UNIVERSITY OF ZAGREB
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

MASTER THESIS No. 275

VESSEL REGISTRATION PLATE DETECTION USING A MONOCULAR CAMERA

Maja Magdalenić

Zagreb, February 2024

UNIVERSITY OF ZAGREB
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

MASTER THESIS No. 275

VESSEL REGISTRATION PLATE DETECTION USING A MONOCULAR CAMERA

Maja Magdalenić

Zagreb, February 2024

UNIVERSITY OF ZAGREB
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

Zagreb, 02 October 2023

MASTER THESIS ASSIGNMENT No. 275

Student:	**Maja Magdalenić (0036509024)**
Study:	Computing
Profile:	Software Engineering and Information Systems
Mentor:	prof. Nikola Mišković

Title: **Vessel registration plate detection using a monocular camera**

Description:

The goal of this task is to develop a system that will recognize vessels and read license plates using a single monocular camera. This task includes the development of a detection or segmentation model that will recognize ships (potentially other objects in the environment), as well as the development of a license plate detection model. The result of the task is the image from the camera with labelled boat's position and the number of the license plate. The student must develop an algorithm that runs in real time, without significant delay.

Submission date: 09 February 2024

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 2. listopada 2023.

DIPLOMSKI ZADATAK br. 275

Pristupnica:	**Maja Magdalenić (0036509024)**
Studij:	Računarstvo
Profil:	Programsko inženjerstvo i informacijski sustavi
Mentor:	prof. dr. sc. Nikola Mišković

Zadatak: **Detekcija registracijske oznake plovila korištenjem monokularne kamere**

Opis zadatka:

Cilj ovog zadatka je razviti sustav koji će prepoznavati plovila i iščitati registarske oznake koristeći jednu monokularnu kameru. Ovaj zadatak uključuje razvoj modela detekcije ili segmentacije koji će prepoznavati brodove (potencijalno i druge objekte u okruženju), kao i razvoj modela detekcije registarske oznake. Rezultat zadatka je prikaz slike iz kamere s naznačenom pozicijom broda i ispisom njegove registarske oznake. Student mora razviti algoritam koji se izvodi u stvarnom vremenu bez značajnog kašnjenja

Rok za predaju rada: 9. veljače 2024.

Content

Introduction

Shipping plays a crucial role in facilitating global trade, connecting economies and providing an efficient means of transportation for various goods and resources. A precise and effective method for identifying ships holds significant importance for the intelligent transportation system (ITS) in scenarios related to waterway shipping. Another reason for the increasing emphasis on ship license plate recognition these days is the development of smart ports, with their primary focus being on preventing unlisted ships, stopping illegal fishing, and eliminating potential collisions among other objectives. In addition, efficient ship license plate recognition can be a valuable component of maritime security in areas where ships require protection from terrorism, piracy, robbery and illegal trafficking of goods and people. Ship license plate serves as the unique identity card for a ship, providing valuable information about the ship's condition or even its owner. It is also one of the key input information for ship license plate recognition.

The main task of this work is to develop a system that receives an image from a monocular camera as input and outputs the read license plate of the ship. The problem of recognizing license plates is divided into two main components: the detection phase, which includes locating the part of the image containing the license plate, and the recognition phase, which uses the detected part of the image to read the characters. There are various methods that can be used to perform both phases. In this thesis we compare different combinations of methods to identify the one that achieves the highest accuracy in the assumed environment.

In the first chapter of this thesis, an overview is provided on works of similar interests, with a focus on the methods applicable to our study. Chapter 2 Dataset contains a description of the dataset used for training, testing and validating machine learning models in the context of detection. This dataset represents the environment in which we assume the recognition model will operate. This chapter is followed by the 3 Methodology, where used methodologies and models are described. In 4 Results chapter, the results obtained by different methods are presented, compared, and discussed, followed by the conclusion. At the end, there is an additional chapter containing instructions for using the developed recognition model.

1. Related work

In contrast to the well-established task of car license plate recognition, ship license plate recognition remains challenging. This difficulty comes from the complex scenarios, various conditions, and a lack of available data. So far, only a few results have been reported.

The only license plate recognizer we are familiar with that deals with exactly the same topic (Ship license plate recognition) works with Chinese license plates. In this work, an approach for online adaptive real-time ship plate recognition based on DCNN (DRASLPR) is proposed [1]. The workflow of the ship license plate recognition system developed in this work is illustrated in Figure 1.1. First, DRASLPR collects and decodes video stream from cameras. Then, it locates the ship using SSD (Single Shot Detector). Subsequently, it detects the ship license plate using a new detection approach based on a modified SSD, referred to as SPD (Ship Plate Detector). Unlike typical objects, ship license plates tend to have large aspect ratios. Consequently, SPD defines these aspect ratios as elongated rectangular boxes. To generate rectangular receptive fields, SPD departs from conventional 3x3 convolutional filters and instead employs non-uniform 1x5 filters. These filters are better suited for detecting ship license plates with larger aspect ratio, and this is why SPD has better performance than SSD for ship license plate detection. The last step is plate recognition based on DCNN classifier trained on ship classification dataset to distinguish ship license plates. The training dataset is the rectangular area of ship plates, and each class represents a ship plate. Additionally, in recognition part, DRASLPR uses AIS (Automatic Identification System) to get names and locations of ships.

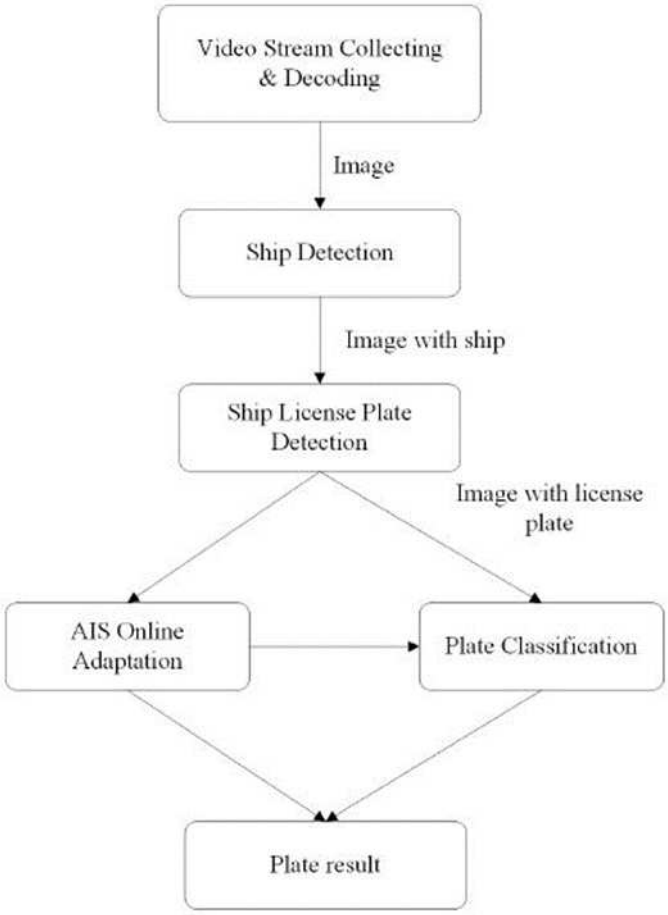

Figure 1.1 DCNN based Real-time Adaptive Ship License Plate Recognition workflow

An alternative approach to the task involves employing general license plate recognition methods designed for other vehicles. For instance, car license plate recognition methods are well-established and integrated to daily life, playing a crucial role in applications such as traffic management, law enforcement, and automated toll collection systems. The idea is to utilize these methods as a foundation and modify the necessary components to suit our specific environment. One of the recent studies on this topic separated the recognition system in just two components: plate detection using CNN and character recognition using OCR (Optical Character Recognition). The presented system workflow is shown in Figure 1.2. The paper [2] also presents a comparison of various methods, demonstrating that the chosen approach achieves the highest accuracy, with 100% detection accuracy and 96,23% recognition accuracy.

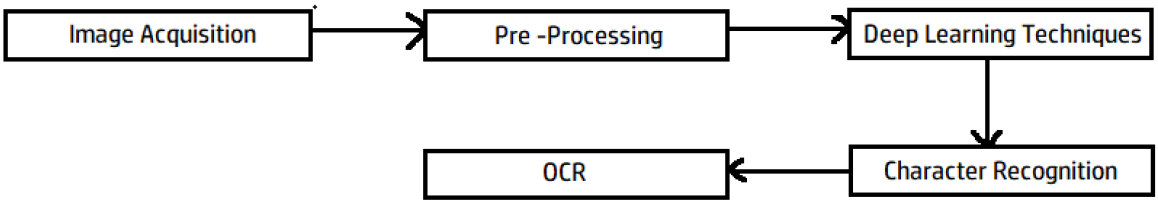

Figure 1.2 OCR based car license plate recognition workflow

The platerecognizer.com platform offers automatic license plate recognition software that is can be used in various environments and can be optimized for different locations. This recognition software is available online as a web application, and additionally, they provide the source code, allowing us to explore local implementation with our own dataset. In a preliminary experiment, the recognizer was used both online and offline with images containing ships from the dataset. The dataset description will be presented in chapter 2 Dataset. The region was set to Global. The results appear accurate for images with clearly visible license plates and vehicles positioned close enough to the camera. However, in other scenarios, a noticeable decrease in accuracy is visible. The example of usage is shown in Figure 1.3. While this experiment does not offer highly informative insights at this stage, we can compare its results with those obtained from the recognition system developed as part of this thesis in subsequent analyses.

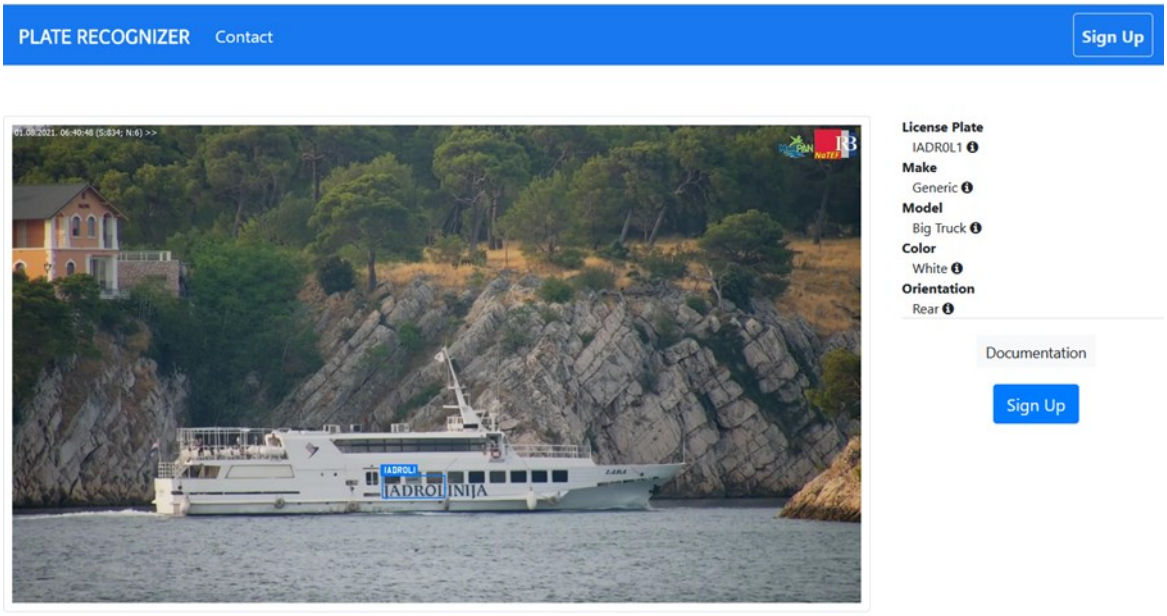

Figure 1.3 Online license plate recognizer results

In the first mentioned approach the problem is the inability to use text recognition system described, as we require our recognizer to identify latin letter characters. Nevertheless, the paper [1] presents license plate detection method that appear to be the most accurate in marine environment in which their dataset was collected. It is reasonable to assume that this method would perform well in our marine environment as well. Regarding the recognition aspect, we can train a CNN classifier model with our own dataset. This approach would enable us to adapt the recognition part for identifying latin letter characters, but later, we will see that this was not feasible in our case.

Some of the methods used in car license plate recognizer described before, due to their high accuracy and simplicity, are reasonable to utilize in this work and will be further described below.

1.1. Optical character recognition (OCR)

Our first goal in the process of developing ship license plate recognizer is to find and recognize text in the image. Our starting point is an already existing tool called optical character recognition (OCR). OCR is a technology used to extract text from real images, captured text or documents and converting it into machine-readable text. It works in a way that separates text from the background based on pixel differences [3]. We assume that our camera captures a maritime landscape that eventually contains ships and the only text we expect to see in the image (excluding date and timestamps) should be the ship's name, some marks or license plate details. Since the dataset that will be used in this work, which will be described in 2 Dataset chapter, does not contain many images with ship license plates, initially we will focus on extracting any type of text available in the image.

OCR tool works in four main steps:

- **Image acquisition** – In this step OCR generates a black-and-white version of the input image. Since OCR is a binary process, it identifies elements that exist or don't. In an ideal scanned image, any black will be a part of a character that need to be recognized and any white part will be part of the background. Reducing the image to black and white is the first stage in figuring out the text that need to be recognized [3].
- **Preprocessing** – This stage is the one in which the OCR software cleans the input image and make it easier for OCR to read the text. It includes cleaning techniques like deskewing or tilting the image to fix the alignment issues, removing any digital image

spots, or smoothing the edges of text images, cleaning up boxes and lines in the image etc [3].

- **Text recognition** - This stage typically includes processing and recognizing one character, word, or block of text at a time. The two main types of processes that are used to identify characters are pattern recognition and feature extraction.
  - **Pattern recognition** functions by isolating a character image known as a glyph and matching it with a similar stored glyph. For effective pattern recognition, the stored glyph should be written in the same font and size as the input glyph. This approach proves effective particularly with scanned document images typed in recognizable fonts [3].
  - **Feature extractions** involves breaking down or analysing glyphs into distinct elements like lines, enclosed shapes, line orientations, etc. These elements are used to identify the closest match of the most similar stored glyph or "the nearest neighbour" [3]. Most modern OCR-s work by feature extraction rather than pattern recognition. Most of them use artificial intelligence.
- **Post – processing –** Once the program had recognized all the characters it presents the recognized text which can potentially have errors. In post-processing part program tries to identify misspelling or other mistakes that can be noticed. The fixing part can be done using a spell checker that identifies misspelling and offers some alternatives, word or sub-word-based language models etc. This step can be particularly important part if we are working with captured text document, but in our case, it will not have such an impact as the text we want to recognize does not contain words with meaning.

1.1.1. OCR evaluation metrics

Utilizing OCR tools for text recognition requires evaluating the tool on a test dataset to obtain information about its accuracy, ultimately enabling us to compare its results with the results given by other recognition methods. OCR accuracy is defined as the process of comparing the output of OCR with the original version of the same text (ground truth). For example, if our input image contains 100 characters and the OCR correctly identified 99 of them, the character level OCR accuracy is 99%. As it is mentioned above, recognizing process contains many different steps and each of them has an influence on the accuracy level achieved at the end of the process. The metrics used for evaluation are as follows:

- **Character error rate (CER)**

  CER evaluation is based on the concept of Levenstein distance, which means that we count the minimum number of transformations required to transform the ground truth into the OCR output [4]. For example:

  Ground truth: ZD3850

  OCR output: ZO850

  →O instead of D, missing 3

  →Number of transforms = 2 ; Number of correct characters = 4

$$CER = \frac{T}{T+C} \times 100\% = \frac{2}{2+4} \times 100\% = 33{,}33\% \quad (1)$$

  There is no single value to define a good CER value as it depends on the use case. For complex cases involving heterogeneous and out-of-vocabulary content, a CER value as high as 20% can be considered satisfactory.

- **Word error rate (WER)**

  WER evaluations is also based on the concept of Levenstein distance, but it considers the whole word as an element, not just the character. It is highly correlated with CER value although the absolute WER value is expected to be higher than the CER value [4].

- **Precision**

  Precision metric is the percentage of correctly recognized items with respect to the total word count of the OCR output, where items refer to either characters or words [4].

$$precision = \frac{number\ of\ correct\ items}{number\ of\ items\ in\ OCR\ output} \quad (2)$$

- **Recall**

  Recall metric is the percentage of items correctly recognized by the OCR engine, where items refer to either characters or words. In the OCR- related literature, the term OCR accuracy often refers to recall [4].

$$recall = \frac{number\ of\ correct\ items}{number\ of\ items\ in\ ground\ truth} \quad (3)$$

2. Dataset

For the purposes of this work the following two datasets were used:

- Šibenik Bay CCTV static camera (Sibenik dataset.7z)

The images from the dataset were collected at the Šibenik Bay using a CCTV static camera. Most images show ships from the side during the day and in clear weather conditions. The images are in HD resolution (1280x720 pxls). The dataset contains annotations for ship detection with one class (Ship/boat) in YOLO format. The examples of images from the dataset are shown in Figure 2.1.

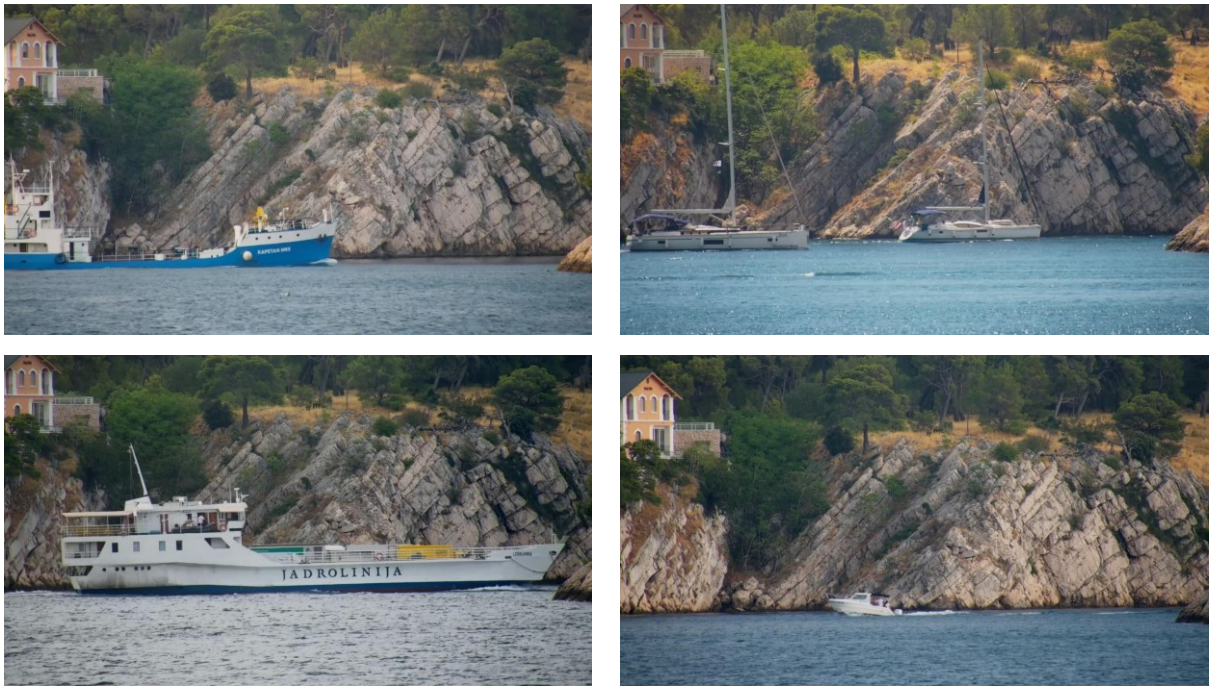

Figure 2.1 Examples of the images from the Šibenik Bay Dataset

- The Split Port Ship Classification Dataset [5]

The Dahua LR camera recorded video sequences of maritime traffic from February 2020 to December 2022. In addition, the video sequences were recorded at different times of the day, an under different weather and sea state conditions. Then, images with a full HD resolution were extracted from the video sequences and manually selected for dataset. The images are saved in .jpg format. It should be noted that the focus was on images of ships leaving or entering the port of Split, and that the dataset shows mainly the port and

starboard sides of the ships. Based on the goals and purpose of the dataset, as well as the nature of the Port of Split, the images in the dataset are annotated for the detection of 12 specific ship categories (Small Craft, Small Fishing Boat, Small Passenger Ship, Fishing Trawler, Large Passenger Ship, Sailing boat, Speed Craft, Motorboat, Pleasure Yacht, Medium Ferry, Large Ferry, High Speed Craft) in YOLO format. The images in this dataset contain timestamps. In our case, these timestamps could be recognized as text on the original image. To avoid that, the images were cropped in a certain ratio. The examples of images from the dataset are shown in Figure 2.2.

Figure 2.2 Examples of the images from Split Port dataset

The datasets were primarily used for training the text detector, the description of which is provided in 3.4.2 Training YOLOv8 for text detection. The first step in preparing the datasets for our use case was going through the whole datasets to separate the images where ships contain textual labels of any kind (registration plates, ship names…) as images without text did not hold value for training. Since the developed text detector takes an image of the ship as input and provides text detection as output, for training purposes, images of the ships without the surrounding environment were required. Fortunately, both datasets have annotation files for ships, so images of the ships (without the surrounding environment) are easily obtained by cropping the original image according to the annotations.

After cropping the ships from the original images, the obtained images were annotated for text detection using the CVAT web platform. The CVAT tool was chosen for annotation for several reasons: the tool is completely free, the number of annotations is unlimited, and the platform is easy to use.

The final dataset contains 7201 images – 3582 images obtained by cropping ships from the Šibenik Bay dataset, and 3619 images obtained by cropping ships from the Split Port dataset. The dataset is divided into training, testing and validation part in the ratio of 8:1:1.

3. Methodology

Ship license plate detection includes two sub-tasks: ship license plate detection and ship license plate text recognition. The first one is to locate a license plate on the vehicle by predicting a geometric bounding box and the second one is to recognize the text characters using the regions of interest identified by the detection model. The combinations of different methods used for these two phases in this work are presented and explained in the following subsections.

3.1. Choosing the best performing OCR

The two of the most popular OCR tools today are Paddle OCR and Easy OCR. Both tools are free, and this makes them more accessible to all kinds of audience. They have been used for text extraction across various applications and are known for their high accuracy.

Paddle OCR is a deep learning-based OCR system created by PaddlePaddle, a Chinese AI firm. Paddle OCR is built on the PaddlePaddle framework, which is well-known for its quick and efficient deep learning algorithms. The model supports numerous languages, including Chinese, English, Japanese and Korean, and can properly detect different text styles and fonts. It has achieved state-of-the-art performance on various OCR benchmarks, including the ICDAR (International Conference on Document Analysis and Recognition) 2015 and ICDAR 2017 competitions. It has a user-friendly interface that allows users to quickly train and deploy OCR models. One of the limitations of the model is limited community support. Paddle OCR is a relatively new OCR system, and its community is not as large as some of the other OCR-s, making it harder to find resources and support [6].

Easy OCR is a python-based OCR library that supports over 70 languages and can recognize various text styles and fonts. It is known for its ease of use and fast processing speed. The OCR has a simple interface and can be easily integrated into python applications. The downside of this model is limited customization. Easy OCR does not provide as many customization options as some of its competitors including Paddle OCR, making it harder to fine-tune models [6].

Both models are optimized for speed and can process large volumes of images in real-time, making them suitable for applications that require high throughput.

The question here is which of the two should be used in our assignment. There is not much information about which of the two models is better in which conditions. In a recently published work [7] comparing various OCR tools, the first comparison was made only on images of numbers, and in this case, Paddle OCR achieved very high accuracy compared to others. In the second comparison, which was made on text documents, contrary to expectations, Easy OCR turn out to be the most accurate. Paddle OCR showed weakness especially with special characters, periods and commas, and it was not much faster with GPU. The results of the comparison are presented in Table 1.

Table 1 Blu delta comparison of Paddle OCR and Easy OCR [7]

	Paddle OCR	Easy OCR
Exact match %	49,05	64,62

From the conducted comparison, it is evident that the accuracy of OCR relies primarily on the specific use case and the nature of the data it processes. For this reason, in our case, the choice was made depending on testing both tools on the test dataset described in Dataset chapter. Testing was performed using the first two methods described below. Following the testing, it is necessary to evaluate the model, the results of which are shown in Table 2. There is not a significant difference in the accuracy of the two OCR tools using Method 1, but the improvement of Paddle OCR is visible in the other case. Based on the results, we can conclude that in our environment, Paddle OCR achieves higher accuracy in text recognition, making it the preferred choice for the use case in this thesis.

Table 2 Comparison of Paddle OCR and Easy OCR tested on the test dataset

	OCR	CER %	WER %	Words Accuracy %	Character Accuracy %
Method 1	Paddle	78,95	86,22	13,78	21,05
	Easy	75,03	89,39	10,61	24,97
Method 2	Paddle	54,99	73,83	**26,17**	**45,01**
	Easy	77,62	89,73	**10,27**	**22,38**

3.2. Method 1 – Direct ship registration plate detection and recognition using Paddle OCR

The first and simplest method that we use is letting OCR handle both phases of the recognizer. The OCR tool has a built-in detection component that we can use, or we can develop the detection part of the system on our own. We can also choose from different algorithms used for text detection, and selecting the desired algorithm is done simply by setting the `det_algorithm` argument in the OCR model. In this work, the Differentiable Binarization (DB) algorithm is used, this is the default text detection algorithm in Paddle OCR, the OCR tool that we chose to use. The scheme of the starting model is shown in Figure 3.1 while it will be enhanced later. The model input is the original image from the testing part of the dataset described in 2 Dataset. The image is forwarded to the OCR tool, which initially detects the text in the image using the DB algorithm. The identified text is subsequently transferred to the recognition part. Additionally, our selected OCR offers several algorithms from which we can choose to perform recognition. In this work, the default, Convolutional Recurrent Neural Network (CRNN) algorithm is chosen.

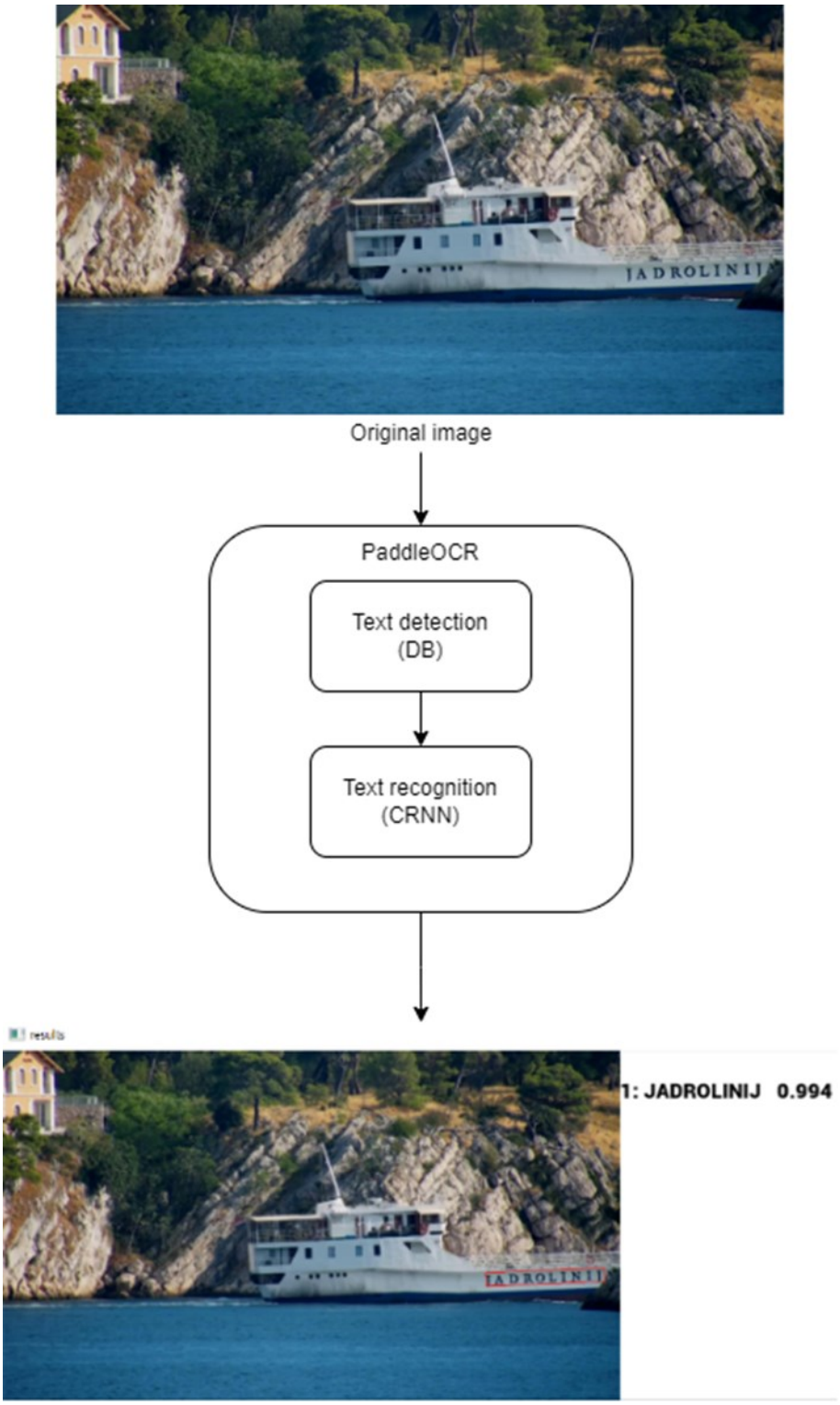

Figure 3.1 The structure of the initial model

3.3. Method 2 – Using YOLOv8 ship detector and Paddle OCR

In the next step, the idea is to “bring the ship closer to the camera”, meaning we will modify the detection part of the system so that it provides the recognition part with a clearer and closer text image. One solution is to divide the detection into two steps: ship detection and text detection. Effectively, this means that the input to the OCR is no longer the original image from the camera, but an image specifically focused on the zoomed-in ship. After that, the OCR uses its default text detection algorithm on the ship and proceeds with the recognition. For that purpose, a ship detection algorithm is required. The scheme of the described enhanced model is shown in Figure 3.2.

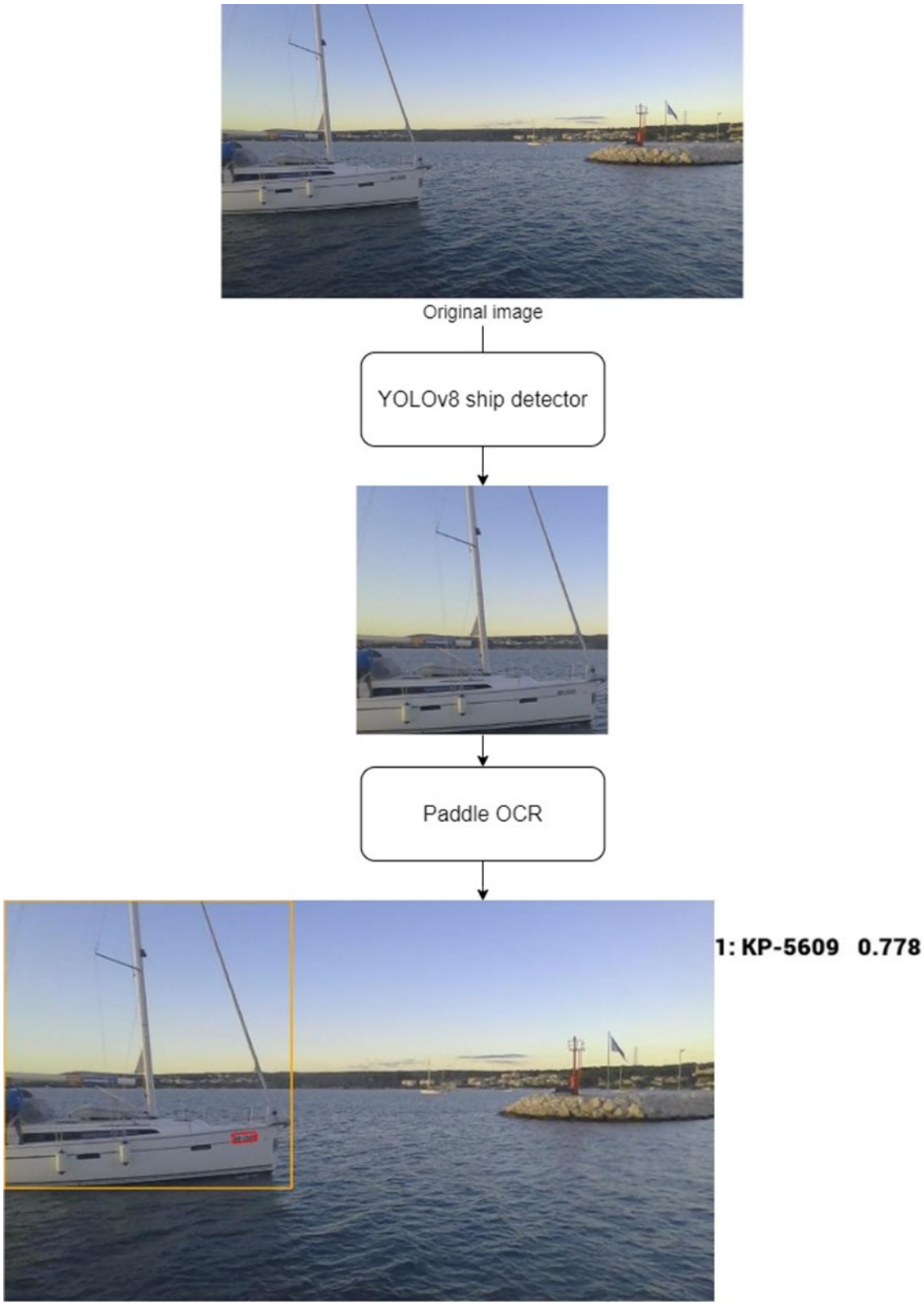

Figure 3.2 Method 2 model scheme

3.3.1. Ship detection algorithm

The detection model used in this work was developed for the USV-based Maritime Obstacle Segmentation and Detection challenge that took place as a part of the second Workshop on Maritime Computer Vision (MaCVI) 2024. The task for the Obstacle Detection challenge was to develop an obstacle detection method that detects the obstacles in an input image and represents their location in the image using rectangular bounding box. There were 8 expected classes of dynamic obstacles (boat, buoy, other, rowboat, swimmer, animal, paddle board, float) and three classes that cover the pixels in the image that do not correspond to any of the dynamic obstacle's classes [8]

As a detection method in the challenge, the latest version of YOLO (You Only Look Once) algorithm, YOLOv8 was used. The detailed algorithm architecture will be described in 3.4.1. The largest YOLOv8 model pretrained on the COCO dataset, YOLOv8x, served as a starting point and was subsequently fully trained for the challenge.

The datasets used for training the detection model include:

- Open Images V7 (20000 images) (https://storage.googleapis.com/openimages/web/index.html)
- Common Objects in Context images (776 images) https://storage.googleapis.com/openimages/web/index.html
- Web scraped images (469 images) – random web-scraped relevant images of a maritime environment, but mostly images from the Caltech
- Šibenik Bay CCTV static camera (2333 images) Sibenik dataset.7z
- LaRS dataset – newly released dataset consisting of 4000+ USV-centric scenes captured in various aquatic domains, designed specifically for this use case (https://lojzezust.github.io/lars-dataset/)

The neural network was first pre-trained on the mentioned datasets, with the final training conducted on the LaRS dataset. The training pipeline is shown in figure below.

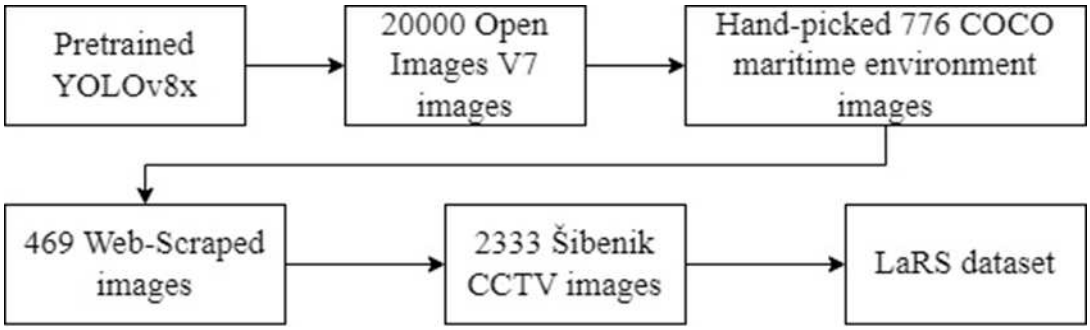

Figure 3.3 Training pipeline [8]

Detection results given by using the final model are presented in Figure 3.4 and Figure 3.5.

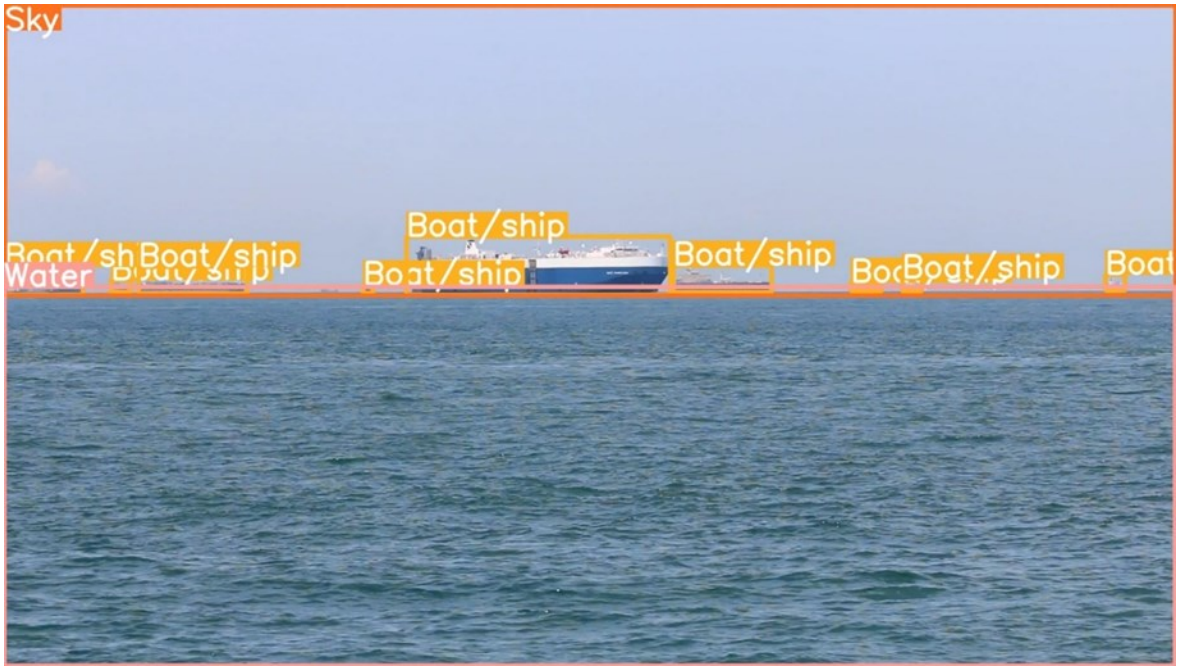

Figure 3.4 Detection results in an open water environment [8]

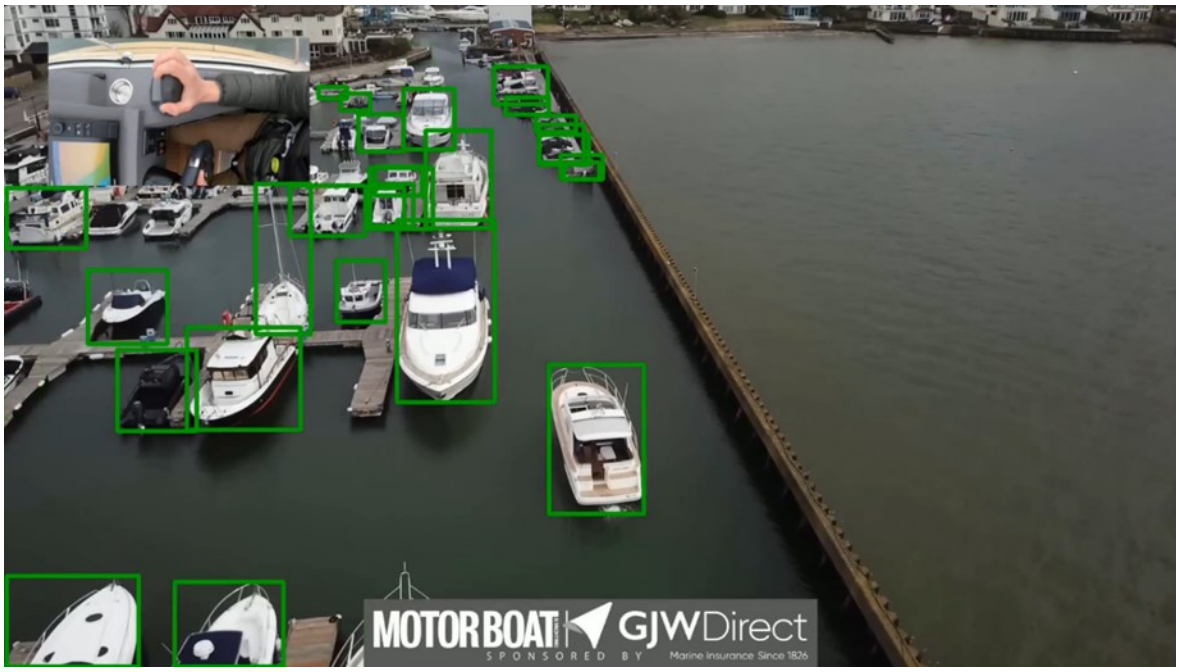

Figure 3.5 Detection results from a drone shot in a crowded marina [8]

With an F1 score of 50,51 the model achieves the highest accuracy among models developed in the challenge, securing the first place.

As it was mentioned before, detection algorithm developed as a part of the challenge detects 10 classes (static obstacle, water, sky, boat/ship, row boats, paddle board, buoy, swimmer, animal, float, other). The usage of the algorithm is limited to ship detection in our case, and for that reason, the model output in this work is filtered to two classes: boat/ship and row boats.

3.4. Method 3 - Using YOLOv8 ship detector, YOLOv8 text detector and Paddle OCR for recognition

In Method 3, a new text detection algorithm has been developed and will be used instead of the default detection algorithm (DB) that was used in the first two methods as a part of the OCR. YOLOv8 ship detector from Method 2 is also used in this method. It gets an original image as an input and sends the output (cropped ship image) to text detection model. Text detection model provides a bounding box with the detected text, and this part of the ship is then sent to the OCR. In this case, we set the OCR `det` argument to `False`, and OCR does not execute the detection algorithm on the image, as we only use the part of OCR that is used for recognition (CRNN). The scheme of the Model 3 is presented in the image below.

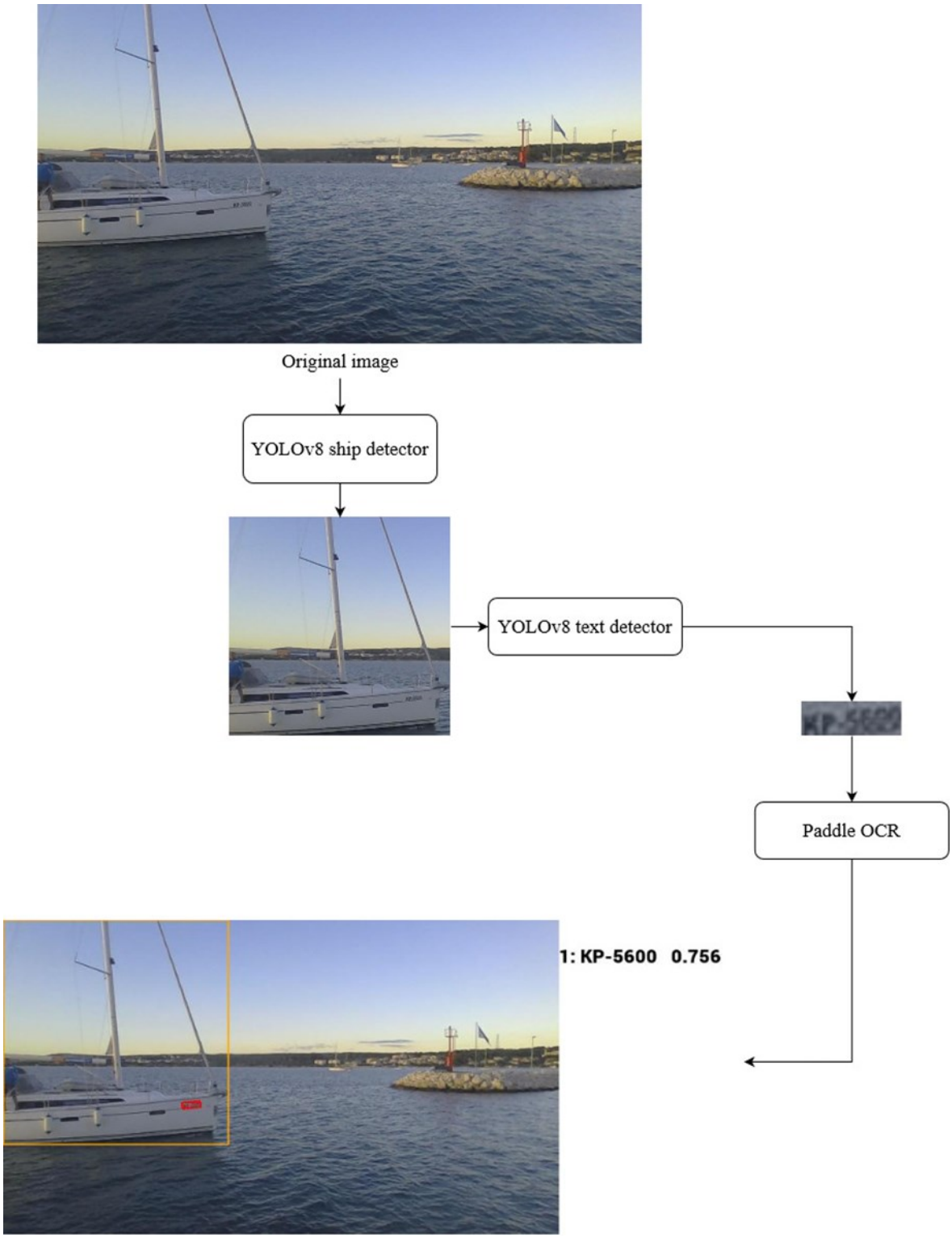

Figure 3.6 Method 3 model scheme

3.4.1. YOLOv8 detection algorithm

YOLO, short for You Only Look Once, is a popular object detection algorithm that has revolutionized the field of computer vision. The key feature of YOLO is its single – stage

detection approach. Unlike two – stage detection models, such as R – CNN, that first propose regions of interest and then classify these regions, YOLO processes the entire image in a single pass. The model works by dividing the input image into a grid and then directly predicts bounding boxes and class probabilities for each grid cell. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object. Each grid cell predicts B bounding boxes and confidence scores for those bounding boxes. This one – step process enables YOLO to be incredibly fast while maintaining accuracy, making it ideal for real – time applications that rely on fast and robust object detection, such as video surveillance, autonomous vehicles, and augmented reality. Fast YOLO is the fastest general – purpose object detector in the literature, and YOLO advances state of the art in real – time object detection. YOLO architecture has seen multiple upgrades and changes over the years. For the purpose of this this work, YOLOv8 has been used. Ultralytics claims that the 8th version of YOLO, the newest version at the time of writing, is in general smaller, faster, and more accurate than its predecessor [9]. The detailed comparison and the model's progress over time are shown in the image below.

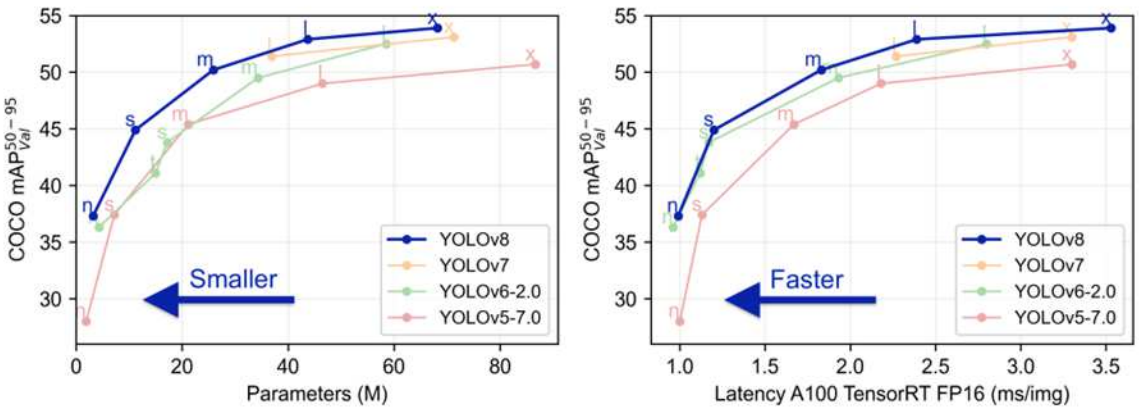

Figure 3.7 YOLOv8 compared to previous YOLO versions across all five model sizes [9]

The actual YOLOv8 paper has not been released yet, hence there is not much information about the architecture of the model. However, we will still try to get an overview of the model as the model architecture is similar to previous versions, with a few modifications. Here's a brief description of the architecture of YOLOv8:

- Backbone network

  YOLO typically employs a backbone network as its feature extractor. The backbone of YOLOv8 is a modified version of the CSPDarknet53 architecture, featuring 53

convolution layers. What sets it apart is the incorporation of cross – stage partial connections, enhancing information flow between layers. This strategic design improves the model's ability to understand complex patterns and relationships within images [11]

- Detection Head

  The head of YOLOv8 is composed of multiple convolutional layers followed by fully connected layers. This segment plays the crucial role of predicting bounding boxes, objectness scores, and class probabilities for identified objects in an image. It acts as the decision – making hub, refining the model's predictions with each layer [11]

A simple scheme of the yolo model architecture is shown in the Figure 3.8.

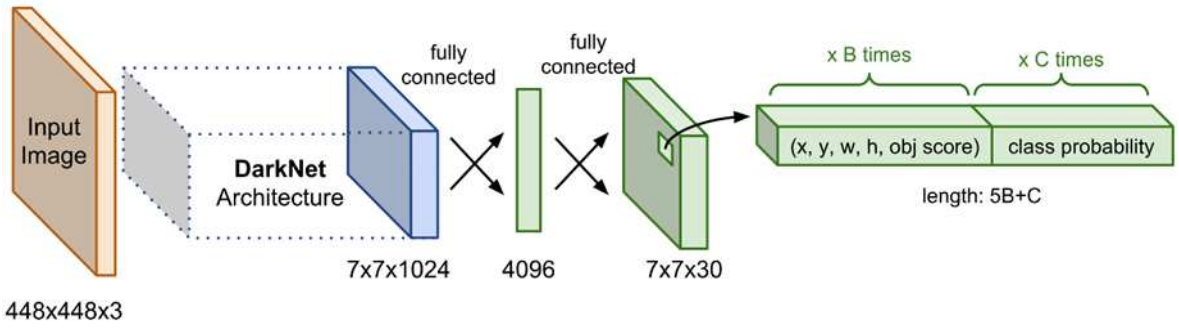

Figure 3.8 The network architecture of YOLO [12]

Key features of YOLOv8 in comparison to its predecessors include:

- Self – attention mechanism

  YOLOv8 incorporates a self - attention mechanism into the network's head. This novel feature enables the model to dynamically focus on different regions of an image, adjusting the significance of various features based on their relevance to the detection task. This enhances adaptability and contributes to an overall improvement in performance [10].

- Multi – Scaled Object Detection

  Within its architecture, YOLOv8 introduces a feature pyramid network. This network comprises multiple layers specifically designed to detect objects at various scales. This multi – scaled approach enhances the model's capability to effectively identify objects of different sizes within an image.

- Anchor – Free detection

Anchor boxes are a pre-defined set of boxes with specific hights and widths, used to detect object classes with desired scale and aspect ratio. They were incorporated in previous YOLO models, and they generally improved training by increasing the average precision. In YOLOv8 the architecture moved away from anchor boxes. The advantage of anchor – free detection is that it is more flexible and efficient, as it does not require the manual specification of anchor boxes, which can be difficult to choose and can lead to suboptimal results in previous YOLO models [10].

- Mosaic Augmentation

  During training, YOLO model does many augmentations to training images. One such augmentation is mosaic data augmentation – a simple technique in which four different images are stitched together and fed into the model as input. This makes the model learn actual objects from different positions and in partial occlusion. Performing mosaic data augmentation is shown to reduce performance, so it was switched off for the last 10 epochs

3.4.2. Training YOLOv8 for text detection

In the previous chapter, the architecture of the YOLOv8 model, some of its important features, and the reasons it achieves the highest accuracy and speed compared to other detection models were presented. These are the reasons why it is ultimately chosen as the algorithm for text detection in Method 3. Ultralytics provides pretrained models of all their YOLOv8 architecture sizes on COCO dataset. The model that was fully trained for the purpose of this work was YOLOv8m model. In Table 3 the comparison of different architecture sizes are presented. In the object detection comparison of the five model sizes, the YOLOv8m model achieved mAP of 50,2% on the COCO dataset, whereas the largest model, YOLOv8x, achieved 53,9% with more than double number of parameters and half the detection speed. Considering that, in our case, the image goes through two detection models and OCR for recognition, speed plays a significant role, which is why YOLOv8m was chosen due to not satisfactory performance.

Table 3 Performance comparison of different YOLOv8 architecture sizes [9]

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

The pretrained YOLOv8m model was additionally trained on the dataset described in 2 Dataset chapter. It is important to note that the model was trained only on images of cropped ships (without the surrounding environment), as this is the type of images the text detector will receive as input from the ship detector. Examples of images used for training are shown in Figure 3.9. As visible from the examples, the dataset contains ships of various sizes, with and without license plates. Some ships have other markings or the ship's name, while others have no text at all. In some images, the text is clearly visible, while in others, we can see that there is some text, but it is blurred and not legible. Training part of prepared dataset contains 5759 labeled images. The training was initiated on 300 epochs on the NVIDIA GeForce GTX 1080 Ti device and it lasted 6,41 hours. Using the early stopping method, it was interrupted after 195 epochs as there was no improvement observed in last 50 epochs.

Figure 3.9 Examples of images used for training the text detector

3.4.3. YOLOv8 text detection model results

Confusion matrix for the validation set of data after YOLOv8m finished training is shown in Figure 3.10. A perfect matrix with no errors would be diagonal. As presented in the upper right field of the matrix, only 3% of the text was predicted as background (not detected, false negative). In the bottom left matrix field, the percentage of "background" labels predicted as text is shown, and it amounts to 14%. This error is not of great

importance in our case since false detected text (which is not actually text, false positive) will not be read by the OCR anyway.

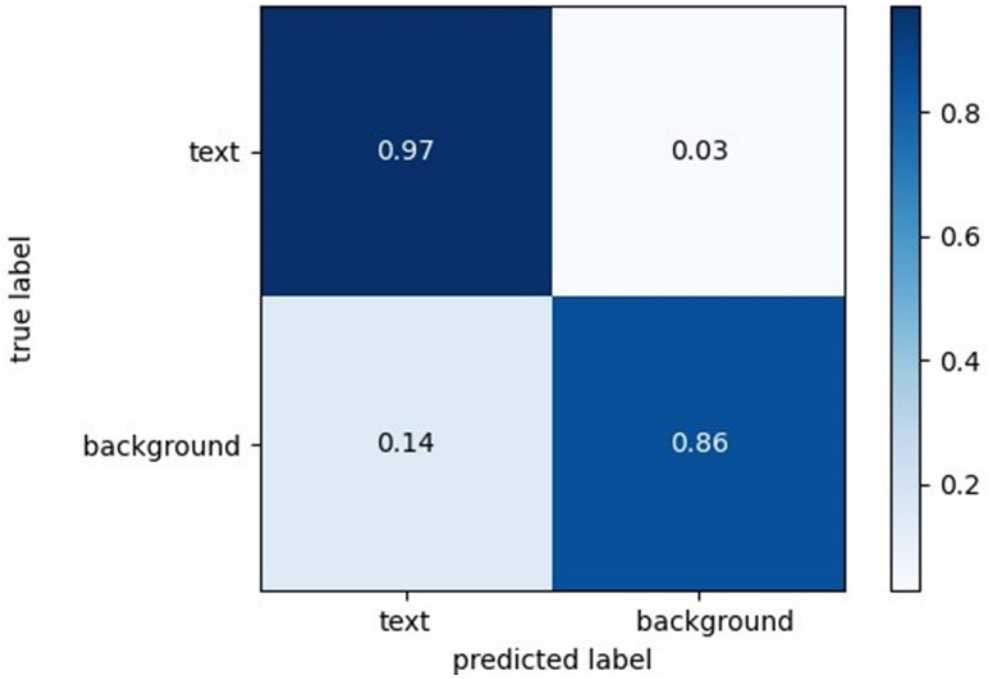

Figure 3.10 Trained model confusion matrix

Some of the other evaluation metrics are presented in Table 4. Precision quantifies the ratio of true positives among all positive predictions (assessment of the model's ability to avoid false positives i.e. false text detection in our case) and recall calculates the ratio of true positives among all actual positives, measuring the model's ability to detect all instances of a class. The F1 score is the harmonic mean of precision and recall, providing balanced assessment of a model's performance while considering both false positives and false negatives. In a perfect scenario, both metrics would be equal to 1. The model achieves precision of 93% and recall of 97%. These numbers confirm what could be inferred from the confusion matrix as well. The model generally achieves very high detection accuracy with a certain percentage of false detections, although this number is not significant for our purpose. Each detected bounding box subsequently goes through the OCR for recognition. In the case of false positives, the characters will not be read.

Table 4 YOLOv8 text detection results

	Precision	Recall	F1
YOLOv8 text detector	0,9367	0,9705	0,9533

3.5. Improvement of method 3 – Resizing detected word

The motivation for the modification in Method 3 is clearly depicted in Figure 3.6, illustrating the path the image takes from input to recognized license plate. For ships that are farther away from the camera, the detected text, and therefore the size of the image sent to the OCR for recognition is of very small dimensions. Some of the texts are barely readable to the human eye. By analysing the sizes of the text detected, it has been observed that the texts correctly recognized by OCR are generally larger than 8000 pixels per image. This is a starting point for the improvement of Method 3. Once the YOLOv8 model detects the text area smaller than 8000 pixels, it is first resized by a certain percentage, and then the enlarged image is sent to the OCR for reading. As we aim to preserve the original aspect ratio of the image, we proportionally increase both dimensions by the same factor, ensuring that after enlargement, the area of the image will be approximately 8000 pixels. Simple code lines with the factor calculation and resizing the image are listed below:

```
height, width = img.shape
size = height * width
      if size < 8000:
            ratio = 8000/size
            factor = math.sqrt(ratio)
      img = img.resize((int(width*factor), int(height*factor))
```

Code 2.1 – Text image resizing factor calculation

The `ratio` is the number that indicates how many times the image needs to be enlarged to reach a size of 8000 pixels. It is calculated by dividing the 8000 pixels by the size of the image. To achieve the wanted size of the image, each dimension should be multiplied by the square root of the `ratio`. This way, we are able to preserve the original aspect ratio and obtain a larger image for the OCR. The schematic diagram of the described model is presented in Figure 3.11.

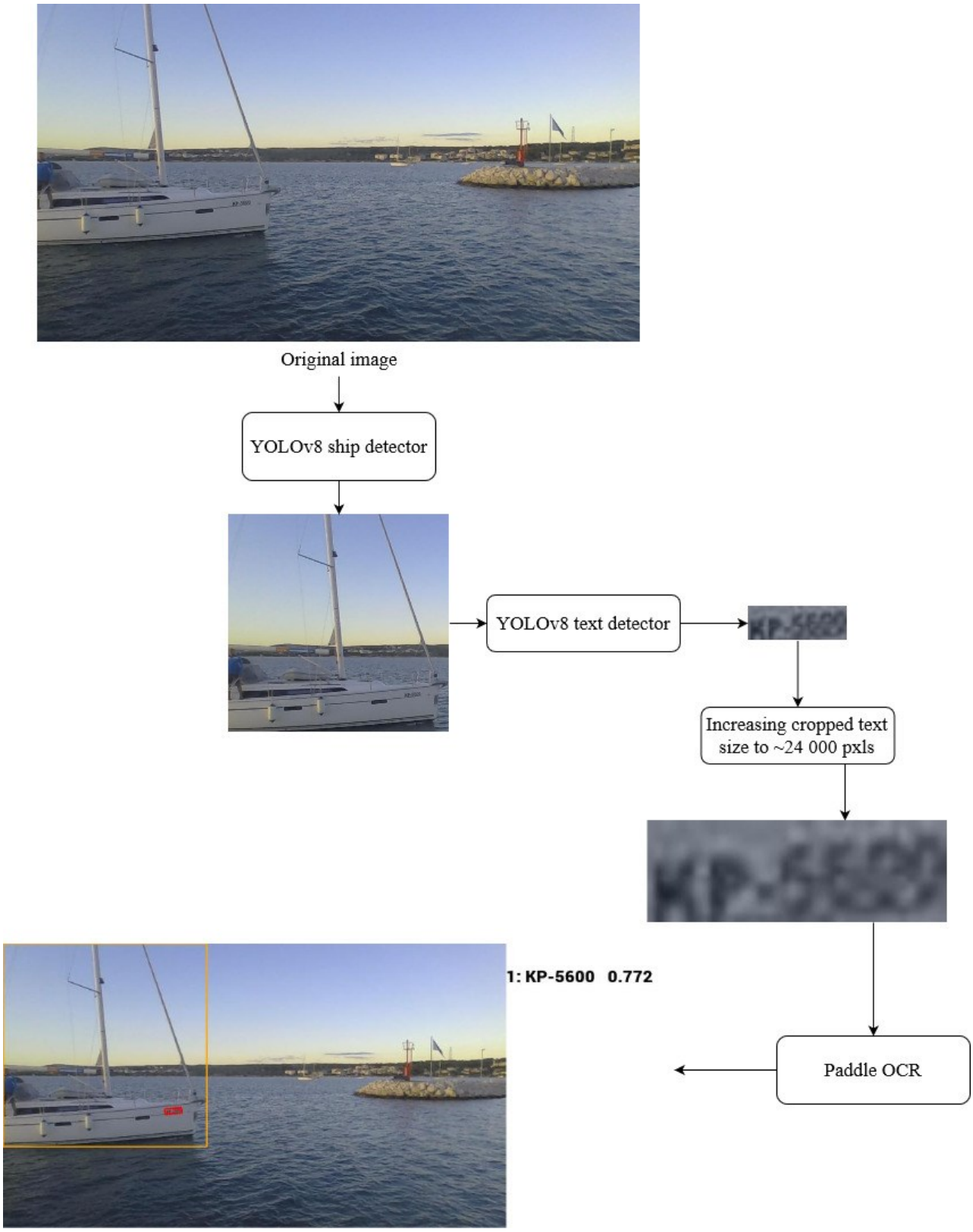

Figure 3.11 Improved method 3 model scheme

In the first text detection enlargement (8000 pixels), the size to which we increase the detection was selected by reviewing the tested images, based on the size of detections and the recognition accuracy. This does not necessarily mean that this size is indeed the best for our application. It is necessary to find the detection size that OCR generally recognizes the best, and then enlarge all detections that are too small to that size. Most images in the test dataset are in full HD resolution (1920 x 1080 pixels), which means that the image contains approximately 2 million total pixels. We started with the assumption that the text will cover at most 5% of the image, which would mean it could be approximately 100 000 pixels in size. The threshold for the detection size that is enlarged, and consequently, the size to which the detection will be enlarged, was set to 8000 pixels in the previous case. In the following cases, the threshold will be incrementally increased by a certain number up to the maximum threshold (100 000 pixels), and for each threshold, the detection model will be tested on the test dataset. An example of gradually enlarged detection is shown in Figure 3.12. After the tests, the recognition results were compared, and it was concluded that OCR has the most accurate recognition of the text size of 24 000 pixels.

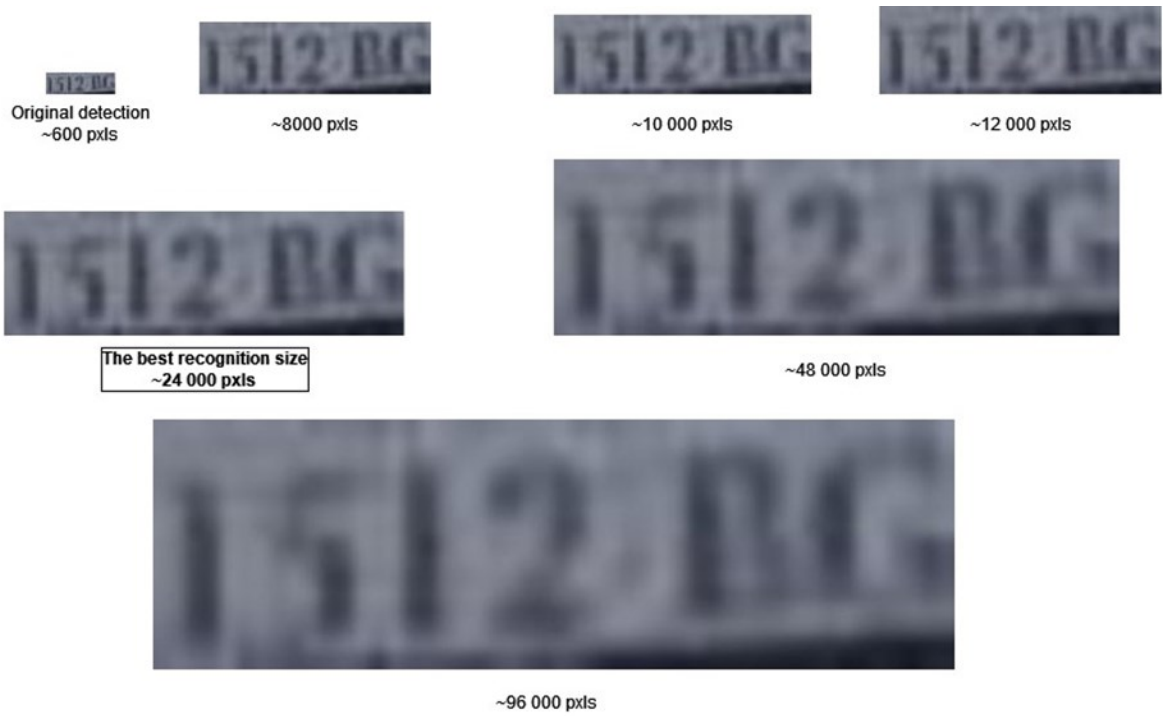

Figure 3.12 Example of gradually enlarged text detection

In the upper image, the enlargement of text detection is visible, but as expected, as we increase the text size, it becomes less clear. We attempted to solve this problem in two ways: by increasing the contrast and brightness, and by sharpening the given text image.

4. Results

In this chapter, the results of the methods described in the previous chapter will be presented, analysed and compared.

4.1. Method 1 results

Method 1 was the simplest method in which text detection and recognition were performed using the default algorithms of Paddle OCR. The results obtained by this method are shown in Table 5. An accuracy of 21,05% in character recognition is certainly not sufficient to accept as our main method. Additionally, if we pay attention to the WER value or words accuracy, we can see that the recognizer accurately recognizes just about 13% of license plates. Such performance is insufficient for using the recognition system for any practical purpose. The presented method is evidently not effective and cannot be used as a solution for our task. It needs to be modified or enhanced.

Table 5 Recognition results using Method 1

	CER %	WER %	Words Accuracy %	Character Accuracy %
Method 1	78,95	86,22	13,78	21,05

Upon reviewing the images and the results provided by Paddle OCR, we conclude that the recognition system has significantly higher accuracy in images where the ship is closer to the camera, with fewer disturbances and clearer text visibility, as could be expected.

4.2. Method 2 results

By reviewing the results and test images from Method 1, described in the previous subsection, we have concluded that the most effective improvement for enhancing the detector's performance would be to revise the detection part. By testing the described method on the test dataset, we obtain the results shown in Table 6.

Table 6 Recognition results using method 2

	CER %	WER %	Words Accuracy %	Character Accuracy %	Precision	Recall
Method 2	54,99	73,83	26,17	45,01	0,50	0,26

Even though the model performs better compared to the one using Method 1, the accuracy of 45,01%, and less than 30% correctly recognized license plates still does not meet our requirements. In the result table, precision and recall metrics have been added. Let's remember, the recall metric is the ratio of correctly recognized words (license plates, in our case) to all words (license plates) present in the images in the test dataset. The ideal value would be equal to 1, which would mean that the model detected all the words and correctly recognized them. Precision value represents the ration of correctly recognized words to all the words that are detected by the model. In our case, the precision value is one time higher than the recall. We can conclude that the model can recognize the words pretty well if they are detected, but our detection model is still not efficient enough.

4.3. Method 3 results

Method 3 was tested on the same test data set, as other two methods presented in this thesis, and the results are shown in Table 7.

Table 7 Recognition results using method 3

	CER %	WER %	Words Accuracy %	Character Accuracy %	Precision	Recall
Method 3	37,68	68,91	31,09	62,32	0,35	0,33

In comparison to Method 2, described in the previous chapter, the new method increased the character recognition accuracy by almost 50%. WER is the metric that gives us information about the correctly recognized license plates, and it is also higher that the metric in Method 2. It is reasonable to infer that the model reads characters more accurately, leading to improved recognition of entire words (license plates). If we take a closer look at precision and recall values, it is evident that the recall value has increased, while precision value decreased. The numbers may appear confusing, but the overall model performance is higher. Recall represents the ratio of correctly recognized words in relation

to the actual number of words in the images. The increase in recall is a direct result of an increase in the number of correctly recognized words, as the actual number of words in the images is always constant in the same dataset. Furthermore, precision represents the ratio of correctly recognized words in relation to the number of words that OCR successfully read. From the recall value, we know that the number of correctly recognized words has increased, and precision will only decrease if the number of words successfully read by OCR has also increased. In short, we have a higher number of words that the model managed to read and a higher number of words that it read correctly. As we did not change the recognition part of the model, we can assume that the increase in the number of read words is a result of the increase in the number of detected words, which is achieved by using the developed YOLOv8 text detection model. Finally, the increase in the number of read words leads to the increase in the number of correctly recognized words.

4.3.1. Improved Method 3 results

The improvement in Method 3 was made by adding an additional step in which the detected text area is enlarged by a certain ratio to a specific size in order to make it "easier" to read. The results obtained by the improved method are presented in Table 8.

Table 8 Method 3 results with added resizing for text detections smaller than 8000 pixels

	CER %	WER %	Word Accuracy %	Character Accuracy %	Precision	Recall
Improved Method 3	30,12	58,80	41,2	69,88	0,46	0,43

An improvement in the model's performance has been achieved, but it may be possible to further enhance the model using the same technique, which is why the size to which we want to enlarge the detection was gradually increased until the optimal size was found.

The results of the best performing model are presented in the table below.

Table 9 Method 3 results with added resizing for text detections smaller than 24000 pixels

	CER %	WER %	Words Accuracy %	Character Accuracy %	Precision	Recall
Improved Method 3	27,31	55,43	44,57	72,69	0,50	0,47

The model achieves the accuracy of 72,69 for character recognition, and 44,57 for words recognition. Out of all textual labels in the images, 47% were correctly read, and out of all textual labels read by the OCR, 50% were read correctly.

4.4. Methods comparison

The results of all presented methods are summarized in Table 10, and the methods that gave a significant improvement in accuracy are highlighted. It is clear that with the improvement of method, the recognition accuracy increases. The character accuracy has been increased from the initial value of 34,97% to the final 72,69%, while the accuracy in recognizing entire words has been increased from the initial 13,78% to 44,57%. Significant progress has been made in recognition. Several examples of using the best preforming model on the test dataset are shown in the Figure 4.1.

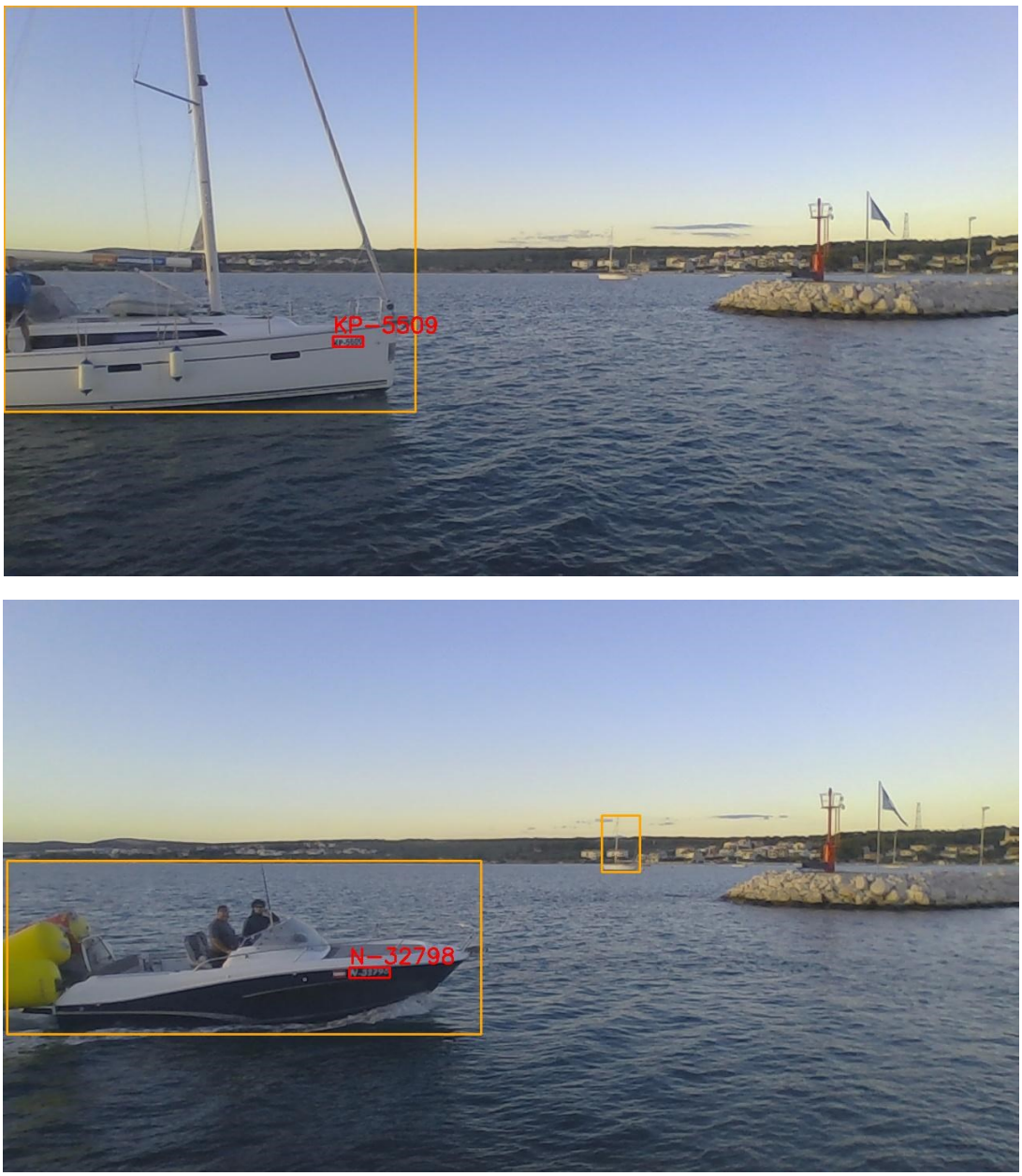

Figure 4.1 Examples of using the best preforming model

Table 10 Comparison of the results of all the presented methods

Method	Ship detection	Text detection	Text resizing (pixels)	Modification	OCR	CER %	WER%	Word Accuracy %	Character Accuracy %	Precision	Recall
1	No	Default	No	No	Paddle	78,95	86,22	13,78	21,05		
	No	Default	No	No	Easy	75,03	89,39	10,61	24,97		
2	**YOLOv8**	**Default**	**No**	**No**	**Paddle**	**54,99**	**73,83**	**26,17**	**45,01**		
	YOLOv8	Default	No	No	Easy	77,62	89,73	10,27	22,38		
3	**YOLOv8**	**YOLOv8**	**No**	**No**	**Paddle**	**37,68**	**68,91**	**31,09**	**62,32**	**0,35**	**0,33**
	YOLOv8	**YOLOv8**	**8000**	**No**	**Paddle**	**30,12**	**58,80**	**41,20**	**69,88**	**0,46**	**0,43**
	YOLOv8	YOLOv8	10000	No	Paddle	29,91	56,55	43,45	70,09	0,48	0,46
	YOLOv8	YOLOv8	12000	No	Paddle	28,25	55,43	44,57	71,75	0,50	0,47
	YOLOv8	**YOLOv8**	**24000**	**No**	**Paddle**	**27,31**	**55,43**	**44,57**	**72,69**	**0,50**	**0,47**
	YOLOv8	YOLOv8	48000	No	Paddle	27,53	55,81	44,19	72,47	0,49	0,46
	YOLOv8	YOLOv8	96000	No	Paddle	27,72	56,18	43,82	72,28	0,49	0,46
	YOLOv8	YOLOv8	24000	Sharpening	Paddle	28,25	55,06	44,94	71,75	0,50	0,47
	YOLOv8	YOLOv8	24000	Contrast, brightness	Paddle	27,31	55,43	44,57	72,69	0,49	0,45

Conclusion

The aim of this thesis was to develop an algorithm for the detection and recognition of vehicle license plates. Three main methods for detection and recognition are presented. Each of them used the OCR tool for detection, recognition or both phases. This proved to be a very good option given the nature of the dataset we had. The dataset used in this work does not contain many clearly readable license plates, so images with such plates were mostly used for testing. This is the reason why we were unable to train a network for character recognition on our dataset and instead had to use a pretrained model. For the detection task in some methods, YOLOv8m model (the latest version of YOLO) trained on the available dataset was used. This was a great choice considering the accuracy and speed of the YOLOv8 model. With an F1 score of 0,95 for the trained model, text detection has been raised to a very high level. The text recognition phase was performed using the default OCR algorithm for recognition, with enlargement of detected text to the size most readable by the OCR. The ratio of text enlargement was chosen by testing different detection sizes on the test dataset and comparing the results.

Finally, the best developed model achieved a word recognition accuracy of 44,57% and a character recognition accuracy of 72,69%. The model's progress is significant compared to the initial results.

Since the text detection is accurate enough, the only viable option for further improvement would be enhancing the recognition model. The Paddle OCR tool, used in this work, offers the option of fine tuning the default recognition model on our own dataset. However, as mentioned, this was not possible in this work due to the lack of data with clearly readable text.

Literature

[1] W. Zhang, H. Sun, J. Zhou, X. Liu, Z. Zhang and G. Min, "DCNN Based Real-Time Adaptive Ship License Plate Recognition (DRASLPR)," 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Halifax, NS, Canada, 2018, pp. 1829-1834

[2] Yash Shambharkar, Shailaja Salagrama, Kanhaiya Sharma, Om Mishra and Deepak Parashar, "An Automatic Framework for Number Plate Detection using OCR and Deep Learning Approach" International Journal of Advanced Computer Science and Applications (IJACSA), 14(4), 2023.

[3] Timalsina A., Anysis and Benchmarking of OCR Accurcy for Data Extraction Models, (2023., September). Link: https://www.docsumo.com/blog/ocr-accuracy; accessed September 15, 2023.

[4] Clemens Neudecker, Konstantin Baierer, Mike Gerber, Christian Clausner, Apostolos Antonacopoulos, and Stefan Pletschacher, A survey of OCR evaluation tools and metrics, In Proceedings of the 6th International Workshop on Historical Document Imaging and Processing (HIP '21), New York, USA, (2021.)

[5] Petkovic, M.; Vujovic, I.; Lusic, Z.; Soda, J. Image Dataset for Neural Network Performance Estimation with Application to Maritime Ports. J. Mar. Sci. Eng. 2023, 11, 578.

[6] Ufuk Dag, Comparison of Paddle OCR, Easy OCR, Keras OCR, and Tesseract OCR, Link: https://www.plugger.ai/blog/comparison-of-paddle-ocr-easyocr-kerasocr-and-tesseract-ocr, accessed November 6, 2023.

[7] Christian Weiler, OCR and deep OCR in comparison, Link: https://www.bludelta.de/en/ocr-and-deepocr-in-comparison/, accessed November 21, 2023.

[8] Kiefer B., Žust L., Kristan M., Perš J., Teršek M., Wiliem A., Messmer M., Yang C., Huang H., Jiang Z., Kuo H., Mei J., Hwang J., Stadler D., Sommer L., Huang K., Zheng A., Chong W., Lertniphonphan K., Xie J., Chen F., Li J., Wang Z., Zedda L., Loddo A., Di Ruberto C., Vu T., Nguyen-Truong H., Ha T., Pham Q., Yeung S., Feng Y., Thanh Thien N., Tian L., Michel A., Gross W., Weinmann M., Carrillo-Perez B., Klein A., Alex A., Solano-Carrillo E., Steiniger Y., Bueno Rodriguez A., Kuan S, Ho Y., Sattler F., Fabijanić M., Šimunec M., Kapetanović N.; Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops, 2024, pp. 869-891

[9] Ultralytics, YOLOv8 docs documentation, (2023.), Link:https://docs.ultralytics.com/guides/yolo-performance-metrics/ , Accessed: January 2024.

[10] Ultralytics, YOLOv8 docs documentation, (2023.), Link: https://docs.ultralytics.com/yolov5/tutorials/architecture_description/#2-data-augmentation-techniques, Accessed: January 2024.

[11] Deci Algorithms team, YOLOv8 vs. YOLO-NAS showdown: Exploring advanced object detection , (2023.), Link: https://deci.ai/blog/yolov8-vs-yolo-nas-showdown-exploring-advanced-object-detection/, Accessed: January 2024.

[12] Redmon J., Divvala S., Girshick R., Farhadi A., You Only Look Once Unified, Real-Time Object Detection, University of Washington, Allen Institute for AI, Facebook AI Research

[13] Krishnakumar M., A gentle introduction to YOLOv8, (2023.), Link: https://wandb.ai/mukilan/wildlife-yolov8/reports/A-Gentle-Introduction-to-YOLOv8--Vmlldzo0MDU5NDA2, Accessed: January 2023.

Summary

Accurate ship identification is vital for global trade and maritime security, especially in smart port initiatives. Ship license plate recognition helps prevent illegal activities and ensures efficient transportation of goods. Two of the main phases in recognizing ship license plates are license plate detection and license plate recognition. In this thesis, three methods were developed for the mentioned application. All three methods us Paddle OCR tool for text recognition, while the text detection models differ in each method. In the first method, the Paddle OCR default detection and recognition algorithms are used on the original image. In the second method, the ship is first detected using YOLOv8 available detection model, and then the image is sent to the Paddle OCR tool, which again uses default algorithms for text detection and recognition. The third method also uses the YOLOv8 available model for ship detection, for text detection it uses the YOLOv8 model developed in this work, and finally the detected text is sent to Paddle OCR for recognition only. The third method was further improved by the process of enlargement the text detection to the optimal size, the one most easily readable to the OCR. The last described method also gives the best results on the test dataset. The final model achieves an accuracy of 44,57% in words detection and 72,69% in character detection.

Key Words:

ship;license plate recognition;text recognition;text detection;detector;character recognition;deep learning;ship license plate recognition;detection model;optical character recognition

Sažetak

Identifikacija brodova ključna je u područjima globalne trgovine, pomorske sigurnosti, a posebno u razvoju pametnih luka. Prepoznavanje registarskih oznaka brodova pomaže sprječavanju ilegalnih aktivnosti i osigurava učinkovit prijevoz robe. Dvije glavne faze u prepoznavanju registarskih oznaka brodova su detekcija registarske oznake i prepoznavanje registarske oznake. U ovom radu razvijene su tri metode za navedenu primjenu. Sve tri razvijene metode za prepoznavanje teksta koriste Paddle OCR alat, dok se modeli za detekciju teksta u svakoj metodi razlikuju. U prvoj metodi na ulaz dovodimo originalnu sliku i na njoj koristimo algoritme za detekciju i prepoznavanje teksta koje uobičajeno koristi Paddle OCR. U drugoj metodi, najprije se provodi detekcija broda pomoću YOLOv8 dostupnog modela za detekciju, te se nakon toga slika šalje Paddle OCR alatu koji na njoj opet koristi uobičajene algoritme za detekciju i prepoznavanje teksta. Treća metoda za detekciju brodova također koristi YOLOv8 dostupan model, za detekciju teksta koristi YOLOv8 model razvijen u ovom radu, te se na kraju detektirani tekst šalje Paddle OCRu samo na prepoznavanje. Treća metoda dodatno je poboljšana postupkom povećanja detekcije teksta na optimalnu veličinu, onu najlakše čitljivu OCRu. Posljednja opisana metoda daje i najbolje rezultate na testnom skupu podataka. Konačni model postiže točnost od 44,57% u prepoznavanju riječi i 72,69% u prepoznavanju znakova.

Ključne riječi :

brod;registarska oznaka;prepoznavanje teksta;detekcija tekst;detector;prepoznavanje znakova;duboko učenje;prepoznavanje registarske oznake;model za detekciju;optičko prepoznavanje znakova

Attachment

Along with this paper, a Jupyter notebook file is provided for testing the best developed model on a custom dataset.

Software instructions

1. Install Jupyter Notebook on your computer
2. Open the provided .ipynb file using Jupyter Notebook
3. Set the variable `img_path` to the path of the folder containing the images you want to test
4. Set the variables `ship_det_model` and `text_det_model` models to the paths of the provided models
5. Run the commands in the notebook