Ministry of Science and Higher Education of the Republic of Kazakhstan

SDU University

Chazhabaev Ablaykhan

Comprehensive Comparative Analysis of Machine Learning Algorithms for Vehicle Number Plate Detection in Challenging Real-World Environments

THESIS

Presented in Partial Fulfilment for the

Degree of Master of Technical Science in Computer Science

(degree code: 7M06102)

Department of Computer Science

Faculty of Engineering and Natural Sciences

Supervisor: **Sadyk Ualikhan**

Kaskelen, June 2024

SDU University
Faculty of Engineering and Natural Sciences
Department of Computer Science / Department of Mathematics and Natural Sciences

Dean of Faculty of Engineering and Natural Sciences

Assistant Professor, PhD. Akhmedov Ramis

____

«04» __________06__________ 2024

Topic of the thesis:

Comprehensive Comparative Analysis of Machine Learning Algorithms for Vehicle Number Plate Detection in Challenging Real-World Environments

Thesis submitted as part of the requirements for the award of the MSc in "7M06102 - Computer Science", SDU University,

Head of Department Zhanar Mukash

Academic Supervisor Sadyk Ualikhan

Master student Chazhabaev Ablaykhan

Kaskelen, 2024

Declaration

I confirm that this is my own work and the use of all material from other sources has been properly and fully acknowledged.

Ablaykhan Chazhabaev

June, 2024

Acknowledgements

I want to thank my supervisor MS Ualikhan Sadyk for valuable and invaluable help during my master's thesis work. Your professional guidance, expertise and mentorship have been invaluable to my education and success in the process. Thanks to your guidance and support, I have been able to expand my knowledge and skills in computer education and to formulate and develop my research . Your mentoring has helped me understand complex is- sues, find the right methods, and achieve meaningful results. I am immensely grateful for your trust, understanding and encouragement that you have given me throughout the process of working on my dissertation.

Dedication

This thesis is dedicated to:

My supervisor and many other for their support, help, sense of humour and useful comments for improving this research work .

Abstract

Automatic vehicle license plate detection (ALPR) plays a crucial role in various , such as traffic control, safety systems and electronic payment collection. However, the real environment creates significant problems for ALPR systems due to factors such as different lighting conditions, occlusion, poor image quality and various license plate formats. The study conducted a comprehensive comparative analysis of machine learning algorithms to determine vehicle licence plates in such complex scenarios. We explore both traditional and deep approaches to learning, assessing their strengths and weaknesses in solving real world problems. The analysis takes into account such factors as accuracy, processing speed. We hope our little research contributes to the development of such systems.

Аңдатпа

Автоматты табуды, нөмірлік белгілері көлік құралдарын (ALPR) шешуші рөл атқарады әр түрлі сияқты қозғалысын басқару, қауіпсіздік жүйесін және электрондық төлемдерді жинау. Дегенмен, нақты сәрсенбі жасайды елеулі проблемалар жүйелері үшін ALPR үшін мұндай факторлар ретінде әр түрлі жарық жағдайларына, атеросклерозды тарылулары, нашар кескін сапасы және әр түрлі форматтары нөмірлік белгілер. Зерттеу өткізілді жан-жақты салыстырмалы талдау алгоритмдердің машиналық оқыту анықтау үшін нөмірлік белгілері көлік құралдарын мұндай күрделі сценарийде. Біз зерттеп дәстүрлі, сондай-ақ терең көзқарас бағалай отырып, олардың күшті және әлсіз жақтары проблемаларды шешуге, нақты әлем. Талдау ескереді сияқты факторлар дәлдігі, жылдамдығы өңдеу. Деп үміттенеміз біздің шағын зерттеу дамуына ықпал етеді мұндай жүйелер.

Абстракт

Автоматическое обнаружение номерных знаков транспортных средств (ALPR) играет решающую роль в различных , таких как управление движением, системы безопасности и электронный сбор платежей. Тем не менее, реальная среда создает значительные проблемы для систем ALPR из-за таких факторов, как различные условия освещения, окклюзии, плохое качество изображения и различные форматы номерных знаков. В исследовании проведен всесторонний сравнительный анализ алгоритмов машинного обучения для определения номерных знаков транспортных средств в таких сложных сценариях. Мы исследуем как традиционные, так и глубокие подходы к обучению, оценивая их сильные и слабые стороны в решении проблем реального мира. Анализ учитывает такие факторы, как точность, скорость обработки. Надеемся наше небольшое исследование способствует развитию таких систем.

Abbreviations

ANPR — Automatic number-plate recognition

ML — Machine Learning

YOLO — You Only Look Once

Table of Contents

Chapter 1

Background and motivations

1.1 Introduction

The widespread use of machine learning [1] techniques and their application to computer vision has led to significant advances in various fields. Among them, the detection of vehicle licence plates is important in modern surveillance and safety systems. Accurate and efficient license plate detection is a key component of traffic control, law enforcement and parking access control. However, real-world problems such as differences in lighting, weather and plate design require reliable and adaptable solutions.

Our research is crucial because it seeks to bridge the gap between the theoretical advancements in machine learning and the practical needs of real-world situations. While machine learning algorithms have shown impressive capabilities in controlled environments, their performance in complex, uncontrolled settings has been inconsistent. Our study is dedicated to addressing this vital issue by assessing and contrasting various algorithms for license plate detection, with a particular focus on their ability to adapt to real-world problems. A successful analysis and evaluation of these algorithms under varying lighting conditions, adverse weather situations, and different license plate designs could greatly enhance the efficiency of surveillance, traffic management, and law enforcement systems [2]. In this introduction, we aim to provide the context, relevance, importance, and objectives of our research.

1.2 Motivation

The ability to accurately detect vehicle licence plates is fundamental to a wide range of applications ranging from law enforcement and traffic management to toll and parking systems. With the expansion of urban areas and the increase in the number of vehicles, the demand for effective and reliable Automatic Licence Plate Recognition (ANPR) systems has increased significantly [3]. However, real-world conditions create numerous problems that can hamper the functioning of these systems.

Factors such as changing lighting conditions, occlusions caused by other vehicles or objects, deformation of licence plates and different design of plates make it difficult to detect licence plates. In these circumstances, traditional methods often make it difficult to maintain high accuracy and efficiency, underlining the need for more sustainable solutions.

Machine learning algorithms have shown great promise in solving these problems, offering advanced techniques to improve detection accuracy and reliability. Despite their potential, there is a lack of comprehensive comparative analysis to assess the effectiveness of different machine learning approaches in different and complex real-world scenarios [4].

This study is intended to fill this gap. By systematically comparing various machine learning algorithms, including both traditional computer vision and modern deep learning models, The study aims to identify the most effective methods of detecting vehicle licence plates. Understanding the strengths and weaknesses of different algorithms will give developers, researchers, and practitioners valuable information to help them choose and implement the most appropriate solutions for their particular needs.

1.3 Aims and objectives

The main objective of this study is to conduct a comprehensive comparative analysis of machine learning algorithms for the recognition of vehicle licence plates in difficult real conditions. The following steps will be taken to achieve this goal. First, data will be collected by analysing different platforms and extracting the necessary data. The data would then be carefully processed and standardized. On the basis of these prepared data, a detailed analysis will be conducted to compare the effectiveness of the different machine learning algorithms.

My goal is to do a comprehensive comparative analysis of machine learning algorithms to determine vehicle license plates in difficult real world conditions. To this end, the following actions will be undertaken:

- Platform analysis .
- Collection of data from supervisors .
- Analyzing and processing data .
- Study of different algorithms
- Do comprehensive comparative analysis

Chapter 2

Literature review

The history of the development of Automatic License Plate Recognition (ANPR) technology dates back to the second half of the 20th century, when in the late 1970s, leading research was carried out at the British Transport Research Laboratory. The ANPR concept was originally designed to automate the identification of vehicle numbers using images and optical character recognition. System prototypes in the 1980s were the first step in the development of ANPR, using basic character recognition algorithms to decode alphanumeric combinations in images [5].

The 1990s marked a significant breakthrough in ANPR when technology became commercially available and applied in various sectors. This coincided with improvements in digital imaging technology and computing capabilities, which allowed for more sophisticated ANPR systems with real-time number recognition capabilities. Commercial use of ANPR systems has become common in law enforcement, road control and toll collection.

At the turn of the 21st century, ANPR technology had already established itself as a key element of modern transport and security infrastructure. The rapid development of machine learning and artificial intelligence gave impetus to the further development of ANPR, increasing the efficiency and accuracy of recognition systems. The integration of ANPR systems with cloud computing and data analytics has opened up new possibilities for real-time monitoring, predictive analysis and operational decision-making in the field of traffic control and law enforcement. At present, ANPR technology continues to evolve, driven by continuous research and innovation, which promises further improvements in efficiency, accuracy and applicability in various fields.

With the development of technology, especially machine and deep learning algorithms, there is the emergence of many ANPR models that open up new possibilities for creating more accurate, efficient and universal license plate recognition systems such as YOLO[b], SSD[b], Recurrent Neural Networks[b] and so on. These models represent only a small part of the variety of license plate recognition solutions and are popular with developers and users because of their efficiency, reliability and flexibility[6].

With the advent of new technologies, their scope of application is also expanding. Today it is difficult to find an area where ANPR systems are not used.

They have been applied in areas ranging from law enforcement and traffic management[b] to road safety and urban infrastructure[b].

It is important to note that ANPR systems continue to evolve and improve, thus increasing their potential. For example, with improved machine learning and deep learning algorithms, ANPR systems can provide more accurate and faster license plate recognition even under difficult lighting conditions or at high speed.

In addition, the integration of ANPR systems with other technologies such as artificial intelligence systems and the Internet of Things (IoT)[b] opens up new perspectives and opportunities for their application. This may include, for example, automatic real-time traffic light control based on traffic flow data, parking needs analysis and dynamic parking management.

One of the significant trends in the area of ANPR is the rapid development of artificial intelligence (AI) technologies. The integration of ANPR with AI is one of the key developments in this technology. In the past, ANPR systems used mainly classical image processing and optical character recognition algorithms for license plate analysis. However, with the development of deep learning and neural networks in recent decades, the ability to recognize and analyse images has expanded significantly.

The use of artificial intelligence in ANPR systems significantly improves the accuracy and efficiency of license plate recognition. Deep neural networks can automatically extract features from images and learn from large amounts of data to recognize complex images such as license plates in different lighting environments, with different backgrounds and perspectives. This allows for more reliable and fast recognition, even under difficult conditions, which greatly increases the efficiency of the system as a whole.

The integration of artificial intelligence also extends the functionality of ANPR systems. For example, AI can automatically analyze traffic data, detect anomalies or undesirable events such as traffic violations or security threats. Through continuous learning and adaptation, ANPR AI systems are able to respond effectively to changing conditions and requirements, making them more flexible and multifunctional tools for traffic management and road safety.

Despite all the advantages, however, ANPR faces serious challenges. Privacy and cybersecurity issues require increased attention and the development of robust data protection measures to prevent unauthorized access and leakage. High implementation and maintenance costs may limit their widespread adoption, especially in countries with limited resources. The legal and ethical aspects of mass surveillance require the development of transparent and harmonized international regulations that balance technological capabilities with citizens' rights.

The future of automatic license plate recognition therefore depends on the ability to adapt to new challenges and to ensure the safety and effectiveness of their solutions. The combined efforts of researchers, developers and legislators are necessary to create sustainable and ethical systems that will serve the public good by improving the quality of life and safety in our cities.

ANPR is an important and evolving technology that plays a key role in modern security and transport management systems. Through the application of advanced artificial intelligence techniques, ANPR provides greater accuracy and efficiency in number recognition under different conditions [7]. Trends in technology indicate an expansion into new applications and a desire for more adaptive and flexible systems. However, for ANPR to continue to develop successfully, attention needs to be paid to issues of data confidentiality, ethics and standardization to ensure a balanced and responsible approach to its implementation.

Chapter 3

Method and methodology

3.1 Dataset collection

One part of our research work is data collection. Data collection plays a key role in any research, especially when it comes to platform analysis. The importance of data collection can be considered from several perspectives, such as the objectivity of the analysis, the identification of trends and patterns, effectiveness assessment, planning and forecasting [8].

The first stage of the data collecting phase involves the exploration of data sources. In our case, this is a vehicle sales and rental platform in Central Asia. This phase is critical as the selection of correct and reliable data sources affects the accuracy and quality of subsequent analysis [9]. The process of data sourcing consists of steps such as the identification of key platforms, an analysis of data availability and an assessment of the reliability of sources. The process of identifying the main platforms involves examining well-known and widely used websites in Kazakhstan, Kyrgyzstan, Uzbekistan, Turkmenistan and Tajikistan. The accessibility analysis includes an assessment of the structure of the web pages, the availability of the required data and the ease of retrieval, and finally the reliability assessment of the sources consists of verifying the reliability and relevance of the data presented on each platform. This process helped eliminate unreliable sources and focus on those that provide accurate and relevant information.

Platform list:

- Kazakhstan: Wheels (kolesa.kz), Market.kz, OLX.kz
- Kyrgyzstan: Diesel.kg, AVTO.kg, Lalafo.kg
- Uzbekistan: Avtoelon.uz, OLX.uz
- Turkmenistan: Car.tm, TmCars
- Tajikistan: Somon.tj, Avto.tj

The second part of the data collection phase involves writing scripts to extract data from platforms. This phase is key to automating the collection of information and ensuring the accuracy and completeness of the data required for analysis. Before we started writing scripts, we did a thorough analysis of the structure of the web pages of the selected platforms. This included studying the HTML code of pages, identifying the necessary elements (e.g., headers, descriptions, prices, images) and their location on the page. To automate the data extraction process, we chose Python as the main programming language. The main libraries used in scripts include: BeautifulSoup: To parse the HTML code and extract the desired elements from the pages [10]. Requests: To perform HTTP queries and retrieve the contents of web pages. Regular Expression (re): For searching and processing specific text data within an HTML code

The extracted data was storesssd in JSON format. This format was chosen because of its convenience for storing structured data and the ease of subsequent processing and analysis, and because of its comprehensibility for people and machines. The JSON file structure made it easy to identify and extract the necessary fields for further work.

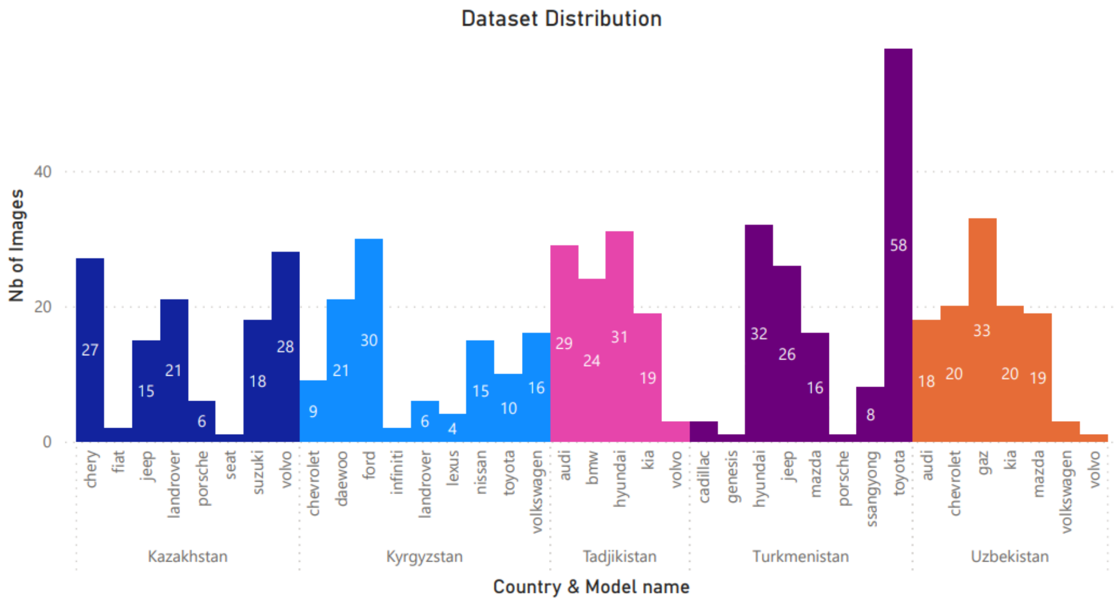

Figure 3.1 - Distribution of total images in different classes and countries

3.2 Data analyzing and Preparation

The data processing is the next crucial step. Efficient data processing improves data quality by enabling more accurate and meaningful analysis, improving analytical models and facilitating better decision-making. Without careful data processing, data analysis would be unreliable and potentially misleading. To produce accurate data, this phase involves several important steps [11].

3.2.1 Step 1: Remove Duplicates and Handle Missing Values

This initial step is crucial as it prevents distortions in the analysis results by addressing duplicate records and missing data points [12]:

- Delete Duplicates: Identify and remove duplicate records to ensure that each entry in the dataset is unique. This helps in maintaining data integrity and accuracy.

```
},
{
    "url": "/d/obyavlenie/prodam-mashinu-IDozyWD.html",
    "id": "dobyavlenieprodam-mashinu-IDozyWD.html",
    "brand": "suzuki",
    "title": "Продам машину",
    "Модель": " Vitara",
    "Тип кузова": " Внедорожник",
    "Год выпуска": " 1996 ",
    "Коробка передач": " Автоматическая",
    "Цвет": " Синий",
    "Вид топлива": " Дизель",
    "Состояние машины": " Хорошее",
    "Количество хозяев": " 1",
    "Доп. опции": " Эл. стеклоподъемники, Электрозеркала, Растаможена",
    "Пробег": " 5 000 км",
    "Объем двигателя": " 2 см³"
},
```


Figure 3.2 - Extracted data in JSON

- Handling missing values is crucial for several reasons. This ensures that data integrity is maintained. Missing values can distort statistical analysis and lead to incorrect conclusions. Proper processing ensures accurate reflection in the data set of the studied phenomenon. The next berth is improved model performance. In machine learning, models based on data with missing values may show low performance and low predictive accuracy. Eliminating missing values could improve the quality of models, leading to more reliable predictions. And the most obvious is the ease of data analysis. Many statistical methods and algorithms require complete datasets to function properly. The processing of missing values allows the use of these methods to facilitate a wider range of analyses.
  Referring to any missing or missing entries in the data set, depending on the nature of the data and the context of the analysis, the missing values may be regulated by:
  - Imputation: Fill in missing values using appropriate methods such as median, mean or mode to keep as much data as possible.
  - Delete: Delete records with missing values, especially if missing data are minimal and have no significant impact on the overall integrity of the data set.

Effectively managing duplicates and missing values ensures that the dataset is clean and ready for reliable analysis, leading to accurate and meaningful understanding.

3.2.2 Step 2: Standardization of Collected Data

The importance of standardizing values is that it ensures consistency and comparability across the data set, which increases the accuracy and reliability of analysis. By standardizing, we can decrease the variance caused by different units, formats, or scales, which aids in integrating data from multiple sources and facilitates the utilization of various analytical methods [13]. By using this process, bias can be minimized, statistical and machine learning models can be improved, and data processing can be simplified. In general, standardisation improves the quality of data, making it more understandable and useful for obtaining meaningful information and informed decisions. This phase involves standardizing the data to ensure consistency and uniformity across the dataset, facilitating easier analysis and processing.

- Standardized Image Format: Images may come in various formats depending on the source platform. To streamline subsequent analysis and image processing, all images will be converted to a single, uniform format—JPG. This conversion ensures compatibility and simplifies handling.
- Resizing Images: Along with format standardization, it is essential to standardize the size of the images. Every image will be resized to a uniform dimension of 964x964 pixels. This resizing ensures that all data is consistent and compatible for future use, particularly in applications like machine learning, where uniform image size is crucial for model training and evaluation.
- Image Labeling: Image labeling (or annotation) involves assigning descriptive tags to the images that highlight their content. For instance, tags could include "vehicle" or "license plate." Accurate labeling is vital for subsequent analysis and essential for machine learning tasks such as object classification or detection. It provides the necessary context and metadata that enhances the effectiveness and accuracy of automated analysis.

By standardizing image formats and sizes, and ensuring precise labeling, we create a cohesive and uniform dataset ready for detailed analysis and advanced machine learning applications [14].

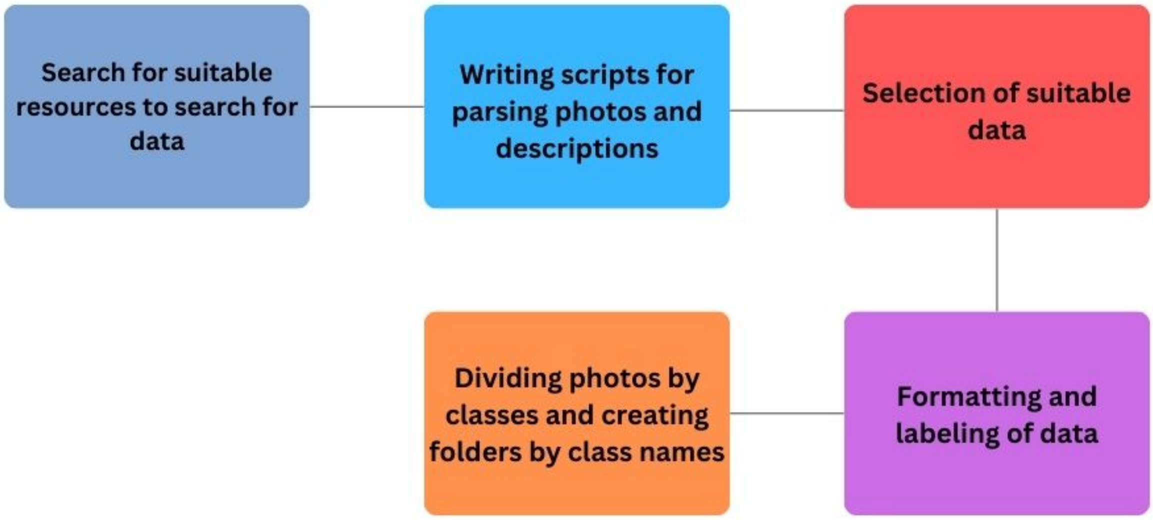

Figure 3.3 - Data collection scheme

3.3 Full description of data collection scheme

Comprehensive comparative analysis of machine learning algorithms for the detection of vehicle licence plates in difficult real world conditions requires a diverse and representative data set. It is important that these data reflect a wide range of conditions that may affect the accuracy and reliability of algorithms. In particular, data from Central Asian countries are of unique interest because of their specific characteristics, such as the variety of licence plates, lighting conditions, weather and road conditions. This section describes how to search for and collect such data.

3.3.1 Data retrieval methods

Scientific articles and conferences
Scientific articles and conferences are a valuable source of information on current trends and best practices in license plate detection. The study of such resources provides insight into the latest developments and methodologies that can be applied to the study [15]. Resources: Google Scholar, IEEE Xplore, Springer, ACM Digital Library. "license plate detection", "vehicle number plate recognition", "challenging environments", "machine learning", "Central Asia".

Public datasets
Public datasets provide ready-to-use images that can be useful for algorithm learning and testing. These data are often annotated and structured to facilitate their integration into the research process.

- Resources: Kaggle, UCI Machine Learning Repository, GitHub.
- Examples of datasets: OpenALPR Benchmark, MVTec AD, Car License Plates.

Auto markets in Central Asian countries
Car markets provide access to live images of cars with license plates, which is particularly valuable for this study. Data from such sources may include different survey conditions and license plate characteristics. Countries: Kazakhstan, Uzbekistan, Turkmenistan, Kyrgyzstan, Tajikistan.

Examples of sales markets:

- Kazakhstan: Kolesa.kz, Olx.kz
- Uzbekistan: Avtoelon.uz, RST.uz
- Turkmenistan: Avto.tm
- Kyrgyzstan: Avto.kg
- Tajikistan: Somon.tj

3.3.2 Reasons for choosing data from Central Asia

1. **Different kinds of shooting conditions**
   Regional features such as different shooting angles, variety of lighting conditions (bright sun, fog, dusty roads) and weather conditions (snow, rain), provide unique challenges for machine learning algorithms, which contributes to their more effective learning and testing.

2. **Differences in licence plates**
   License plates in Central Asian countries may differ in format, color, font and structure. This allows for a more versatile model that can handle a variety of input data.

3. **Cultural and economic factors**
   Cars in this region may have specific features, such as a large number of imported cars with different numbers, adding variability to the dataset.

4. **Insufficient research data**
   Data from this region may be less studied than data from Europe or North America. This makes it possible to bring novelty and uniqueness to the study, as well as to increase its academic and practical value.

3.3.3 Developing scripts for parsing photos and metadata

Data collection and pre-processing are key steps in the vehicle license plate detection study. To obtain relevant data from web resources such as car marketplaces, it is necessary to develop scripts for automated image parsing and associated metadata. This section describes the methods and tools used to create such scripts.

Data parsing techniques:

- HTML parsing is a type of data parsing technique that is used to extract information from HTML documents. HTML is a markup language used to create web pages. HTML parsing includes splitting a raw HTML document into separate tags and attributes to extract relevant information.

- XML parsing is a type of data parsing technique that is used to extract information from XML documents

- JSON analysis is similar to XML analysis, but is used to extract information from JSON documents [16].

```
model_name = 'seat'
model_url = 'https://www.olx.kz/transport/legkovye-avtomobili/seat/'

r = requests.get(model_url)
htmldata = r.text
soup = BeautifulSoup(htmldata, 'html.parser')
print(soup.find_all("a", {"class": "css-z3gu2d"}))
linkList = []
json_data_array = []
print(os.path.dirname(__file__))

for index, item in enumerate(soup.find_all("a", {"class": "css-z3gu2d"})):
    print(f"{index})", f"\033[96m{item['href']}\033[00m")
    linkList.append(item['href'])
```

Figure 3.4 - Example of parsing code

Data processing and storage

1. Saving Images

2. Extracted images must be saved to the local file system or cloud storage for further processing and anonymity.

```python
with open("kz.json", "r+", encoding='utf8') as json_file:
    file_data = json.load(json_file)
    file_data['data'].extend(json_data_array)
    json_file.seek(0)
    json.dump(file_data, json_file, indent = 4, ensure_ascii=False)
```

Figure 3.5 - Example of data storage code

This small piece of Python code adds the contents of the JSON file kz.json to the dictionary using the UTF-8 encoding. Then, fresh entries from json_data_are added to the list associated with the data key in this dictionary. To ensure that the file is successfully overwritten, the code resets the beginning of the file after the dictionary update. Finally, the code format the output with four indents for readability and provides for saving characters without ASCII before writing the revised dictionary back to the JSON file. This procedure stores the modified information and adds new data to an already existing list, thereby updating the JSON file.

Metadata collection

It is also important to collect related metadata (e.g., description of the car, model, year of manufacture,brand, title, id, url etc.) that can be useful for analysis and classification [17].

This fragment of Python code goes through the URL list (linkList), makes HTTP GET-requests for each URL using the request module, and analyzes HTML content, via BeautifulSoup for specific information. It advertises json_data dictionary to store the received data, including the URL, purified identifier, brand and product name. For each image found with this class, it loads the image, clears the file name, and stores it locally.

```
for item in linkList:
    pr = requests.get(f'https://www.olx.kz{item}')
    prHtml = pr.text
    prSoup = BeautifulSoup(prHtml, 'html.parser')
    json_data = {}
    title = prSoup.find("h4", {"class": "css-1juynto"})
    json_data['url'] = item
    json_data['id'] = re.sub('[\!@#$/]', '', item)
    json_data['brand'] = model_name
    json_data['title'] = title.text if title.text else ''

    for index, item in enumerate(prSoup.find_all("img", {"class": "css-1bmvjcs"})):
        file_name = re.sub('[\!@#$/]', '', json_data['id']) + f" ({index})"
        # re.sub('[!@#$]', '', file_name)
        with open(file_name, 'wb') as f:
            f.write(requests.get(item['src']).content)

    for index, item in enumerate(prSoup.find_all("p", {"class": "css-b5m1rv er34gjf0"})):
        data = item.text.split(':')
        if len(data) != 2:
            continue
        key = data[0]
        value = data[1]

        json_data[key] = value

    json_data_array.append(json_data)


    print(prSoup.find_all("p", {"class": "css-b5m1rv er34gjf0"}))
```

Figure 3.6 - Example of metadata collection code

In addition, the user extracts key-value pairs from the text repeatedly using certain paragraph tags and stores them in the JSON data dictionary. To use or process later, it adds each json_data dictionary to the list (json_data_array). This code effectively uploads photos, cleans web pages, analyzes data and collects information in an organized manner.

3.3.4 General information about metadata

Information regarding the list of vehicles is contained within these metadata. Here is a breakdown for each field:

- url: The URL path for the vehicle listing.
- id: A unique identifier for the vehicle listing.
- brand: The brand or manufacturer of the vehicle (in this case, "infiniti").
- title: The title or headline of the vehicle listing, which likely includes additional details such as the model year, color, and payment method.
- Year: The year of manufacture of the vehicle (in this case, "2010").
- Color: The color of the vehicle.
- Model: The model of the vehicle (in this case, "QX56").
- Payment: The method of payment for the vehicle.
- Fuel: The type of fuel used by the vehicle.
- Customs clearance: The customs clearance status of the vehicle.
- Technical condition: The technical condition of the vehicle.
  In general, these metadata offer comprehensive information on a specific list of vehicles, such as the make, model, year, color, payment method, fuel type, customs status, and technical condition.

3.3.5 Selection of suitable data

After the collection of data from various sources such as websites, APIs, and public datasets, a thorough selection should be done. The purpose of this step is to ensure that the data is of high quality and relevant for future machine learning algorithm training and testing. This section describes methods of data sampling, including image pre-processing and filtering.

Data sampling methods

1. **Image preprocessing**
   Image preprocessing: Low-quality images must be excluded since they might have a detrimental impact on model accuracy. These comprises photos that are poor quality, blurry, or corrupted.
2. **Data completeness analysis**
   Ensure that the required information for each image is available (e.g., car description, model, year of manufacture). Missing or partial data may make model analysis and training challenging.
3. **Manual check**
   Visible image verification: In certain cases, automated filtering may be inadequate. A human review of a small sample of the data will assist to ensure its quality and relevance.
   Annotation verification: If the data includes annotations, it is required to ensure their accuracy and completeness. Improper annotations have a major impact on model learning and assessment [18].

3.3.6 Split images by classes and create folders for each class

When the data has been prepared and tagged, the images are organized into a directory structure, with each folder representing a certain class or category. Sorting images into classes and establishing folders for each is a key step in preparing data for machine learning models. This creates the handy data format required for efficient model learning and testing.

Figure 3.7 - Splitted data by class

Figure 3.8 - Example of collected data

3.4 Comparison with other datasets

In the field of vehicle and license plate identification, the presence and use of various datasets are essential for creating strong machine learning models. This research examines many prominent datasets, each presenting distinct attributes and difficulties that contribute to progress in this subject.

Vehicle-1M is a comprehensive dataset consisting of around 1 million photos depicting 55,527 cars. The photographs were obtained from security cameras put in numerous cities, capturing both daytime and nighttime scenes from various perspectives (front and back). This dataset offers an extensive and all-encompassing source for analyzing and investigating the identification and detection of vehicles in diverse metropolitan environments [19].

CompCars specifically targets a collection of 136,727 photos of Chinese model automobiles that have been obtained from security cameras. This dataset focuses on automobiles that are often seen in China, providing significant information for tasks related to detecting and recognizing vehicles peculiar to the area [20].

AOLP is a focused dataset specifically designed for the purpose of license plate detection. The samples were obtained in Taiwan from various places and under varying circumstances, including different times and vehicle movements. The collection is divided into three separate categories, offering a valuable source for license plate identification and examination [21].

The Dataset was obtained from Central Asian nations including Kazakhstan, Kyrgyzstan, Uzbekistan, Turkmenistan, and Tajikistan. The collection comprises photographs captured at different points in time and in diverse circumstances, accurately representing real-life situations with varied lighting conditions. This dataset focuses on the distinct difficulties associated with identifying and recognizing vehicles and license plates in Central Asia, therefore enhancing our comprehensive knowledge of these specific tasks on a worldwide scale.

Through the comparison of various datasets, our objective is to emphasize their individual strengths and uses, finally providing guidance for the enhancement of vehicle and license plate recognition systems that are more efficient and versatile.

3.5 Data and discussion

After extracting and processing the data, we obtained a clean and standardized data set. This dataset is unique because it has a high degree of accuracy and integrity, making it a solid basis for further analysis. Due to careful processing, data is now free of duplicates and passes, standardized by format and size, which simplifies their use in various analytical tasks and machine learning models. This qualitative data set allows for deeper and more reliable analysis leading to more informed and accurate conclusions.

The dataset includes information on vehicles and licence plates of Central Asian countries, showing unique characteristics for vehicle categories, licence plate types and regional variations. This comprehensive data set is invaluable to local researchers in machine learning and computer vision, enabling them to develop and evaluate techniques for automated license plate recognition. It also serves researchers specializing in vehicle recognition and segmentation, especially those developing traffic management applications in Central Asia. The images in the dataset collected under different conditions and at different times have different illumination and quality, reflecting realistic scenarios and increasing the practical relevance and applicability of the dataset.

Chapter 4

Methods of ML

4.1 Overview

To detect license plates in our photos, we have used a number of advanced machine learning and computer vision techniques, particularly using the YOLO family of models (You Only Look Once) [22]:

- YOLO v5
- YOLO v7
- YOLO v8

These models are well-known for their capacity for real-time object detection, offering the speed and accuracy compromise required to locate and identify license plates in various conditions.

4.2 Score metrics

For our machine-learning algorithms, we use the following scoring metrics:

- F1-score
- Accuracy
- Runtime

F1-score The harmonic mean of precision and recall, providing a balanced assessment of a model's performance It combines two metrics - precision and recall to get one overall score. [link] . Precision and recall are two good evaluation parameters, which are used for the assessment of efficiency of the classification models [23]

$$Precision = \frac{TP}{TP + FP} \tag{4.2.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.2.2}$$

$$\text{mAP} = \frac{1}{n} \sum_{i=1}^{n} \text{AP}_i \tag{4.2.3}$$

Precision, also called a positive predictive value, determines the accuracy of the model's positive predictions. It is calculated as the ratio of the number of true positive predictions (correctly identified positive cases) to the total number of positive predictions (both true and false).

Recall, also known as sensitivity or true positive rate, measures the ability of the model to correctly identify all relevant positive instances. The ratio of true positive predictions to the total number of actual positives (both true positives and false negatives) is what it is

In many applications, there is a trade-off between accuracy and revocation. Improving one often results in reducing the other. The balance between accuracy and recall can be adjusted according to the specific requirements of the tasks.

Mean Average Precision, or mAP, is one of the most effective comprehensive metrics, which can be applied to the evaluation of the object detection and multi-class classification algorithms. [link] It then takes the mean of all Average Precision (AP), where AP is the measure of the area under the curve between precision and the recall rates for a specific class. Thus, mAP integrates these scores, and gives a single number which represents the overall performance of a method at detecting and localizing objects of all classes possible. The most important reason to use this metric is when the user values accuracy but also needs for recall , and this is why it is often utilized in evaluation of models in computer vision challenges like COCO and PASCAL VOC benchmarks .

Steps to Calculate mAP

- Compute Precision and Recall at Different Thresholds. For each class, compute precision and recall at various thresholds of detection confidence or ranking.
- Plot Precision-Recall Curve. Plot the precision-recall curve for each class by varying the detection threshold.
- Calculate Average Precision (AP). The Average Precision for each class is the area under the precision-recall curve. This can be approximated using numerical integration methods like the trapezoidal rule.
- Compute mAP. mAP is the mean of the AP scores across all classes. If there are n classes, mAP is given by:

Runtime is definitely an essential factor because it determines the performance and optimality of the entire system. The intuition behind this is that a lower role of any operation or algorithm is also an indication that the process is done more quickly, which may be especially much desired by users in the case of restricted Internet services and / or applications where an instantaneous response is required.

This gains the proposal several advantages in increasing performance, which are brought by reducing the outer loop's runtime. Secondly, it helps you to decrease the amount of load on your server and, as a result, decrease the maintenance costs since the efficient execution of operations will need fewer server resources. This can be especially important when working with large data sets or applications are loaded, for example, in the form of web services.

Low runtime increases user satisfaction as they are quicker to get changes in operations or requests. This increases the overall efficiency of users and improves the interaction with the system.

Different methods can be used to estimate the running time, including measuring the real-time performance of an operation or algorithm on specific data, or using asymptotic complexity to estimate the performance of the algorithm depending on the size of the input data.

Runtime estimation and performance optimization are important tasks in various fields, such as software development, computing, databases, car training and others. The efficient use of resources and the reduction of lead times are key to improving the system's efficiency and meeting user needs.

4.3 YOLO (You Only Look Once)

The YOLO model (You Only Look Once) is one of the popular algorithms for detecting objects in computer vision. YOLO has several versions, with improved accuracy and speed over time, and it is widely used in various areas including autonomous vehicles, surveillance systems, and augmented reality.

The working principle of Yolo consists of several stages. YOLO takes the input image, which you need to analyze and define the objects on it.The Image is then divided into a grid of cells. Usually used are grids of size such as 7x7 or 13x13. For each split grid cell, YOLO makes predictions about the presence of an object and its attributes. These predictions include the coordinates of a bounding rectangle (bounding box) around an object, the probability of having an object in a cell, and the probabilities of object classes. After obtaining predictions for each cell, YOLO adjusts them to the context of adjacent cells and applies thresholds to reduce false positives. YOLO then uses an algorithm called "non-max suppression" to remove duplicate predictions and select the most likely objects. Eventually, YOLO returns discovered objects together with their classes and coordinates of bounding rectangles.

In the current experiment, I chose the YOLO models versions 5, 7 and 8. These models are characterized by improvements in precision and object detection speed. YOLOv5, developed by Ultralytics, focuses on ease of use. YOLOv7 offers improved performance on various hardware platforms, and YOLOv8 includes recent advances in machine learning and integration with modern tools. Using the obtained data, I trained these models and compared their results in the following parameters: F1-score, Accuracy (mAP @ 0.5 IOU) and Runtime (runtime).

4.3.1 Result of Experiment

Table 4.1 - Yolo model result

Metric	**Yolo 3**	**Yolo 5**	**Yolo 7**
F1-score	0.609 Col 2	0.482	0.0609
Accuracy	0.932	0.967	0.975
Runtime	0.470 hours	0.365 hours	0.069 hours

Evaluation of the effectiveness of YOLO object detection models: YOLOv5, YOLOv7 and YOLOv8

The detection of the license plate object has made significant progress with the development of various models within YOLO (You Only Look Once). It consists of YOLOv5, YOLOv7 and YOLOv8 stand out as outstanding versions, each of which brings its own set of improvements and optimizations. This exchange provides a comparative analysis of these three models based on the key performance indicators: F1-score, Accuracy and Runtime.

F1-Score

F1-score, an important measure for assessing the balance between accuracy and repetition, emphasizes the model's reliability in handling false positives and false negatives. According to their results, they:

YOLOv5 achieved an F1-score of 0.609, demonstrating a strong balance and indicating a reliable performance in object detection tasks. YOLOv7 reported an F1-score of 0.482, which, while still respectable, suggests a slightly lower performance in balancing precision and recall compared to YOLOv5. YOLOv8 showed a significantly lower F1-score of 0.0609, This indicates difficulties in maintaining an optimal balance between accuracy and conservation. This can be caused by various factors, such as the specificity of the model's learning or the change in the data set.

Accuracy

The accuracy that reflects the proportion of correctly identified cases among all cases is a direct measure of the overall effectiveness of the model. The results are as follows:

YOLOv5 recorded an accuracy of 0.932, demonstrating its capability to accurately identify objects in a given dataset. YOLOv7 outperformed YOLOv5 with an accuracy of 0.967, showcasing improvements and refinements in the model's architecture and training process. YOLOv8 topped the chart with an accuracy of 0.975, indicating the highest reliability among the three models in terms of correctly identifying objects.

Runtime

Runtime is one of the critical metrics that reflects the efficiency and speed of a model, especially important for real-time applications. The comparative analysis reveals:

YOLOv5 had a runtime of 0.470 hours, making it relatively time-efficient but leaving room for improvement. YOLOv7 showed an enhanced runtime of 0.365 hours, indicating better optimization and faster processing capabilities. YOLOv8 significantly reduced the runtime to 0.069 hours, demonstrating exceptional efficiency and making it highly suitable for applications requiring real-time object detection.

Conclusion

The comparative analysis of YOLOv5, YOLOv7 and YOLOv8 based on F1-score, accuracy and execution time provides valuable information about their respective strengths and improvement directions.

YOLOv5 stands out for its balanced F1-score, indicating a reliable performance in managing false positives and negatives. YOLOv7 offers a middle ground with improved accuracy and runtime over YOLOv5, but with a slight compromise on the F1-score. YOLOv8 excels in accuracy and runtime, making it highly efficient for real-time applications, though its lower F1-score suggests potential issues in precision-recall balance.

These findings suggest that the choice of model should be tailored to specific application needs, considering the trade-offs between accuracy, runtime efficiency, and the balance of precision and recall.

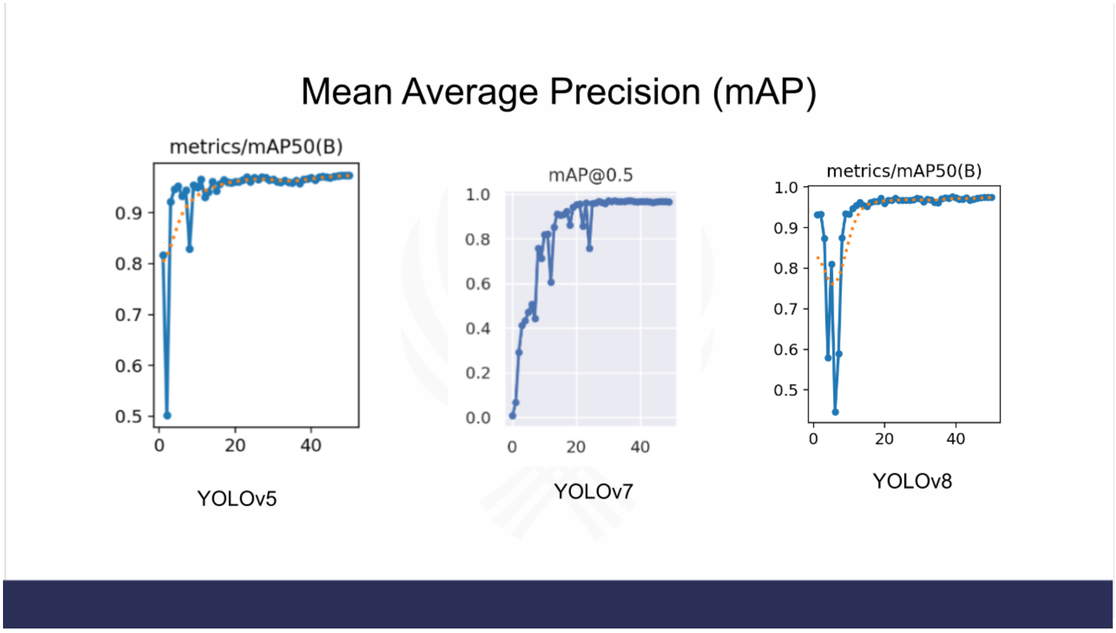

Figure 4.1 - Mean Average Precision graph

This image is a comparison of accuracy and recall scores for three versions of the YOLO object detection model: YOLOv5, YOLOv7 and YOLOv8. Each subgraph illustrates the performance of models over a range of 50 epochs. YOLOv5 and YOLOv8 have separate graphics for accuracy and recall, while YOLOv7 shows a combined graph. Graphs show how the accuracy of each model and revocation values improve and stabilize over time, providing an understanding of their accuracy and detection reliability.

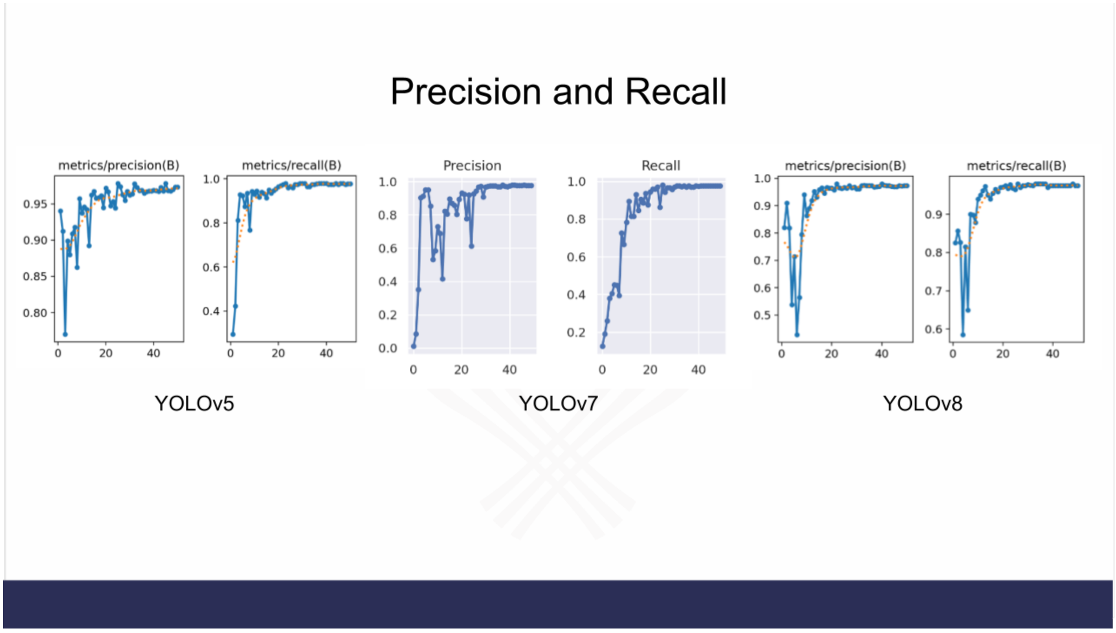

Figure 4.2 - Precision and Recall result graph

4.3.2 Description of the experiment process

Step 1. The first step is to install and import the required components and packages.

```
!mkdir {HOME}/datasets
%cd {HOME}/datasets
!pip install roboflow

from roboflow import Roboflow
from IPython.display import display, Image
```

Figure 4.3 - Import necessary packages

- Importing the os module: The os module provides functions for interaction with the operating system. It allows you to work with the file system, receive information about the current working directory, create, delete and modify files and directories, and perform other system operations.
- Getting the current working directory: HOME = os.getcwd() uses the.getcwd() function to get the current working directory path. The result is saved in the HOME variable. This can be useful for retrieving the path to files and directories relative to the current working directory.
- Installing the ultralytics package: The pip install ultralytics command is used to install the Python ultralytics package via the pip package manager. The ultralytics package contains tools and libraries for computer vision, particularly for working with the YOLO (You Only Look Once) models for object detection.

```
loading Roboflow workspace...
loading Roboflow project...
```

```
[ ] rf = Roboflow(api_key="[redacted])
    project = rf.workspace("label-a6heb").project("alll-vquhn")
    version = project.version(1)
    dataset = version.download("yolov5")
```

```
%cd {HOME}
```

Figure 4.4 - Robflow data extracting code

Step 2. In the second step, the code demonstrates the process of downloading the YOLOv5 dataset from a project on the Roboflow platform using the Roboflow library

- Creates a Roboflow object using the API key. The API key authenticates the user and provides access to his projects and data on the Roboflow platform.
- The code selects a specific workspace and project. In this case, the workspace with the identifier "label-a6heb" and the project with the identifier "alll-vquhn" are used. This allows access to the desired project for further operations.
- Next, you select a specific version of the project. This is version number 1. Each version of the project can store different versions of the data and annotations.
- The YOLOv5 dataset is loaded. This call downloads all the necessary files and annotations to use the YOLOv5 model in the object detection problem.

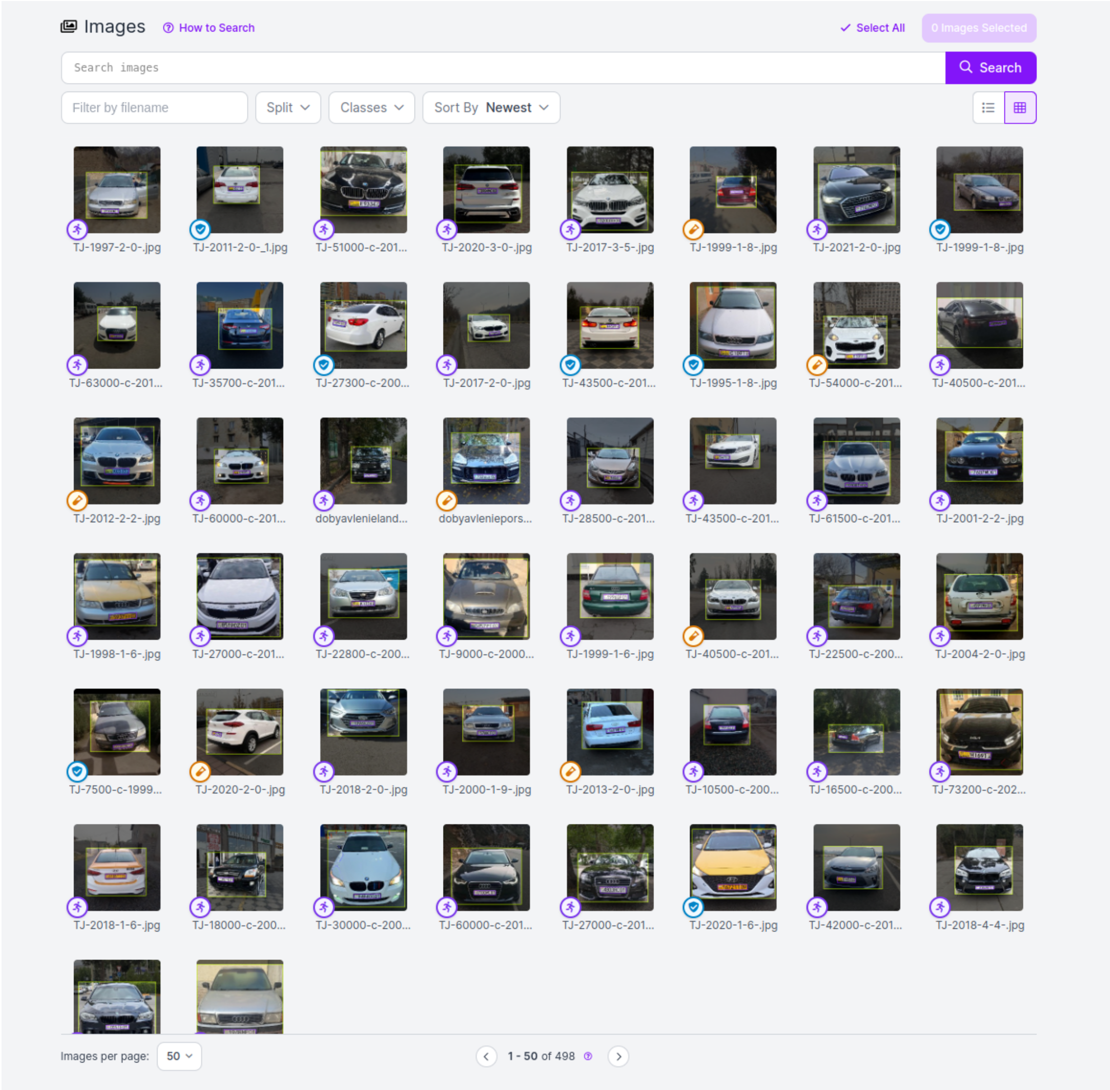

Figure 4.5 - Dataset in Roboflow

Step 3. The third step of the code demonstrates the process of starting the learning model YOLOv5 for the object detection task using the previously loaded dataset.

Learning process of the YOLOv5 model using the following set of parameters:

- task=detect: Specifies that the task is to detect objects.
- mode=train: Sets the operating mode in "train" (training).
- model=yolov5nu.pt: Specifies the path to a pre-trained model to be used as a starting point for further learning.
- data=dataset.location/data.yaml: Specifies the path to the dataset configuration file that was previously loaded. This file contains information about the dataset structure and data paths.
- epochs=50: Sets the number of epochs (iterations) for the learning process. In this case, the model will be trained for 50 epochs.

```
optimizer: AdamW(lr=0.001667, momentum=0.9) with parameter groups 69 weight(decay=0.0), 76 weight(decay=0.0005), 75 bias(decay=0.0)
TensorBoard: model graph visualization added ✅
Image sizes 640 train, 640 val
Using 8 dataloader workers
Logging results to runs/detect/train3
Starting training for 50 epochs...

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       1/50      2.57G     0.9103        2.2      1.249         54        640: 100% 22/22 [00:05<00:00,  3.97it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 4/4 [00:02<00:00,  1.96it/s]
                   all        100        200       0.94      0.295      0.816      0.564

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       2/50      2.32G     0.8664      1.251      1.185         46        640: 100% 22/22 [00:02<00:00,  9.78it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 4/4 [00:00<00:00,  6.45it/s]
                   all        100        200      0.912      0.422      0.503      0.378

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       3/50      2.32G     0.8627      1.111      1.171         44        640: 100% 22/22 [00:02<00:00,  9.54it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 4/4 [00:00<00:00,  6.25it/s]
                   all        100        200       0.77      0.812      0.921      0.674

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       4/50      2.29G     0.8858      1.075      1.202         45        640: 100% 22/22 [00:02<00:00, 10.08it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 4/4 [00:00<00:00,  7.31it/s]
                   all        100        200      0.899      0.929      0.946      0.631

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       5/50      2.29G     0.8964      1.044      1.196         58        640: 100% 22/22 [00:02<00:00, 10.21it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 4/4 [00:00<00:00,  7.62it/s]
                   all        100        200       0.88      0.923      0.952      0.626

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       6/50      2.29G     0.9253      1.041      1.246         65        640: 100% 22/22 [00:02<00:00, 10.14it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 4/4 [00:00<00:00,  7.55it/s]
                   all        100        200      0.909      0.875      0.933      0.665

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       7/50      2.29G     0.8691     0.9241      1.176         50        640: 100% 22/22 [00:02<00:00, 10.20it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 4/4 [00:00<00:00,  7.85it/s]
                   all        100        200      0.918      0.935      0.944      0.717

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       8/50      2.29G     0.8492     0.8977      1.181         67        640: 100% 22/22 [00:02<00:00, 10.37it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 4/4 [00:00<00:00,  8.17it/s]
                   all        100        200      0.862      0.768      0.829      0.577
```

Figure 4.6 - Model training process

Step 4. The final step will be to create graphs that demonstrate the results.

The result is shown in the form of graphs. Graphs are constructed using blue and red lines. The blue lines show the variable values, and the red lines show the error values.

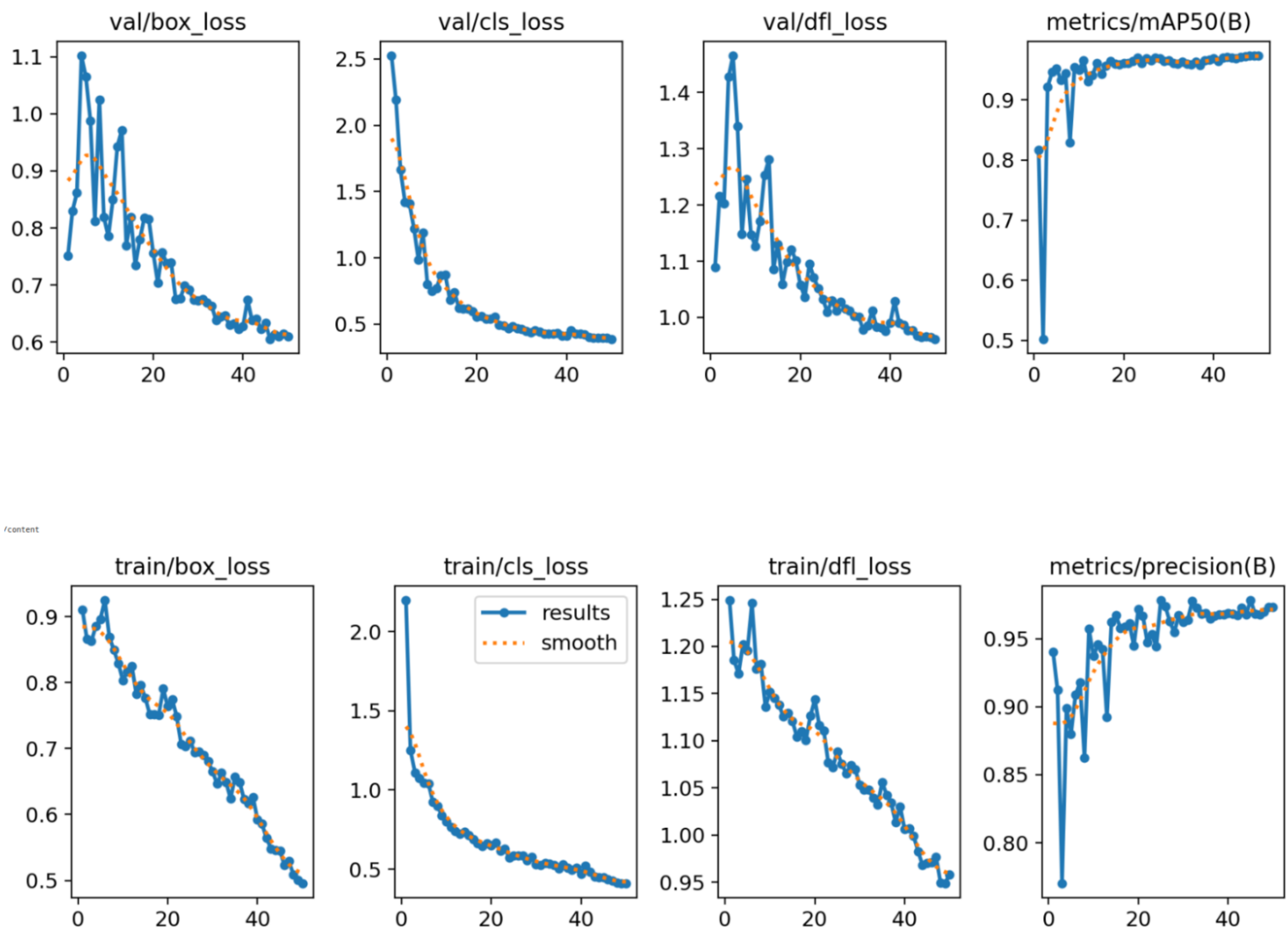

Figure 4.7 - Result graph

The following graphs are shown:

val-box-loss: This graph shows the average value of box loss (box loss) on the validation dataset. Crate losses are a metric that measures how well the model predicts the boundaries of objects in images.

val-cls-loss: This graph shows the average value of loss classes (classification loss) on the validation dataset. Class loss is a metric that measures how well the model predicts classes of objects in images.

metrics/mAP50(B): This graph shows the mean value of the mean precision mean (mean average precision) at a threshold of 50% on the validation dataset. The mean exact definition is a metric that measures how well the model predicts objects in images.

metrics/mAP50-95(B): This graph shows the mean value of the mean accurate determination at a threshold of 50% to 95% on the validation dataset. The mean exact definition is a metric that measures how well the model predicts objects in images.

Metric

The lower part of the image shows the values of different metrics. These metrics include:

- **epoch**: This metric shows how many times the model has been trained on the whole data set.
- **batch**: This metric shows how many images were processed simultaneously during training.
- **img_per_batch**: This metric shows how many objects were on each image.
- **lr**: This metric shows the learning speed with which the model was trained.
- **time**: This metric shows how long it took to learn the model.

Chapter 5

Conclusions and future work

5.1 Conclusions

Licence plate object identification has advanced significantly with the YOLO (You Only Look Once) object detection models YOLOv5, YOLOv7, and YOLOv8. We assess these models using important performance metrics: runtime, accuracy, and F1 score

The F1-score, accuracy, and runtime of three YOLO object identification models—YOLOv5, YOLOv7, and YOLOv8—are compared and evaluated in this paper. With an F1-score of 0.609, YOLOv5 exhibits a good balance; YOLOv7 trails somewhat with an F1-score of 0.482, while YOLOv8 suffers with a much lower F1-score of 0.0609, suggesting difficulties in preserving accuracy. At 0.975, YOLOv8 scores highest in accuracy, closely followed by YOLOv7 at 0.967 and YOLOv5 at 0.932. Notably efficient runtime is YOLOv8, which comes in first, then YOLOv7, and finally YOLOv5. All things considered, the model selected will rely on particular application needs after weighing the trade-offs between precision and recall, runtime efficiency, and accuracy.

These results imply that, taking into account the trade-offs between accuracy, runtime efficiency, and the balance of precision and recall, the model selection should be adapted to particular application requirements.

5.2 Future work

Future research in the field of number plate identification in challenging real-world situations might prioritize many crucial areas, including enhancing the dependability of detection algorithms under severe illumination conditions. Challenging weather conditions and obstructions continue to pose substantial difficulties. However, by using modern artificial intelligence approaches like deep reinforcement learning and transformer models, significant boosts in performance may be achieved.

Furthermore, it is crucial to optimize the algorithms used for real-time processing on peripheral devices in order to effectively use them in smart cities and autonomous cars. Utilizing cross-domain adaptation and knowledge transfer may facilitate the development of models that exhibit strong performance across diverse contexts. Investigating the feasibility of integrating multimodal sensors by mixing optical data with radar or LIDAR data has the potential to enhance the precision and dependability of detection.

As these technologies grow more widespread, ethical and secrecy concerns become more important. A crucial area of investigation will be guaranteeing the confidentiality of data and adhering to legal norms, all while maintaining a high level of detection efficiency. By using these skills, future research may greatly enhance and enhance the dependability of vehicle license plate identification systems in real-world scenarios.

We are also thinking of expanding our current data set to meet other research goals. This includes collecting more photos that cover different types of cars and license plates in addition to numerous geographic areas and lighting conditions. More data would help to develop a model and would be a more effective method of identifying and detecting objects in a number of realistic scenarios. Other tests are also planned using different approaches to increase and prevent the volume of data. In addition, multiple algorithms and models will be used to improve the accuracy and efficiency of licence plates and vehicle identification.

Plans for the future also include for testing with alternative object identification models such RetinaNet [24], CenterNet [25], EfficientDet [26], and so on. The purpose of these tests is to evaluate their performance in various weather, lighting, and license plate designs scenarios. In order to increase the usefulness of models in actual situations, it is intended to carry out comparative study on important metrics like accuracy (Accuracy), F1-Score, and lead time (Runtime). These research will contribute to the identification of the optimal methods for rapid and accurate traffic management and video surveillance jobs.

Bibliography

[1] Asif Ali Laghari Aabdullah Ayub Khan. Machine learning in computer vision. *ResearchGate*, 1(1):1–89, 04 April 2021.

[2] Kamil Dimililer. Vehicle detection and tracking using machine learning techniques. *10th International Conference on Theory and Application of Soft Computing, Computing with Words and Perceptions - ICSCCW-2019*, 6(1):373–381, 01 2020.

[3] Al-Zoabi A. Ali S. Ahmad, F. Automatic number plate recognition (anpr): A comparative analysis of machine learning algorithms. *Artificial Intelligence Research*, 6(1):1–63, 11 2021.

[4] N.-Shah S.A.A. Lubna, Mufti. Automatic number plate recognition:a detailed survey of relevant algorithms. *Sensors 2021*, 21(9):1–63, 11 2021.

[5] The history of anpr. *ANPR-international*.

[6] A; Affendi H R; Aidil A M. S Fakhar A G; Saad H, M; Fauzan K. Development of portable automatic number plate recognition (anpr) system on raspberry pi. *International Journal of Electrical and Computer Engineering*, 9(3):1805–1813, 06 June 2019.

[7] Asif Siddiq Muhammad Murtaza Khan Muhammad Usman Ilyas Saleh Alshomrani Ishtiaq Rasool Khan, Syed Talha Abid Ali. Automatic license plate recognition in real-world traffic videos captured in unconstrained environment by a mobile camera. *Electronics 2022*, 11(9), 04 April 2022.

[8] Kim A. Hoffman Bojana Lobe, David Morgan. Qualitative data collection in an era of social distancing. *International Journal of Qualitative Methods*, 11(9), 07 July 2020.

[9] Syeda Ayeman Mazhar. Methods of data collection: A fundamental tool of research. *Journal of Integrated Community Health*, 10(1), 06 June 2021.

[10] Ayat Abodayeh; Reem Hejazi; Ward Najjar; Leena Shihadeh; Rabia Latif. Web scraping for data analytics: A beautifulsoup implementation. *IEEE*, 10(1), 06 June 2021.

[11] Pieter Colpaert Brecht Van de Vyvere. Using anpr data to create an anonymized linked open dataset on urban bustle. *Springer Nature*, 14(17), 04 April 2022.

[12] Iskandar Ishak Hamidah Ibrahim Mustafa Alabadla, Fatimah Sidi. Systematic review of using machine learning in imputing missing values. *IEEE*, 14(17), 04 April 2022.

[13] Standardization in data management: Simplifying information exchange. *fastercapital.*

[14] Unlocking the power of data with data standardization. *intone.*

[15] Yongjun Zhang Yansheng Li, Jiayi Ma. Image retrieval from remote sensing big data: A survey. *Information Fusion*, 67(17), 03 March 2021.

[16] Alexey Golubev Danila Parygin Tatiana Smykovskaya Vyacheslav Cherkesov, Vitaliy Malikov. Parsing of data on real estate objects from network resource. *Proceedings of the IV International research conference "Information technologies in Science, Management, Social sphere and Medicine*, 67(17), 12 December 2017.

[17] M.C. Andrews. Metadata basics for web content. *Association for Information Science and Technology*, 2020.

[18] Mohammad Sultan Mahmud; Joshua Zhexue Huang; Salman Salloum; Tamer Z. Emara; Kuanishbay Sadatdiynov. A survey of data partitioning and sampling methods to support big data analysis. *TUP*, 3(2):1–89, 02 February 2020.

[19] Zhao C. Liu Z. Wang J. Lu H. Guo, H. Learning coarse-to-fine structured feature embedding for vehicle re-identification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), 2018.

[20] H. Frigui F. Tafazzoli and K. Nishiyama. A large and diverse dataset for improved vehicle make and model recognition. *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, USA,*, pages 874–881, 2017.

[21] J. C. Chen G. S. Hsu and Y. Z. Chung. Application-oriented license plate recognition. *IEEE Transactions on Vehicular Technology*, 62(2):552–561, 02 February 2013.

[22] ZHANG Duo SHAO Yanhua. A review of yolo object detection based on deep learning. *Journal of Electronics Information Technology*, 42(10), 2022.

[23] Jitendra V. Tembhurne Tausif Diwan, G. Anirudh. Object detection using yolo: challenges, architectural successors, datasets and applications. *Springer*, 82(2):243–275, 08 August 2022.

[24] Roberto Del Prete, Maria Daniela Graziano, and Alfredo Renga. Retinanet: A deep learning architecture to achieve a robust wake detector in sar images. In *2021 IEEE 6th International Forum on Research and Technology for Society and Industry (RTSI)*, pages 171–176, 2021.

[25] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *2019*

IEEE/CVF International Conference on Computer Vision (ICCV), pages 6568–6577, 2019.

[26] Mingxing Tan, Ruoming Pang, and Quoc Le. Efficientdet: Scalable and efficient object detection, 11 2019.