мережі

Розроблена мережа складається зі згорткового шару (convolutional), який містить 64 нейрони з фільтром розміром 3 х 3. На вхід подаються зображення сірого кольору розміром 44 х 24 пікселі. Функція активації для даного шару – ReLu, вона відіграє важливу роль для виявлення нелінійних залежностей.

Наступний – шар пулінгу (pooling) з розміром пулінгу 3 х 3. На даному кроці виконується зменшення розмірності вхідного зображення та виділяються ключові області вхідного зображення, за якими навчатиметься мережа. Шар, який відповідає за уникнення перенавчання (dropout) є наступним та приймає значення в 0,25. В даному випадку відключається 25% нейронів у випадковому порядку.

Дана послідовність шарів повторюється ще двічі з послідовним збільшенням кількості фільтрів. Це дозволяє брати якомога більше факторів до уваги та робити мережу універсальною. Повнозв’язний шар реалізовує перетворення з попередніх кроків в одновимірний вектор, який складається з 36 нейронів, тобто на виході мережа зможе розпізнати 10 цифр та 26 літер.

Таблиця 2 – структура розробленої мережі

№	Шар	Кількість фільтрів/нейронів	Розмір фільтру	Розмір карти	Функція активації
	Input	-	-	44 x 24 x 1	
1	Conv2D	64	3 x 3	44 x 24 x 1	ReLu
1	MaxPooling2D		3 x 3	44 x 24 x 1	
1	Dropout	0.25			
2	Conv2D	128	3 x 3	44 x 24 x 1	ReLu
2	MaxPooling2D		3 x 3	44 x 24 x 1	
2	Dropout	0.3			
3	Conv2D	256	3 x 3	44 x 24 x 1	ReLu
3	MaxPooling2D		3 x 3	44 x 24 x 1	
3	Dropout				
	Flatten	-	-	-	-
	Dense	512	-	-	ReLu
	Dropout	0.5			
	Dense	36	-	-	Softmax

Дана структура розробленої мережі має певний потенціал для масштабування в подальшій перспективі, проте сфера використання даної системи є досить вузькою, оскільки мережа адаптована лише під розпізнавання образів номерів. В цілому дана нейромережа має потенціал для ефективного розпізнавання та щоб забезпечити оптимальну продуктивність та точність, було розглянуто додаткові аспекти, такі як оптимізація гіперпараметрів та можливість використання додаткових технік обробки зображень, такі як обтікання за контурами, нормалізація, переведення зображення в двовимірний простір.

В даному випадку також враховано контекст застосунку, тому вхідний шар приймає зображення розміром 24 x 44 пікселі, саме таким є величина одного елементу знаку на класичній фотографії. Застосунок розроблено з

урахуванням обмеження можливостей обладнання, це дозволяє зосередити та розширити можливості для застосування розробки. В цілому архітектура нейромережі дозволяє брати до уваги чимало факторів при розпізнавання, проте присутні також негативні аспекти пов’язані з розпізнаванням зашумлених або фото з дефектами. Модель не має спеціалізованих шарів або механізмів для виявлення та видалення даних нюансів. Вона зосереджена на виявленні об’єктів та патернів, що може призвести до незадовільного розпізнавання на зашумлених або пошкоджених зображеннях. Також розмір фільтрів у шарах згортки Conv2D дорівнює 3 x 3, що може бути недостатнім для ефективного виявлення інтенсивного шуму, проте це компенсується розміром вхідного зображення та великою кількістю фільтрів.

2.5. Моделювання поведінки застосунку

Для кращого розуміння роботи застосунку ідентифікації номерних знаків змодельована UML діаграма, яка зображує залежність між подіями. На основі даної діаграми можемо зробити висновок, що прослідковується послідовність процесів та кожен наступний процес напряму залежить від попереднього. Запуск програми передбачає вибір фотографії, яку необхідно подати на обробку. Вибір фотографії здійснюється шляхом відкриття активного каталогу та можливістю пошуку зображення на локальному пристрої. Обране фото дублюється на головному екрані зображення, після чого користувачу стає доступна функція обробки фотографії. Здійснюється розпізнавання знаку, результат виводить під фото на головному екрані застосунку з можливістю уточнення результатів. У випадку коректного розпізнавання, програма здійснює пошук в базі даних відразу, інакше передбачена можливість ручного вводу, якщо частина тексту була розпізнана некоректно. Результат пошуку виводиться в крайньому нижньому положенні головного вікна програми.

Рисунок 2.7. UML діаграма діяльності

Висновки за розділом ІІ

Підсумовуючи, в даному розділі було розглянуто ключові аспекти застосунку, оцінено переваги та недоліки як архітектури програми так і структури нейронної мережі. Описано основні елементи системи, надано експертну оцінку критеріям, які впливають на якість та збіжність нейромережі. Згадано про способи та методи покращення архітектури моделі за умови наявності оптимальної кількості ресурсів та вимогам поставленої задачі. Розглянуто інструментарій та середовище розробки програмного застосунку.

РОЗДІЛ 3. ПРОЕКТНО-ТЕХНОЛОГІЧНА РЕАЛІЗАЦІЯ ЗАСТОСУНКУ

3.1. Попередня обробка та дослідження даних

Даний набір даних представлений у вигляді масиву даних у вигляді JSON. Структура набору даних подається у форматі ключ-значення:

{«brandmodel»:»Shantui SD16 – 1»,

«cartype»:»Вантажний автотранспорт»,

«color»:»НЕВИЗНАЧЕНИЙ»,

«vehiclenumber»:»34226BH»,

«bodynumber»:»»,

«chassisnumber»:»CHSD16AATL1047235»,

«enginenumber»:»1220A000365»,

«illegalseizuredate»:»2022-03-24T00:00:00»,

«organunit»:»ШЕВЧЕНКІВСЬКЕ ВІДДІЛЕННЯ ПОЛІЦІЇ ПРИМОРСЬКОГО ВІДДІЛУ ПОЛІЦІЇ В МІСТІ ОДЕСІ ГУНП В ОДЕСЬКІЙ ОБЛАСТІ»,

«insertdate»:»2022-04-28T00:00:00»}

Для обробки даних такого формату доцільно подати інформацію у вигляді DataFrame:

	brandmodel	cartype	color	vehiclenumber	bodynumber	chassisnumber	enginenumber	illegalseizuredate	c
0	Shantui SD16 - 1	Вантажний автотранспорт	НЕВИЗНАЧЕНИЙ	34226BH		CHSD16AATL1047235	1220A000365	2022-03-24T00:00:00	ШЕВЧЕН ВІД ПРИМОР
1	Bobcat S650 - 1	Вантажний автотранспорт	НЕВИЗНАЧЕНИЙ	34271BH		1J951000008LC2331	A3NW14664	2022-03-24T00:00:00	ШЕВЧЕН ВІД ПРИМОР
2	DACIA - LOGAN	Легковий автотранспорт	СІРИЙ	BB3466EX	UU1LSD4GH39367660		K7JA710UE48481	2022-04-28T00:00:00	Г УПР/ НАЦІОН ПОЛІЦІЇ
3	АВТОСПЕЦПРОМ - САШМД	Легковий автотранспорт	БІЛИЙ	AP9462EC	Y79941118H9C64130		10TRJA0984824	2022-03-03T00:00:00	Г УПР/ НАЦІОН Г
4	FORD - TRANSIT	Легковий автотранспорт	СИНІЙ	BB5903CK	WF0XXXTTFX7Y85272	WF0XXXTTFX7Y85272	7Y85272	2022-04-30T00:00:00	Г УПР/ НАЦІОН ПОЛІЦІЇ

Рисунок 3.1. Представлення даних у вигляді DataFrame

Доцільно провести аналіз даних та дослідити типи полів, оскільки різні типи даних потребують різних методів та способів обробки. Для цього

визначимо розмір датасету, кількість унікальних значень, максимальні, мінімальні значення, а також частоту входження елементу.

	brandmodel	cartype	color	vehiclenumber	bodynumber	chassisnumber	enginenumber	illegalseizuredate	organunit	insertdate
count	77674	77674	77674	77674	77674	77674	77674	77674	77674	77674
unique	19097	9	192	68417	57594	27028	51383	9218	803	8754
top	DAEWOO - LANOS	Легковий автотранспорт	БІЛИЙ					2022-02-24T00:00:00	ГОЛОВНЕ УПРАВЛІННЯ НАЦІОНАЛЬНОЇ ПОЛІЦІЇ В ЛУГА...	2011-01-28T00:00:00
freq	968	46775	13613	8976	19605	50142	25681	1506	2089	488

Рисунок 3.2. Статистичний опис набору даних

З даних робимо висновок, що найбільша кількість правопорушень здійснюється проти легкових автомобілів, при цьому в більшості випадків ідентифікувати область не вдається.

Наступним кроком є підготовка даних для навчання моделі. Необхідно розділити кожен образ літери та цифри в окремий активний каталог та циклом почергово генерувати набір даних для навчання мережі.

Структура файлової системи виглядає наступним чином:

- Літери
  - A
    - 0.png
    - 1.png
    - …
  - B
    - 0.png
    - 1.png
    - …
- Цифри
  - 0
    - 0.png
    - 1,png
    - …
  - 1
    - 0.png
    - 1.png

В результаті формування набору даних, генеруємо єдину вибірку та розбиваємо її на тренувальну та тестову у співвідношення 80% на 20%.

Проаналізувавши існуючі методи нормалізації, прийнято рішення здійснити дану операцію шляхом ділення кожного еталону на евклідову норму (L2), звідси й назва – L2 нормалізація. Здійснюємо перетворення даних для якіснішого навчання моделі. Зводимо кожне значення зображення в двовимірний масив, тобто перетворюємо його в чорно-біле.

3.2. Вибір середовища та інструментарій реалізації

Для реалізації застосунку було розглянуто низку інструментів для роботи з даними, нейронними мережами та алгоритмами. Ключовим інструментом для розробки став Python. Він дозволяє використовувати широкий вибір наявних бібліотек та фреймворків для обробки та аналізу, інтегрувати з іншими мовами та інструментами, а також надає можливість створювати й масштабувати нейромережі різної складності.

Для даного інструменту обрав середовище Anaconda, оскільки дана платформа розроблена в основному для роботи з інструментами Data Science. З іншого боку Anaconda дозволяє здійснювати управління пакетами програмного забезпечення, що розширює можливості для інтерактивного програмування. Після встановлення даного інструменту, користувач отримує доступ до до середовища Jupyter разом з іншими інструментами таких як Spyder, Orange, VSCode, PyCharm, DataSpell, Rstudio.

Перелік бібліотек, використаних в ході створення застосунку:

- Pandas – бібліотека для обробки та аналізу даних. Дозволяє взаємодіяти з колонковими структурами DataFrame. Використовувалась для представлення та роботи з базою даних.
- Numpy – бібліотека для роботи з масивами та багатовимірними наборами даних.
- Matplotlib – бібліотека для роботи з інфографікою. Використовується також як вбудований модуль для інших модулів.

- OS – бібліотека для роботи з каталогами та файлами на локальному пристрої.
- OpenCV – бібліотека для обробки зображень та комп'ютерного зору. Ключовий атрибут для реалізації програмного застосунку.
- TensorFlow – бібліотека, призначена для розробки та навчання нейронних мереж. Дозволяє розгорнути гнучку інфраструктуру для обчислення чисельних операцій на багатоядерних системах, GPU та TPU. Це дозволяє розробляти складні моделі з великою кількістю параметрів та категоріальних ознак.
- Scikit-learn (Sklearn) – бібліотека машинного навчання. Дозволяє використовувати широкий вибір методів класифікації, регресії, кластеризації, тощо. Використано ключові метрики для оцінки якості моделі.
- Tkinter – бібліотека для створення графічного інтерфейсу користувача (GUI). Модуль також підтримує різні методи розміщення елементів на екрані, включаючи сіткову розмітку і розташування за допомогою пакетів.
- PIL (Python Imaging Library) – це бібліотека для роботи з зображеннями. Вона надає зручний інтерфейс для завантаження, збереження, зміни розміру, обробки та маніпуляцій з кольорами зображень. PIL підтримує різноманітні формати зображень, включаючи JPEG, PNG, GIF, BMP.

Застосунок розроблявся з поступовим виконанням окремих блоків коду. Це дозволяє дослідити окремі частини програми, незалежні один від одного. Завдяки особливостям Python, вдалося імплементувати та пов'язати нетривіальну задачу розпізнавання образів та класифікації окремих його елементів та створення програмного застосунку, який буде простим та зрозумілим у використанні для більшості користувачів.

3.3. Технічна реалізація застосунку

Наступним кроком після приведення даних до нормальної форми є створення моделі. Реалізація вищезгаданої архітектури мережі нейронної мережі відбувається з використанням бібліотеки TensorFlow. На основі класу Sequential створюються подальші шари моделі, з використанням методу add.

Після створення моделі, необхідно її скомпілювати. Компіляція здійснюється з урахуванням основних параметрів, таких як функція втрат (loss), оптимізатора (optimizer) та метрикою для оцінки якості моделі (в даному випадку accuracy). Оптимізатори в нейромережах використовуються для налаштування ваг моделі з метою мінімізації функції втрати під час тренування. В даному випадку обрано оптимізатор Adam, оскільки він є ефективним для широкого спектру завдань і моделей, не вимагає багато налаштування гіперпараметрів. Функцію втрат обрано sparse_categorial_crossentropy, так як дана функція, з множини схожих, оцінює розбіжність між прогнозованими ймовірностями класів і правильними шаблонами. Метрику обрано accuracy, оскільки це ключова характеристика моделі, важливо розуміти наскільки ефективною буде модель в розпізнаванні елементів.

Модель навчається на тренувальних даних, кількість епох – 10. Це зумовлено тим, що мережа має значну кількість шарів, тому раціональним рішенням буде прийняття рішення обрати оптимальну кількість повторень, або ж застосувати додаткову умову, яка аналізуватиме ключову метрику. Якщо точність моделі не зміниться за останні 3 епохи, модель автоматично завершує процес навчання.

```
1 model.fit(X_train, y_train, epochs=10)
```

```
Epoch 1/10
788/788 [==============================] - 22s 28ms/step - loss: 0.7099 - accuracy: 0.7960
Epoch 2/10
788/788 [==============================] - 28s 36ms/step - loss: 0.7001 - accuracy: 0.7993
Epoch 3/10
788/788 [==============================] - 28s 36ms/step - loss: 0.6816 - accuracy: 0.8028
Epoch 4/10
788/788 [==============================] - 29s 37ms/step - loss: 0.6808 - accuracy: 0.8022
Epoch 5/10
788/788 [==============================] - 28s 36ms/step - loss: 0.6650 - accuracy: 0.8082
Epoch 6/10
788/788 [==============================] - 28s 36ms/step - loss: 0.6601 - accuracy: 0.8053
Epoch 7/10
788/788 [==============================] - 29s 36ms/step - loss: 0.6519 - accuracy: 0.8085
Epoch 8/10
788/788 [==============================] - 29s 37ms/step - loss: 0.6493 - accuracy: 0.8102
Epoch 9/10
788/788 [==============================] - 28s 36ms/step - loss: 0.6382 - accuracy: 0.8138
Epoch 10/10
788/788 [==============================] - 29s 37ms/step - loss: 0.6335 - accuracy: 0.8154
```

Рисунок 3.3. Результат навчання мережі

На даному етапі, точність мережі складає 81%, що є досить непоганим показником для слабко структурованих даних. Оскільки датасет для навчання формувався вручну, завдання полягає в отриманні найбільш прийнятного значення метрики при обмежених ресурсах для тренування. Точність моделі значною мірою залежить від контексту елементів, які підлягатимуть навчанню, наприклад цифра 0 та літера О є досить схожими за формою, тому розпізнати дані еталони є задачею не тривіальною. Аналогічна ситуація з цифрою 1 та літерою І. Сформуємо звіт по класифікації для оцінки якості розпізнавання.

Таблиця 3.1. Звіт по класифікації

	precision	recall	f1-score	support
0	0.99	0.94	0.97	105
1	0.97	0.98	0.98	104
2	0.98	0.96	0.97	113
3	1.00	0.96	0.98	102
4	0.98	1.00	0.99	118
5	0.99	0.98	0.99	122
6	0.98	0.99	0.98	125
7	1.00	0.96	0.98	113
8	0.97	0.98	0.98	110
9	0.99	0.98	0.99	121
10	0.71	0.82	0.76	190
11	0.76	0.76	0.76	197

12	0.80	0.87	0.83	216
13	0.80	0.80	0.80	213
14	0.79	0.78	0.79	207
15	0.86	0.83	0.84	224
16	0.79	0.80	0.79	232
17	0.83	0.64	0.72	265
18	0.77	0.77	0.77	228
19	0.74	0.84	0.78	184
20	0.76	0.81	0.78	190
21	0.83	0.95	0.89	171
22	0.77	0.67	0.72	249
23	0.79	0.82	0.81	188
24	0.72	0.60	0.65	215
25	0.78	0.93	0.85	189
26	0.69	0.77	0.73	143
27	0.75	0.82	0.78	197
28	0.80	0.87	0.83	194
29	0.77	0.71	0.74	212
30	0.81	0.76	0.79	211
31	0.79	0.57	0.66	285
32	0.77	0.87	0.81	201
33	0.75	0.80	0.77	191
34	0.74	0.84	0.79	173
accuracy			0.81	6298
macro avg	0.84	0.84	0.84	6298
weighted avg	0.81	0.81	0.81	6298

Зі звіту можемо зробити висновок, що модель краще розпізнає цифри, ніж літери. Це прийнятний висновок, оскільки ключовою складовою номерного знаку є власне номер, комбінація з цифр. Загальний результат макро-середнього є кращим, ніж зваженого середнього, так як кожен елемент має однакову вагу. У випадку формування окремого набору з власними ваговими коефіцієнтами, точність моделі може досягти 90%.

Матриця помилок дозволяє ідентифікувати класи, для яких модель має низьку точність. За допомоги False Positive (FP) і False Negative (FN) визначено класи, які частіше плутаються або неправильно класифікуються моделлю. Це дає змогу зосередитися на поліпшенні результатів для конкретних класів. Підтверджується факт якісного розпізнавання цифр, дещо гірша ситуація з літерами. За результатами матриці помилок, робимо висновок, що більшість елементів, що піддавались розпізнаванню були визначені коректно.

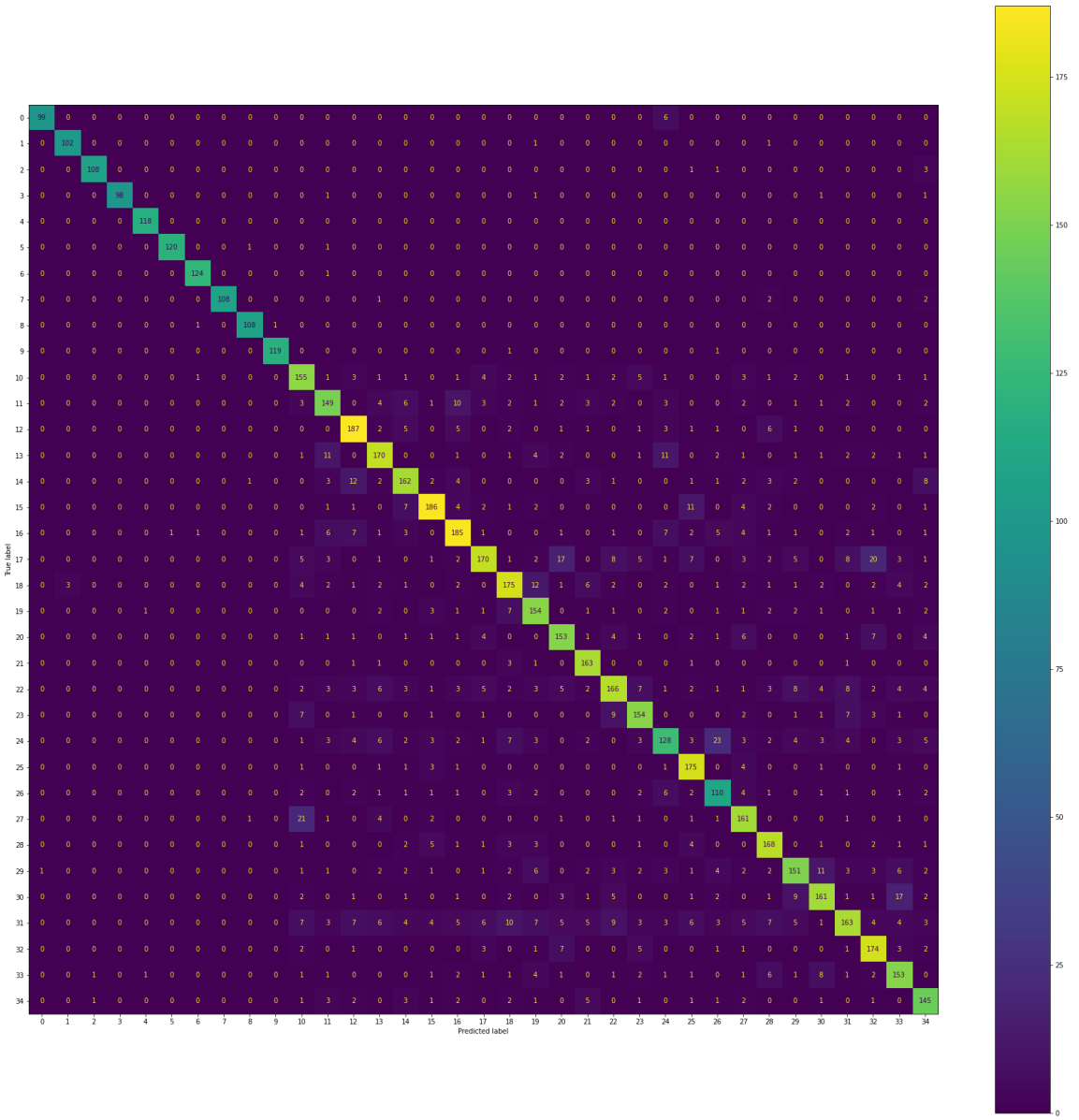

Рисунок 3.4. Матриця помилок

.

3.4. Тестування застосунку

Здійснимо тестування моделі на основі реальних фотографій автомобілів, в яких видно номерний знак. На вхід подаємо картинку, програма повинна обрати для себе робочу область з якому буде працювати (рис. 3.7).

Рисунок 3.5. Виділення робочої області

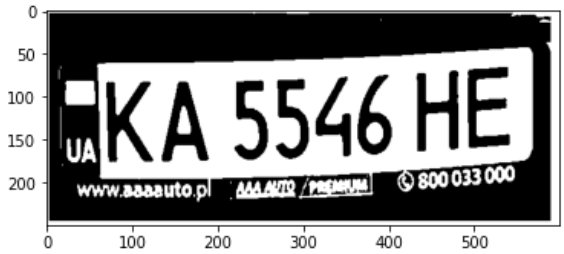

Рисунок 3.6. Перетворення робочої області

Наступним кроком є перетворення обраної області в чорно-білий формат (рис. 3.8) та здійснення сегментації основних елементів (рис. 3.9). Результуючий набір символів представлений на рис. 3.10. Ці дані переводяться в текстовий формат та передаються далі на пошук в базі даних.

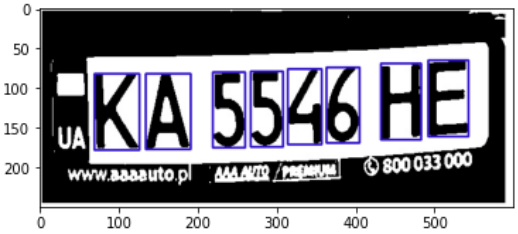

Рисунок 3.7. Здійснення сегментації

Рисунок 3.8. Набір результуючих даних

Після розпізнавання останнього результату, формується список в текстовому вигляді у форматі KA5546HE. Даний запис слугує ідентифікатором для пошуку в базі даних.

Здійснюється запит, який повертає інформацію про автомобіль. У випадку відсутності транспортного засобу, програма повідомляє про це, інакше користувач вводить дані вручну та здійснює перевірку самостійно.

Рисунок 3.9. Результати тестування

Можемо спостерігати, що застосунок визначає літери закордонних, вантажних номерних знаків. Також в ході дослідження визначено, що система спроможна розпізнавати номерні знаки причепів та громадського транспорту.

Можлива складність у визначенні елементів знаку, який піддався трансформації, пошкодженню, в якого присутні потертості або нанесений додатковий захист для рамок. Також виникають труднощі для ідентифікації знаків на жовтому, червоному фоні. Це зумовлено тим, що на етапі перетворення зображення в двовимірний масив, програма визначає контури рамки та фону однаковими, тому для покращення результатів доцільно врахувати та розглянути інші методи для даної когорти.

3.5. Інструктивний матеріал користувача для роботи з нейромережного застосунку ідентифікації номерних знаків транспортних засобів, що перебувають у розшуку

Для роботи з застосунком необхідно встановити Python та відповідне для нього середовище, яке зможе читати файли формату .ipynb. У випадку запуску програми в Google Colab, або через розширення вбудованого IDE, програма буде запущена локально у активному вікні. У випадку використання Jupyter Notebook, користувача буде направлено в браузер для запуску потрібного файлу. В файлі виконуваної програми усі частини коду виконуються послідовно, активний застосунок вимагає наступних інструкцій:

2. Обрати потрібне фото, яке необхідно розпізнати
3. Обрати фото з списку на локальному пристрої
4. Після відображення фото, натиснути обробити
5. У випадку коректного розпізнавання, відповісти, що програма розпізнала номер коректно
6. У випадку хибного розпізнавання записати номер вручну
7. Здійснити перевірку в базі даних
8. Закрити програму

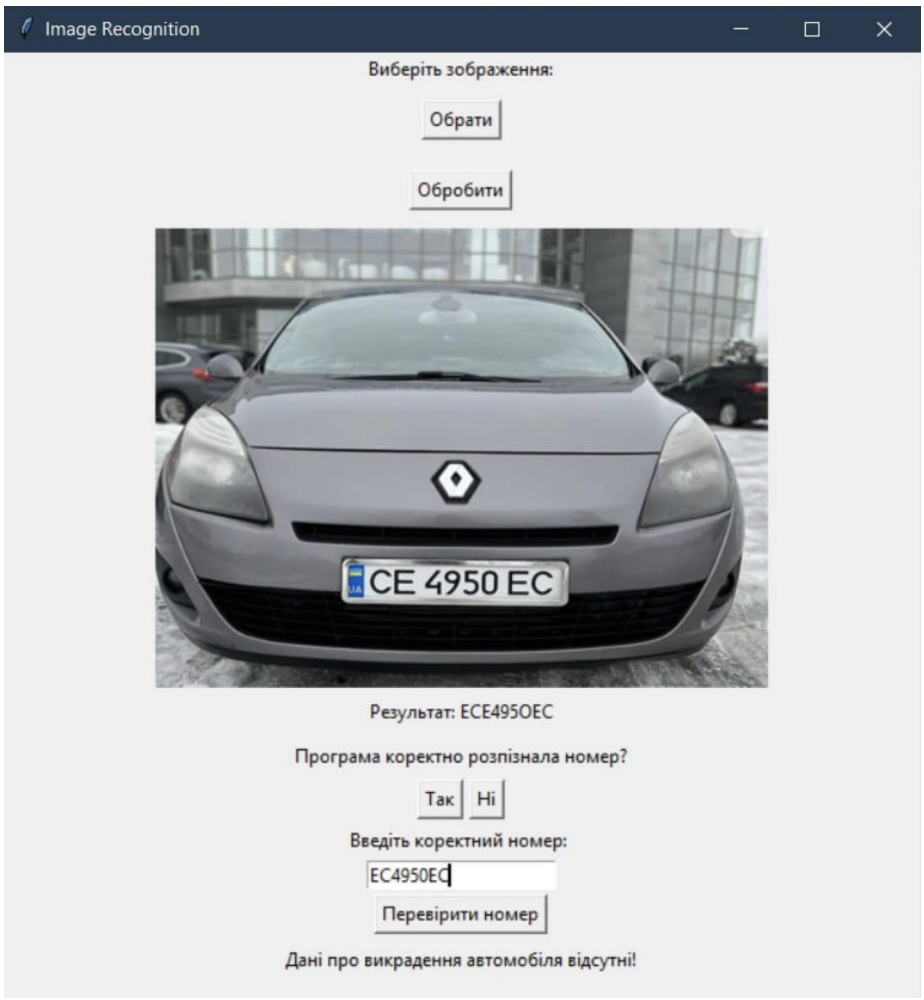

Рисунок 3.10. Вікно програми

Висновки за розділом III

Результатом даного розділу є технічний опис нейромережного застосунку ідентифікації номерних знаків транспортних засобів, що перебувають у розшуку. Описано структуру даних, ключові функції та методи для роботи з нейронними мережами. Розглянуто основні метрики оцінки якості моделі та метрики оцінки для класифікації. Досліджено переваги та недоліки, пов’язні зі складнощами в реалізації. Описано підходи та методи, які допоможуть покращити модель в подальшій модифікації.

Здійснено тестування застосунку, порівняли результати роботи з фотографіями номерних знаків різних типів транспортних засобів, країн. Розглянуто зображення різної якості, освітленості, розміру та зашумленості.

Описано структуру застосунку, послідовність дій для коректного

відпрацювання програми також створено інструктивний матеріал для користувача, націлений на висвітлення ключових елементів додатку.

ВИСНОВКИ

В результаті виконання дипломної роботи було створено нейромережний застосунок застосунок ідентифікації номерних знаків транспортних засобів, що перебувають у розшуку.

Проведено дослідження актуальності вирішуваної проблеми. Оцінено переваги та можливі труднощі в реалізації програмного застосунку. Описано сфери діяльності, які потенційно можуть бути зацікавлені в даному рішенні.

Здійснено аналіз існуючих методів та рішень, націлених на ідентифікацію, класифікацію об’єктів. Розглянуто можливі способи оптимізації даних методів.

Проведено функціональний аналіз, визначено ключові вимоги до програмного застосунку. Розроблено власну модель нейронної мережі та створено архітектуру, в якій чітко визначені зв’язки між компонентами. Описано модель бази даних, яка використовується в додатку з можливістю її наповнення. Програму розроблено відповідно до сучасних стандартів з використанням провідних фреймворків та мови програмування Python.

Даний застосунок здатний з високою точністю ідентифікувати номерні знаки транспортних засобів, що перебувають у розшуку. Він демонструє високу швидкодію та надійність у різних умовах освітлення та розміщення номерних знаків на транспортних засобах. Він є ефективним інструментом для правоохоронних органів, допомагаючи полегшити та прискорити процес виявлення злочинних транспортних засобів, сприяючи підвищенню безпеки громадськості.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Automatic Number Plate Recognition (ANPR) [Електронний ресурс] // Режим доступу: https://shorturl.at/vyJUW
2. Research on Improved Canny Edge Detection Algorithm [Електронний ресурс] // Режим доступу: https://shorturl.at/dtwOV
3. Image Noise Reduction and Filtering Techniques [Електронний ресурс] // Режим доступу: https://www.ijsr.net/archive/v6i3/25031706.pdf
4. What Is Computer Vision & How Does it Work? An Introduction [Електронний ресурс] // Режим доступу: https://shorturl.at/gnBOP
5. Державний портал відкритих даних [Електронний ресурс] // Режим доступу : https://shorturl.at/bejoQ
6. За якими принципами формуються номерні знаки в Україні [Електронний ресурс] // Режим доступу: https://shorturl.at/kwK23
7. NumPy documentation [Електронний ресурс] // Режим доступу: https://numpy.org/doc/
8. TensorFlow documentation [Електронний ресурс] // Режим доступу: https://www.tensorflow.org/api_docs
9. Pandas documentation [Електронний ресурс] // Режим доступу: https://pandas.pydata.org/docs/
10. Learning OpenCV: Computer Vision with the OpenCV Library First Edition
11. Realtime Number Plate Detection using Yolov7 – Easiest Explanation [Електронний ресурс] // Режим доступу: https://machinelearningprojects.net/number-plate-detection-using-yolov7/
12. Lane detection with NumPy [Електронний ресурс] // Режим доступу: https://shorturl.at/gzKPS
13. Machine Learning with Neural Networks An Introduction for Scientists and Engineers [Електронний ресурс] // Cambridge University Press // Режим доступу: https://www.cambridge.org/core/books/machine-learning-with-neural-networks/0028D1883AF842C81340508119AB6491

14. Computer Vision: Algorithms And Applications by Richard Szeliski. [Книга]
15. Computer Vision Algorithms You Should Know About [Електронний ресурс] // Режим доступу: https://datagen.tech/guides/computer-vision/algorithms/
16. YOLOv7 model for computer vision (GitHub) [Електронний ресурс] // Режим доступу: https://github.com/WongKinYiu/yolov7
17. Data processing – Techniques, Concepts and Steps to Master [Електронний ресурс] // Режим доступу: https://www.projectpro.io/article/data-preprocessing-techniques-and-steps/512#mcetoc_1fj6632118
18. Data ROI: How to Estimate the Value of Your Data & Analytics Projects [Електронний ресурс] // Режим доступу: https://eleks.com/blog/data-roi-data-analytics-projects/#:~:text=ROI%2C%20or%20return%20on%20investment,income%20divided%20by%20total%20investment.
19. Python Machine Learning By Example: The easiest way to get into machine learning (English Edition) 1st Edition, Kindle Edition [Книга]
20. Practical Statistics for Data Scientists: 50 Essential Concepts Paperback [Книга]
21. What are data insights? [Електронний ресурс] // Режим доступу: https://data-science-ua.com/blog/what-are-data-insights/
22. What are convolutional neural networks? (IBM) [Електронний ресурс] // Режим доступу: https://www.ibm.com/topics/convolutional-neural-networks
23. CNN Architecture: Explaining 5 Layers of Convolutional Neural Network. [Електронний ресурс] // Режим доступу: https://www.upgrad.com/blog/basic-cnn-architecture/
24. Python 3.11.3. Documentation. [Електронний ресурс] // Режим доступу: https://docs.python.org/3/
25. Best UI Graphics Tool For Python Developers With Starter Code. [Електронний ресурс] // Режим доступу: https://towardsdatascience.com/7-best-ui-graphics-tools-for-python-developers-with-starter-codes-2e46c248b47c

ДОДАТКИ

Відображення основних показників набору даних

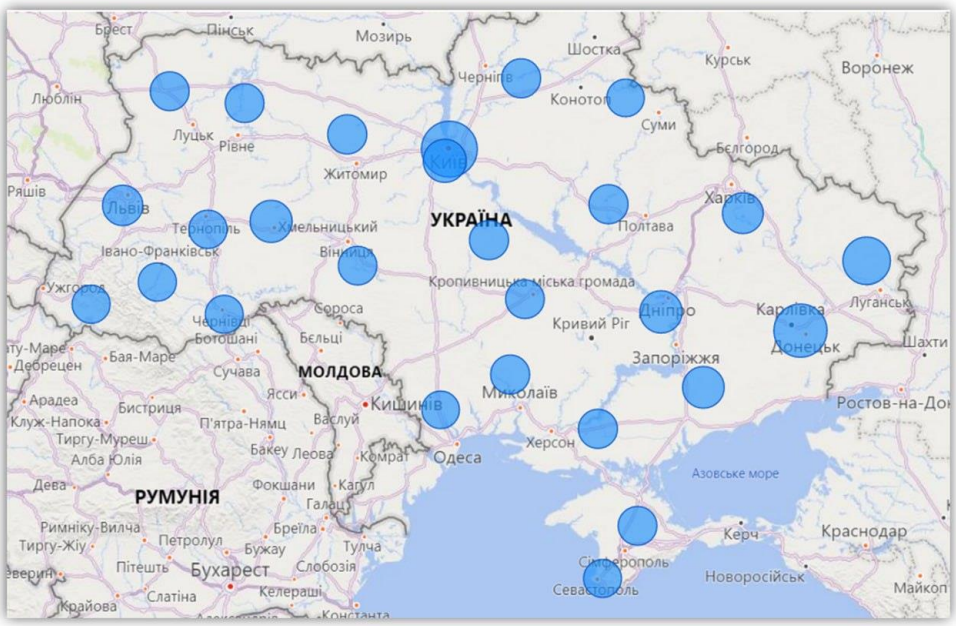

Рисунок 1. Розподіл правопорушень за областями

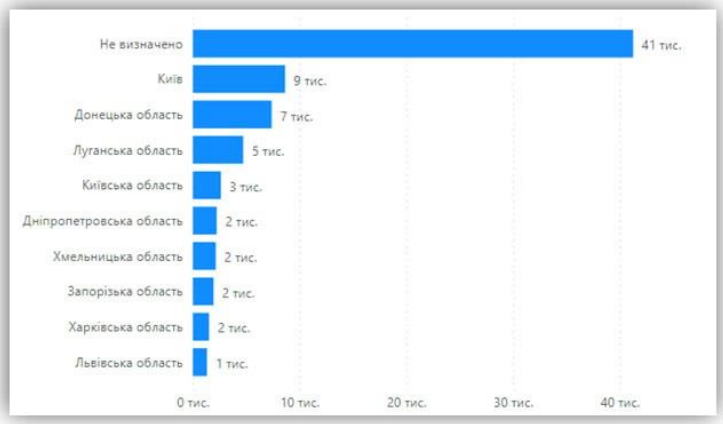

Рисунок 2. Топ областей з найбільшою кількістю правопорушень

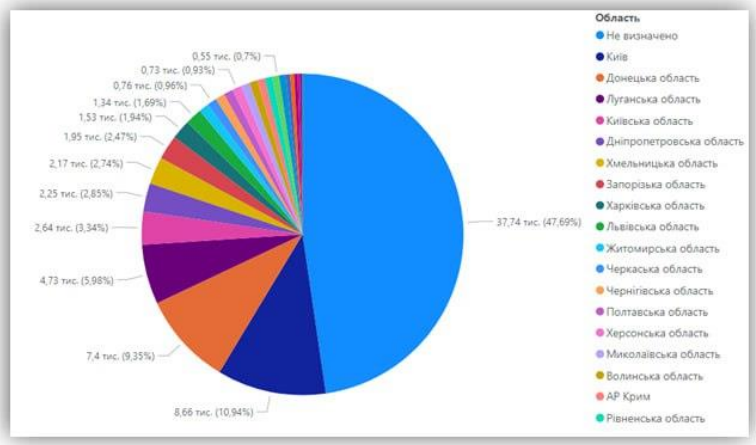

Рисунок 3. Рейтинг областей за кількістю правопорушень

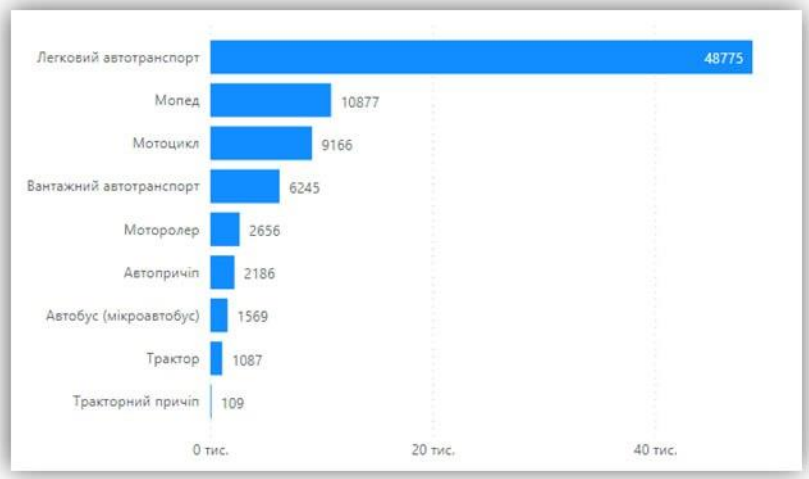

Рисунок 4. Топ типів ТЗ що піддаються викраденню

Лістинг програми нейромережного застосунку ідентифікації номерних знаків транспортних засобів, що перебувають у розшуку

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import os
import cv2
import time
import tensorflow as tf
from tkinter import *
from tkinter import messagebox
from tkinter import filedialog
from PIL import Image, ImageTk
from IPython.display import display
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score, confusion_matrix,
classification_report, accuracy_score, ConfusionMatrixDisplay, roc_auc_score
from tensorflow.keras import optimizers
from tensorflow.keras.models import Sequential
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.layers import Dense, Flatten, MaxPooling2D,
Dropout, Conv2D

df = pd.read_json("C:\Coursova\carswanted.json")

df.head()

df.describe()

path = "C:\\Coursova\\archive\\training_data\\"
folders_numbers = '0123456789'
folders_letters = 'ABCDEFGHIJKLMNOPQRTUVWXYZ'
alphabet = folders_numbers + folders_letters

x = []
y = []

for i in range(len(os.listdir(f'{path}\\numbers\\'))):
    idx = i
    for j in range(len(os.listdir(f'{path}\\numbers\\{i}'))):
        image = cv2.imread(f'{path}\\numbers\\{i}\\{idx}.png')
        grey = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        resized = cv2.resize(grey, (24, 44))
        toBinary = np.where(resized > 128, 255, 0)
        x.append([toBinary])
        y.append(i)
        idx += 10

for i in range(len(os.listdir(f'{path}\\letters\\'))):
    for j in
range(len(os.listdir(f'{path}\\letters\\{folders_letters[i]}\\'))):
        image =
cv2.imread(f'{path}\\letters\\{folders_letters[i]}\\{j}.png')
        grey = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        resized = cv2.resize(grey, (24, 44))
        toBinary = np.where(resized > 128, 0, 255)
        x.append([toBinary])
        y.append(10+i)
```

```
      x = np.array(x)
      y = np.array(y)

      X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.2,
random_state=42)

      X_train = tf.keras.utils.normalize(X_train, axis=1)
      X_test = tf.keras.utils.normalize(X_test, axis=1)

      model = Sequential()
      model.add(Conv2D(64, (3, 3), input_shape=(1, 44, 24), activation='relu',
padding='same'))
      model.add(MaxPooling2D(pool_size=(3, 3), padding='same'))
      model.add(Dropout(0.2))

      model.add(Conv2D(128, (3, 3), activation='relu', padding='same'))
      model.add(MaxPooling2D(pool_size=(3, 3), padding='same'))
      model.add(Dropout(0.2))

      model.add(Conv2D(256, (3, 3), activation='relu', padding='same'))
      model.add(MaxPooling2D(pool_size=(3, 3), padding='same'))
      model.add(Dropout(0.15))

      model.add(Flatten())
      model.add(Dense(512, activation='relu'))
      model.add(Dropout(0.5))
      model.add(Dense(36, activation='softmax'))

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

      model.fit(X_train, y_train, epochs=10)

      model.save("number_plate_recognition_model")

      y_predicted = model.predict(X_test).argmax(axis=1)

      print(classification_report(y_predicted, y_test))

      conf = confusion_matrix(y_predicted, y_test)
      cmp = ConfusionMatrixDisplay(conf)
      fig, ax = plt.subplots(figsize=(30,30))
      cmp.plot(ax=ax)

      plateCascade                                                              =
cv2.CascadeClassifier("C:\Coursova\haarcascade_plate_number.xml")

      def detect_plate(image):
          plateImg = image.copy()
          roi = image.copy()
          plateRect  =  plateCascade.detectMultiScale(plateImg,scaleFactor=1.2,
minNeighbors = 7) #, minSize=(25,25))
          for (x, y, w, h) in plateRect:
              roi_ = roi[y:y+h, x:x+w, :]
              platePart = roi[y:y+h, x:x+w, :]
              cv2.rectangle(plateImg, (x,y),(x+w, y+h), (0, 255, 0), 3)
          return plateImg, platePart

      def show(img):
          img_ = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
          plt.imshow(img_)
          plt.show()

      def find_contours(dimensions, img):
```

```
          #finding dimensions of all countours
          cntrs,        _     =     cv2.findContours(img.copy(),     cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)

          lower_width = dimensions[0]
          upper_width = dimensions[1]
          lower_height = dimensions[2]
          upper_height = dimensions[3]
          cntrs = sorted(cntrs, key=cv2.contourArea, reverse=True)[:15]

          c1 = cv2.imread('contour.jpg')

          x_cntr_list = []
          target_contours = []
          img_res = []
          for cntr in cntrs:
              intX, intY, intWidth, intHeight = cv2.boundingRect(cntr)

              if intWidth > lower_width and intWidth < upper_width and intHeight
> lower_height and intHeight < upper_height:
                  x_cntr_list.append(intX)
                  char_copy = np.zeros((44, 24))

                  char = img[intY:intY+intHeight, intX:intX+intWidth]
                  char = cv2.resize(char, (20, 40))
                  cv2.rectangle(c1,      (intX,      intY),      (intWidth+intX,
intY+intHeight), (50, 21, 200), 2)
                  plt.imshow(c1, cmap='gray')
                  char = cv2.subtract(255, char)
                  char_copy[2:42, 2:22] = char
                  char_copy[0:2, :] = 0
                  char_copy[:, 0:2] = 0
                  char_copy[42:44, :] = 0
                  char_copy[:, 22:24] = 0
                  img_res.append(char_copy)

          plt.show()
          indices     =     sorted(range(len(x_cntr_list)),     key=lambda     k:
x_cntr_list[k])
          img_res_copy = []
          for idx in indices:
              img_res_copy.append(img_res[idx])
          img_res = np.array(img_res_copy)

          return img_res

      def segment_characters(image):
          img_lp = cv2.resize(image, (600, 250))
          img_gray_lp = cv2.cvtColor(img_lp, cv2.COLOR_BGR2GRAY)
          _,    img_binary_lp    =    cv2.threshold(img_gray_lp,    150,    255,
cv2.THRESH_BINARY+cv2.THRESH_OTSU)
          img_binary_lp = cv2.erode(img_binary_lp, (3,3))
          img_binary_lp = cv2.dilate(img_binary_lp, (3,3))

          LP_WIDTH = img_binary_lp.shape[0]
          LP_HEIGHT = img_binary_lp.shape[1]
          img_binary_lp[0:3, :] = 255
          img_binary_lp[:, 0:3] = 255
          img_binary_lp[245:250, :] = 255
          img_binary_lp[:, 590:600] = 255

          dimensions = [LP_WIDTH/10, LP_WIDTH/2, LP_HEIGHT/10, 2*LP_HEIGHT/3,
LP_HEIGHT/10, 2*LP_HEIGHT/3]
          plt.imshow(img_binary_lp, cmap='gray')
```

```
            plt.show()
            cv2.imwrite('contour.jpg', img_binary_lp)

            char_list = find_contours(dimensions, img_binary_lp)

            return img_binary_lp, char_list

        inputImg = cv2.imread("C:\\Coursova\\cars_ua\\car72.jpg")    # 14, 17,
36(en), 63, 49, 55 (curve)
        inpImg, plate = detect_plate(inputImg)
        show(inpImg)
        pic, val = segment_characters(plate)

        for i in range(len(val)):
            plt.subplot(1, len(val)+1, i+1)
            plt.imshow(255-val[i], cmap='gray')
            plt.axis('off')

        l = []
        for item in val:
            idx = np.argmax(model.predict(np.array([[255-item]])))
            l.append(alphabet[idx])

        number = ''.join(i for i in l)
        print(number)

        class App:
            def __init__(self, root):
                self.root = root
                self.root.title("Image Recognition")
                self.root.geometry("600x700")
                self.image = image
                self.NumberPlate = None

                # —твореннЄ елемент≥в GUI
                self.label = Label(root, text="¬ибер≥ть зображеннЄ:")
                self.label.pack()

                self.btn_browse       =       Button(root,       text="ќбрати",
command=self.browse_image)
                self.btn_browse.pack(pady=10)

                self.btn_process      =      Button(root,      text="ќбробити",
command=self.process_image, state=DISABLED)
                self.btn_process.pack(pady=10)

                self.image_frame = Frame(root)
                self.image_frame.pack()

                self.label_result = Label(root)
                self.label_result.pack(pady=3)

                self.label_check = Label(root)
                self.label_check.pack(pady=5)

                self.buttonFrame = Frame(root)
                self.buttonFrame.pack()

                self.correct_result = Label(root)
                self.correct_result.pack(pady=3)

                self.result = Label(root)

                self.entry = Entry(root)
```

```
        def browse_image(self):
            # ¬≥дкритт€ д≥алогового в≥кна дл€ вибору зображенн€
            filename = filedialog.askopenfilename(title="¬ибер≥ть
зображенн€", filetypes=[("Image files", "*.jpg;*.jpeg;*.png")])

            if filename:
                # «авантаженн€ та в≥дображенн€ зображенн€ у GUI
                self.image = Image.open(filename)
                self.image = self.image.resize((400, 300))  # «м≥н≥ть розм≥р
зображенн€ за потреби
                self.photo = ImageTk.PhotoImage(self.image)
                self.image_label = Label(self.image_frame, image=self.photo)
                self.image_label.pack()

                # jктивац≥€ кнопки "ќбробити"
                self.btn_process.config(state=NORMAL)
                self.image = filename

        def process_image(self):
            # ¬иклик функц≥ь з модулю дл€ обробки зображенн€
            inpImg, plate = detect_plate(cv2.imread(self.image))
            pic, val = segment_characters(plate)

            l = []
            for item in val:
                idx = np.argmax(model.predict(np.array([[255-item]])))
                l.append(alphabet[idx])

            self.label_result.config(text="-езультат: " + ''.join(i for i in
l))
            self.label_check.config(text="Програма    коректно    розп≥знала
номер?")

            self.NumberPlate = ''.join(i for i in l)

            self.btn_check_true   =   Button(self.buttonFrame,   text="Так",
command=self.check_true)
            self.btn_check_true.pack(side='left', padx=2)

            self.btn_check_false   =   Button(self.buttonFrame,   text="Ќ≥",
command=self.check_false)
            self.btn_check_false.pack(side='left', padx=2)


        def check_true(self):

            IsEmpty = df.loc[df['vehiclenumber'] == self.NumberPlate].empty
            if IsEmpty:
                self.correct_result.config(text="ƒан≥     про     викраденн€
автомоб≥л€ в≥дсутн≥!")
            else:
                display(df.loc[df['vehiclenumber'] == self.NumberPlate])

        def get_text(self):
            inputText = self.entry.get()
            self.NumberPlate = inputText

            self.result.pack(pady=3)

            IsEmpty = df.loc[df['vehiclenumber'] == self.NumberPlate].empty
            if IsEmpty:
                self.result.config(text="ƒан≥   про   викраденн€   автомоб≥л€
в≥дсутн≥!")
```

```
            else:
                self.result.config(text    =df.loc[df['vehiclenumber']    ==
self.NumberPlate])

        def check(self):
            IsEmpty = df.loc[df['vehiclenumber'] == self.NumberPlate].empty
            if IsEmpty:
                self.result.config(text="ƒан≥   про   викраденнЄ   автомоб≥лЄ
в≥дсутн≥!")
            else:
                self.result.config(text=self.NumberPlate)

        def check_false(self):

            self.correct_result.config(text="¬вед≥ть коректний номер: ")

            self.entry.pack()

            self.btn_get_text   =   Button(root,   text="серев≥рити   номер",
command=self.get_text)
            self.btn_get_text.pack(pady=3)



    if __name__ == "__main__":
        root = Tk()
        root.configure()
        app = App(root)
        root.mainloop()
```