Efficient Object Detection in Railway Environments using YOLOv8

Artur Dylewski
s26752277
a.dylewski@student.utwente.nl

Jakub Kościołek
s2806002
j.m.kosciolek@student.utwente.nl

Arda Akyazı
s2515091
a.akyazi@student.utwente.nl

Abstract

This paper presents a comprehensive exploration of a highly efficient railway track sign detection system. With a specific focus on the Dutch railway network, the project's goal is to develop a model capable of precise identification and classification of various train track signs, including kilometer markers, traffic lights, stop indicators, emergency exit signs, and danger warnings. These objectives align with the broader initiative of digitizing the Dutch railway infrastructure to enhance safety and operational efficiency. The project leverages state-of-the-art techniques, including data augmentation, YOLOv8 models, and advanced inference methods, to achieve remarkable results. This work contributes to the ongoing modernization and safety improvement of railway networks.

Defined goal

Our project's defined goal is to develop a highly efficient and precise railway track sign detection system. Specifically, we aim to create a model capable of accurately identifying and classifying train track signs, including kilometer signs, traffic lights, stop indicators, emergency exit signs, and danger warnings. In addition to detecting said objects, pinpointing the location proves to be a useful tool for displaying the information in advance. This system will play a pivotal role in improving safety and operational efficiency within the Dutch railway network.

I. Motivation

The motivation driving our project stems from the critical need to enhance railway safety and efficiency in the Netherlands. Accurate and real-time detection of train track signs is essential for preventing accidents, ensuring smooth railway operations, and safeguarding the lives of passengers and personnel. By developing a cutting-edge detection system, we
aim to contribute to the broader initiative of digitizing the Dutch railway infrastructure, ultimately leading to a safer, more streamlined, and technologically advanced railway network.

II. Method

Images

We have acquired a substantial dataset from Spoorinbeeld, specifically focusing on oblique images. These images, in tif format, come with associated location and camera rotation data, providing crucial context for the pictures. The images are notably large, with dimensions of 12,000 by 14,000 pixels. In total, we've downloaded approximately 300 of these images, amounting to around 30 gigabytes of data. This extensive dataset is a valuable resource for our project.

Sign dataset

Our thorough analysis of the acquired images has led us to identify several recurrent train track signs, providing us with sufficient data to effectively train our algorithms. These are the train track signs that we have observed:

- km_sign: These signs serve to indicate the location of the train by kilometer (figure 5).
- traffic_light: Traffic lights that are present alongside the train tracks (figure 8)
- stop_indicator: Signs situated at stations that guide trains to halt at precise locations (figure 4).
- emergency_exit: Signs directing passengers to emergency exits along the train tracks (figure 7).
- danger: Signs that serve as warnings of potential hazards or dangers ahead (figure 6).

By identifying and categorizing these recurring signs, we are well-equipped to facilitate effective training.

Labeling

Labeling the images was a meticulous process that required our own manual efforts. We have used the CVAT (https://www.cvat.ai) web app, which was recommended by our supervisor. Overall we uploaded 265 images, for which we made 2 sets: one for labeling train tracks on full images, the other for labeling signs on cutted images. After reviewing all the images, we identified approximately 190 images that contained various signs.

Augmentation

To enhance our model's robustness and reliability across various environments and scenarios, we've implemented data augmentation techniques, ensuring its adaptability and consistent performance.

Optimizing

To optimize the learning times we decided to create another model that could properly detect the train tracks itself, to train in we also decided to scale the images drastically down to again reduce the learning time after they were found they were cropped so the sign recognition algorithm was not learning and searching for the sign in parts of the image that are completely not relevant for the assignment

Model

We've opted for YOLOv8 as our object detection framework. YOLOv8 is known for its cutting-edge speed and efficiency, making it a modern standard in this field. Its wide adoption in the computer vision community ensures we have ample resources, help threads, and additional technologies at our disposal. This choice grants us flexibility, as YOLO models are versatile and can handle various object detection tasks effectively. With YOLOv8, we have a state-of-the-art, well-supported, and highly efficient solution for our project.

Triangulation

In the geolocation calculation process, we utilized triangulation to determine the precise coordinates of detected objects. This technique involved assessing the spatial positions of objects based on information obtained from two distinct camera viewpoints. To execute triangulation, we computed the distance of the object from each camera by analyzing the content of the respective images. Subsequently, combining these distances with the known positions and orientations of the cameras enabled us to triangulate the spatial coordinates of the detected objects. Notably, the geolocation results were presented in accordance with the National Triangle coordinates of the Netherlands (Rijksdriehoekscoördinaten) (Wikipedia contributors, 2023).

OCR

During implementing signs classification, we have realized that they are certain signs that include additional information that can be utilized. Examples of such signs are train stop indicators (figure 4) and kilometer location signs (figure 5). Thus we investigated various OCR techniques to see which will suit our case the best. After careful consideration, we came to the conclusion that OCR techniques work perfectly for stop indicators, however they vastly underperform on kilometer location signs, which are usually of poor quality. However, this did not stop us from utilizing the models. We understood that for the kilometer location signs it does not return anything, if the sign has been misrecognized, which served as an additional assurance for our recognition model.

III. Technical depth

Augmentations

To enhance rigidity and overall performance, as well as to create a more versatile and high-performing dataset, we've implemented the following augmentations:

- Brightness Adjustment: We've applied both positive and negative adjustments, with values of +0.5 and -0.5. This augmentation diversifies the dataset by accounting for varying lighting conditions.
- Contrast Adjustment: Similar to brightness, we've made positive and negative contrast adjustments, with values of +0.5 and -0.5. This helps the model adapt to images with differing levels of contrast.
- Gaussian Blur: We've introduced Gaussian blur with varying radii of 1, 2, and 3. This augmentation simulates different levels of image blur and sharpness, preparing the model for real-world variations in image quality.
- Image Reflection: We've reflected each image around the y-axis, to ensure that the model will correctly recognize the signs that may be on the other side of the train tracks (figure 15).

By incorporating these augmentations into the dataset, we have significantly improved its resilience, adaptability, and performance, making it a more universally applicable resource for our project.

Railway detection model

The model was trained using original images scaled down to 20% of their size, as large train tracks don't require high detail for accurate detection. We employed the YOLOv8 pretrained model, specifically YOLOv8n.pt (nano), chosen for its quick training times and sufficient precision, aligning with the task's requirements.

The training process was streamlined for time efficiency, achieving satisfactory results over 25 epochs. Notably, the model demonstrated minimal misclassifications, revealed during empirical evaluation due to low confidence thresholds.

In Figure 1 of the appendix, we present the model's predictions. To retain valuable information about train tracks, a cropping method with a 5% padding was implemented. This ensures predictions maintain the image's complete context, leading to file reductions of around 30-40%, dependent on the image.

Overall, this model successfully balances efficiency and effectiveness in detection, making it well-suited for our project.

Sign detection model

Using YOLOv8 with the labels mentioned previously, we have developed a model that could reliably recognize train signs, as selected by us.

The model training process can be divided into three key phases:

1. Initial Training: In this stage, we tested the feasibility of the technology, specifically YOLO and SAHI (Akyon, 2022) , to meet our requirements. We started with a smaller dataset and set the training image size (imgsz) to 640 to expedite the training process. We conducted a relatively low number of epochs (40). While the model's performance was initially unreliable and imprecise, it demonstrated the potential for our technology and approach. It successfully detected some objects, indicating we were on the right track.

2. Full Dataset Training: Building on the insights gained from the initial training, we decided to use our complete labeled dataset. We made adjustments to improve the learning process by increasing the imgsz parameter to 1024 and optimizing the learning hyperparameters. We also increased the batch size to leverage the GPU's tensor cores. To achieve better performance, we extended the training to 100 epochs, resulting in a total of 140 epochs. However, the model's performance was still suboptimal (TODO metrics) and exhibits a significant number of false positives, particularly when it comes to recognizing km_signs. To address these issues, we determined that more extensive data augmentation is necessary to eliminate this behavior.

3. Extended Training with Augmented Data: To tackle the challenges posed by our large dataset, we continued training the model with augmented data. Due to the computational resources required for this task, we limited this phase to 25 epochs. As seen in Fig. 3 in the appendix it is clear that the model improved quite significantly aspeecialy in the higher precision mAP (75-90) metrics, with the max being 0.53863 and mAP50 to a high 0.86443, indicating that the model handles the detection reasonably well, but has bigger issues detecting the exact small signs but that again can be treated as characteristic of different labeling styles by members of the team along with the very low pixel amount in the image Nevertheless, the model's performance improved but was not safe from making some mistakes, thats why in further paragraph we discuss further validation of km_signs with OCR algorithm.

In conclusion, while we have made significant progress, there is still work to be done to improve the model's performance.

Prediction using SAHI

To enhance our model's predictions, we adopted the SAHI (Slice Aided Hyper Inference) method. It's particularly effective for our task, which involves detecting small objects like train track signs. SAHI's advantage lies in its precision, ensuring accurate identification and classification of these small elements within images.

The system archives these results by partitioning images into predetermined smaller chunks with a specified degree of overlap. This method allows the algorithm to process a reduced amount of information at each instance, enabling focused analysis of specific image regions. Furthermore, SAHI's ability to focus on specific portions of an image minimizes the risk of missing or misclassifying these small objects, making our model more accurate in its predictions.

OCR

Due to the time constraint and relatively low priority of this task, we decided to search for solutions that will provide us with an already rigid and easy to use model that won't require any additional training and preprocessing from us. We have considered 3 options:

Keras-OCR (https://keras-ocr.readthedocs.io/en/latest/)
This option is advantageous for its seamless integration with existing deep learning workflows through Keras and TensorFlow. It allows OCR model customization and is supported by robust documentation and a community. However, its drawback lied in the complexity of the implementation, which would make it vastly time consuming for us to notice decent changes

pytesseract (https://github.com/tesseract-ocr/tesseract)
This option is notable for its use of Tesseract, a widely adopted OCR engine trained on diverse data. It is user-friendly, easy to set up, and open source, supported by a strong community. However, it may not match the accuracy of custom-trained deep learning models for specific tasks, and its performance can vary based on text complexity and image quality.
We knew that our sign images may have been of poor quality, which made us question the usage of this option.

EasyOCR (https://github.com/JaidedAI/EasyOCR)
Finally this option is highlighted by its claim to be user-friendly, offering support for multiple languages and various pre-trained models. In addition, it is widely used for recognizing numbers on signs and car plates. However, its performance and accuracy might not reach the level of custom-trained models for specific tasks.

After a couple of tests, we came to the conclusion that EasyOCR is the most beneficial for us and checked all of our criterias for this task. The only preprocessing we needed to apply for the image was increasing its size. The model Unfortunately, the model was underperforming for the kilometer location signs, which were of poor quality (figure 9). Nevertheless, while testing, we have come up with additional functionality for which we can utilize the easyOCR model. Namely, it wasn't returning anything for the kilometer location signs that were misdetected by our main model (figure 10). Thus we incorporated additional checks, which ensured that easyOCR is providing any result, which filters the misdetected kilometer location signs.

Triangulation

In the pursuit of geolocating objects based on a pair of images, the determination of the spatial relationship between bounding boxes in these images poses a significant challenge. The mathematical framework employed in this context involves the identification of key points and descriptors within each image, subsequently utilizing a feature detection and matching algorithm. For this purpose, the Oriented FAST and Rotated BRIEF (ORB) feature detector (as opposed to SIFT (*OpenCV: Introduction to SIFT (Scale-Invariant Feature Transform)*, z.d.), for less computational strain), coupled with the BruteForceMatcher, was employed to discern salient features.

The matching of bounding boxes between the images relies on the computation of a homography matrix through the Random Sample Consensus (RANSAC) algorithm (Wikipedia contributors, 2023). This matrix captures the linear transformation necessary to align corresponding points in both images accurately. The application of the homography matrix facilitates the estimation of the pixel coordinates in one image with respect to the other, thereby establishing a mapping between bounding boxes. However, challenges arose when detecting new bounding boxes of the same data type in subsequent images. Achieving an effective mapping in these scenarios proved to be a complex task. The necessity of establishing a one-to-one mapping further compounded the difficulty, demanding a meticulous approach to eliminate inaccurate pairings.

In our mapping process, we initially matched each bounding box in Image 2 with the one estimated from Image 1, considering both spatial proximity and label correspondence. While this approach facilitated the mapping task, it resulted in extraneous mappings that needed to be addressed. To resolve this issue, we implemented a refinement step. By evaluating the distances between potential matches and the bounding boxes in Image 2, we were able to identify and eliminate inaccurate mappings, ensuring a more accurate and reliable one-to-one mapping between corresponding

performed marvelously on the train stop indicators, that we made our main priority. In addition, the methods allowed us to specify what characters should be only available for the model to use, which allowed us to specify that we are interested only in numbers.
good now?
bounding boxes in the two images. This step played a crucial role in enhancing the precision and reliability of our overall mapping process.

With the bounding boxes successfully matched, the subsequent step involves the calculation of geolocation for the detected objects. This requires knowledge of the camera locations and orientations during image capture. The pertinent camera matrices are constructed using this intrinsic camera information, allowing for the transformation of pixel coordinates into real-world coordinates.

This comprehensive methodology, leveraging computer vision techniques and geometric transformations, enables the accurate spatial registration of objects across multiple images and contributes to the overarching goal of geolocation.

IV. Results

In this section, we delve into a comprehensive assessment of the overall model performance. This discussion necessitates an examination of distinct facets of the pipeline, thereby categorizing the evaluation into the train track model, sign detection, OCR performance, and triangulation error. A succinct summary follows, addressing the pivotal question of whether this pipeline attains a level of maturity enabling its practical utilization in any form.

The Railway Detection

The assessment metrics applied to the Railway Detection Model offer valuable insights into its performance. Notably, at an IoU threshold of 50%, the model achieves a robust Mean Average Precision (mAP) of 0.82711, attesting to its accuracy in identifying and localizing objects within railway images. Moreover, maintaining a commendable mAP of 0.6401 across IoU thresholds from 50% to 95% underscores the model's precision in object detection amid varying degrees of overlap. Importantly, given the inconsistencies in labeling by our team members and the nuanced definition of the train track area, the empirical review test becomes crucial. In this context, the model not only passes the test effectively but also exhibits promising indications through predictions and their associated confidences, suggesting its potential in reducing dataset size and enhancing operational efficiency.

Sign Detection

In our evaluation of the object detection model, the metrics mAP50 (mean Average Precision at 50) and mAP50-95 (mean Average Precision between IoU 50% and 95%) have provided valuable insights into the model's performance across different classes.

The mAP50 analysis revealed notable achievements, with the "traffic_light" class attaining the highest score of 0.995, showcasing exceptional proficiency in object detection and localization. The overall model performance is robust, exemplified by a substantial mAP50 of 0.858 for the average. Additionally, classes like "emergency_exit" and "km_signs" displayed commendable mAP50 values, indicating effective detection capabilities.
Turning to the mAP50-95 observations, the "traffic_light" class continued to excel with a high mAP50-95 of 0.839, emphasizing reliable localization across a spectrum of IoU thresholds. The average maintained consistency with a mAP50-95 of 0.539, indicating reliable detection and localization across all classes and IoU thresholds.

In a broader context, the model demonstrated proficiency in object detection, as evident from high mAP50 scores across diverse classes. As seen in Fig.11 the mAP50 across all classes gives a satisfactory amount of precision.

Classes with lower mAP values, specifically "danger" and "km_signs," warrant further investigation for potential improvements, from empirical reviews of the detections and labeled objects from this classes we can see that certain images, of those objects can be as small as 15x15 pixels as seen in fig.12, therefore we believe that more high quality data would be beneficial for the models performance.

In conclusion, this results highlights the model's effectiveness while identifying areas for refinement, emphasizing the importance of continuous evaluation and optimization for sustained and improved performance.

Triangulation

Remarkably, our outcomes have demonstrated a high degree of accuracy, yielding spatial coordinates that closely approximate the actual points on the map. This commendable precision can be largely attributed to the efficacy of our computational processes. However, it is worth noting that the deviations observed might be a result of rounding the numbers during the coordinate calculation phase.

OCR

In evaluating OCR methods, we meticulously reviewed all predictions related to stop indicators, comprising approximately 30 instances across the entire dataset. Notably, all predictions were accurate (figure 13).

In the case of kilometer location signs, a similar manual assessment was conducted on three existing errors in our kilometer sign predictions. Importantly, for each misidentification, the OCR method appropriately failed to recognize any numerical values associated with the given signs.

V. Conclusions

In conclusion, this paper outlines a comprehensive exploration of a highly efficient railway track sign detection system, with a specific focus on the Dutch railway network. Through the application of state-of-the-art techniques, including YOLOv8 models, data augmentation, and advanced inference methods, the developed system demonstrates remarkable results in identifying and classifying various train track signs. The project aligns with the broader initiative of digitizing the Dutch railway infrastructure to enhance safety and operational efficiency.

The chosen YOLOv8 framework proves to be efficient, providing a balance between speed and precision in object detection. The incorporation of OCR techniques enhances the system's capabilities, particularly in recognizing additional information on signs. The utilization of the SAHI method and the selection of EasyOCR for sign classification illustrate a pragmatic approach to address challenges and optimize model predictions. The triangulation process, employing computer vision techniques, contributes significantly to accurate spatial registration of objects across multiple images.

Evaluation metrics reveal the system's effectiveness in railway detection, sign detection, and OCR, with identified areas for refinement. Continuous evaluation and optimization are emphasized to ensure sustained and improved performance, underscoring the importance of ongoing development in this domain.

VI. Critical reflection

Critically reflecting on the development of the railway track sign detection system, several key aspects warrant consideration. The integration of cutting-edge technologies, such as YOLOv8 models and advanced inference methods, undoubtedly showcases the project's commitment to innovation. The strategic use of data augmentation and the meticulous acquisition of a substantial dataset from Spoorinbeeld underscore a methodical approach to system robustness.

However, a critical lens reveals certain challenges and areas for improvement. The model's performance, especially in the detection of smaller signs and specific classes like danger and kilometer location signs, raises concerns. The need for more high-quality data, especially for these challenging classes, becomes evident. The decision to scale down images for efficiency in learning times is understandable, yet it also introduces potential limitations in capturing intricate details, particularly in smaller signs.

The incorporation of OCR techniques presents a practical solution for extracting additional information from signs. However, lack of proper number recognition for kilometer location signs are noticeable, which can be enhanced by investing time in sign numbers labeling and model training to achieve proper number recognition results.

Fundamental and Essential matrices play a crucial role in the triangulation process, ensuring a robust and accurate method for projecting points between images. As fundamental tools in computer vision, they provide a reliable framework for triangulation. Looking forward, exploring and experimenting with these matrices in future work holds promise for refining and advancing our geolocation methodologies. The versatility of these concepts presents exciting opportunities to enhance the precision of triangulation in various computer vision applications. This forward-looking approach aligns with our commitment to continuous improvement and innovation in railway track sign detection.

VII. Ethics

Notably, the images provided to us have deliberately blurred out the faces of any humans present. Therefore the identities of these individuals were already protected.

Additionally, with regard to the object detection model, a conscientious approach was adopted in labeling. Notably, only the necessary traffic signs were labeled, and extra care was taken to abstain from labeling any humans. This intentional omission reflects a commitment to ethical practices, focusing the model's attention solely on the relevant elements without encroaching on privacy boundaries.

APPENDIX

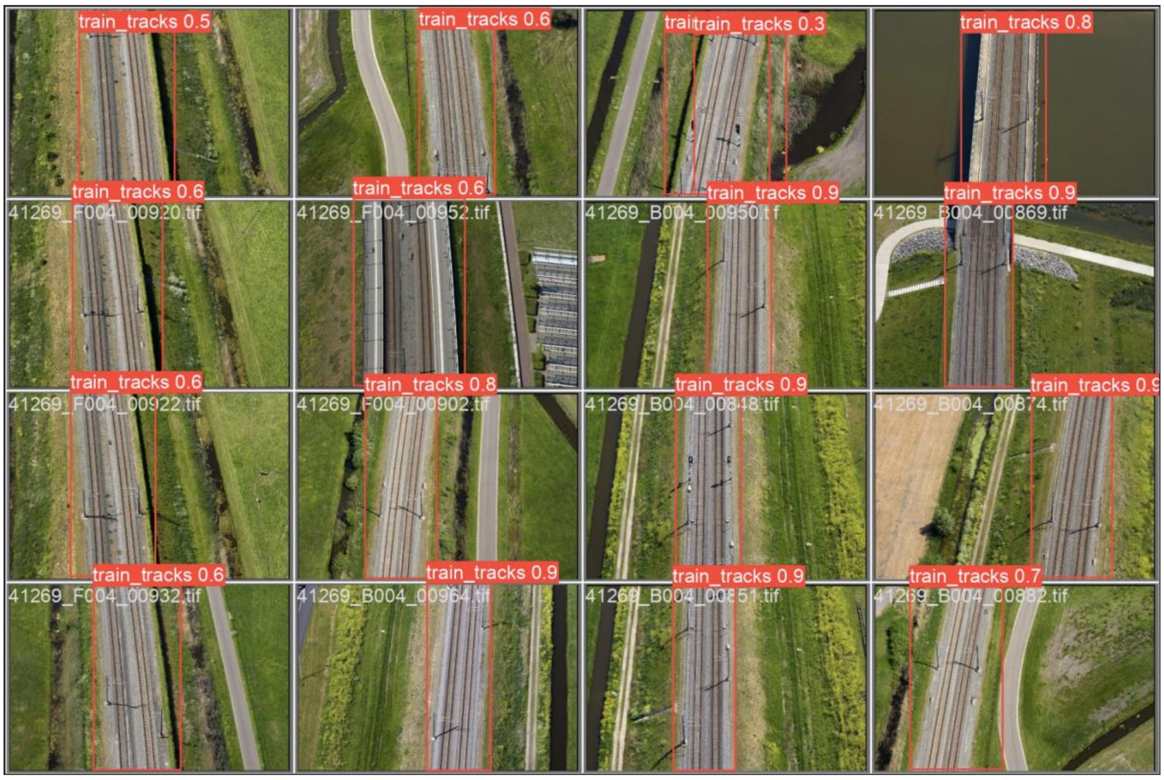


Fig. 1 - Predictions of the algorithm used for image cropping

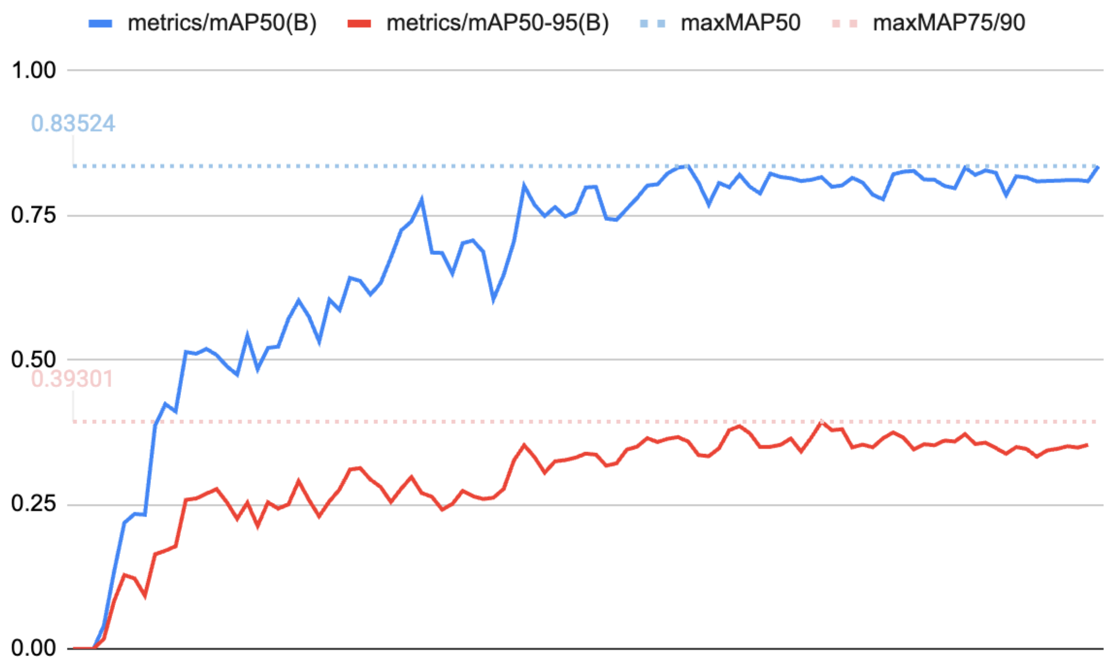


Fig. 2 - mAP metrics of model before training on augmented dataset

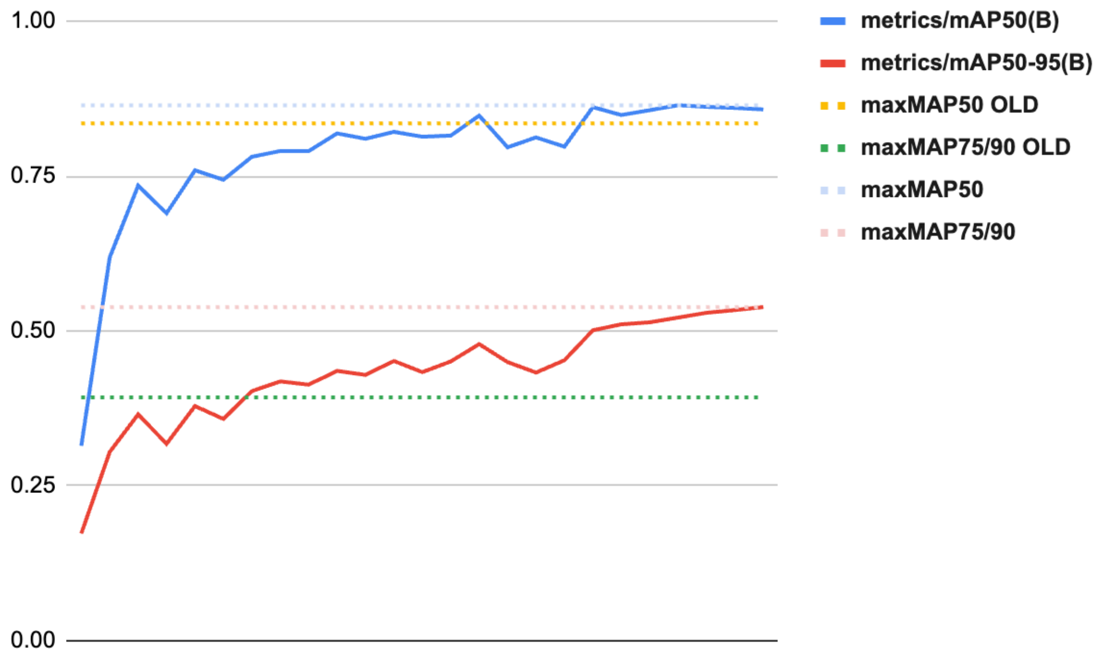


Fig. 3 - mAP metrics of model on augmented dataset

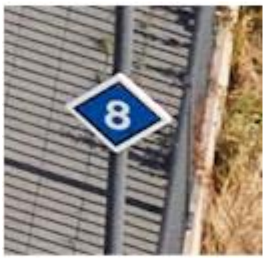


Fig 4 - Example of train stop indicator

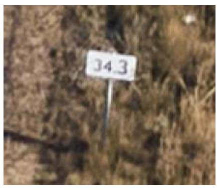


Fig 5 - Example of kilometer location sign

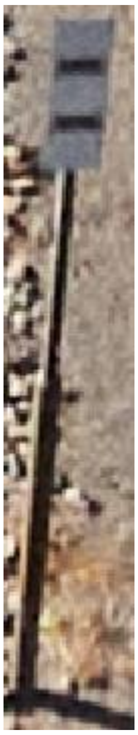


Fig 6 - Example of indication of special danger point behind this light

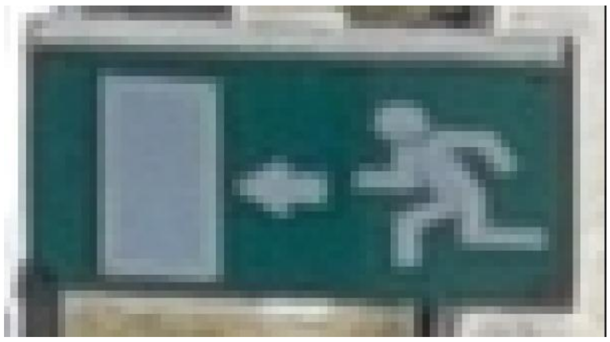


Fig 7 - Example of emergency sign

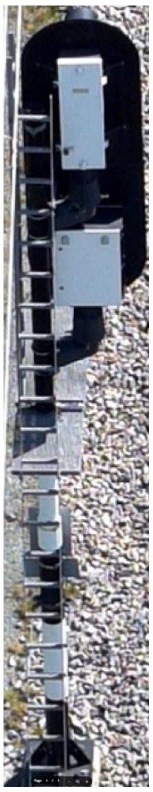
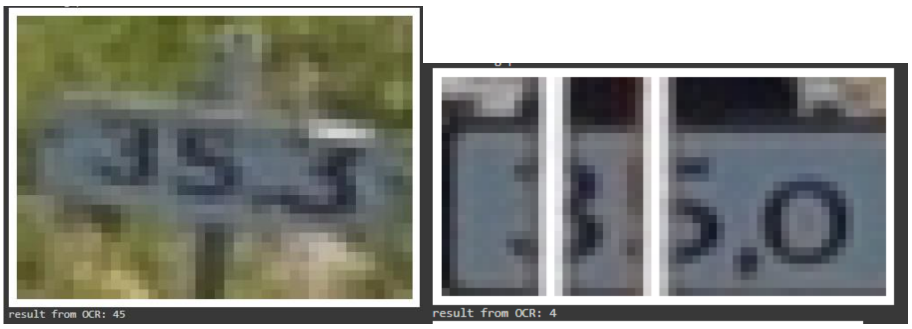
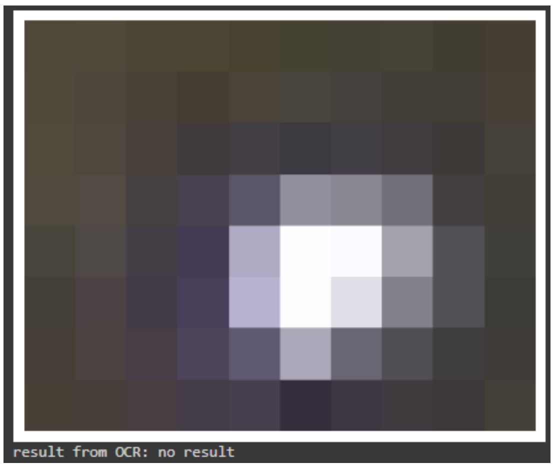


Fig 8 - Example of traffic lights


```
result from OCR: 45
result from OCR: 4
```
Fig 9 - Example of easyOCR model, underperforming for low quality kilometer location signs



```
result from OCR: no result
```

Fig 10 - Result from easyOCR model for the misdetected kilometer location sign

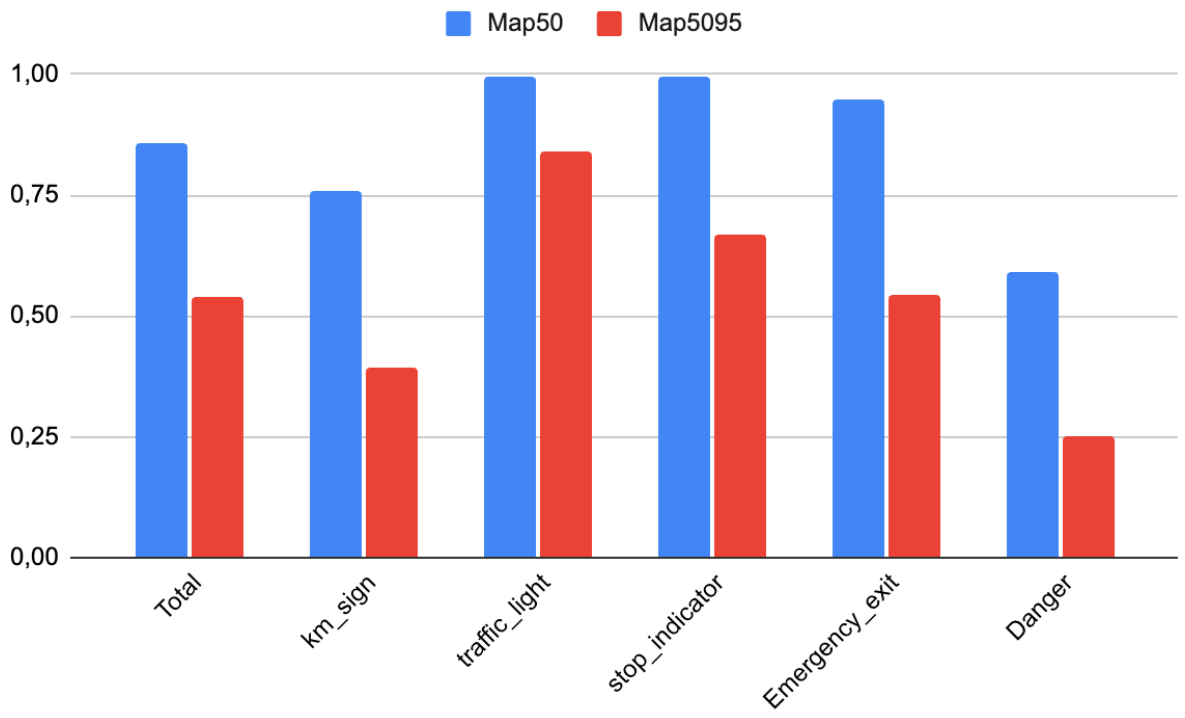
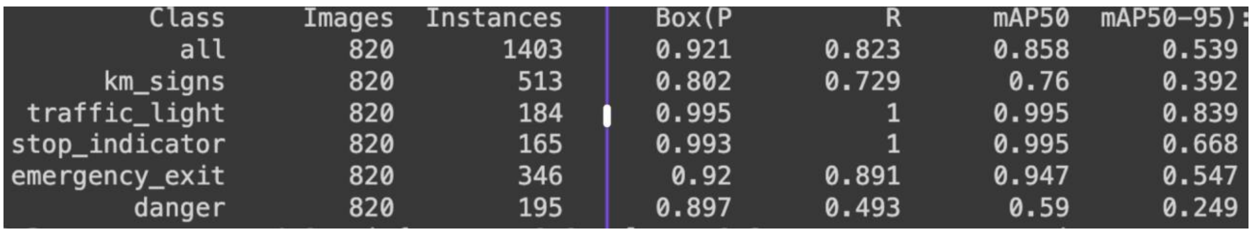


Fig. 11 mAP metrics for the final model;



```
          Class     Images  Instances      Box(P          R      mAP50  mAP50-95):
            all        820       1403      0.921      0.823      0.858      0.539
       km_signs        820        513      0.802      0.729       0.76      0.392
  traffic_light        820        184      0.995          1      0.995      0.839
 stop_indicator        820        165      0.993          1      0.995      0.668
 emergency_exit        820        346       0.92      0.891      0.947      0.547
         danger        820        195      0.897      0.493       0.59      0.249
```

Fig.12 All metrics for all classes

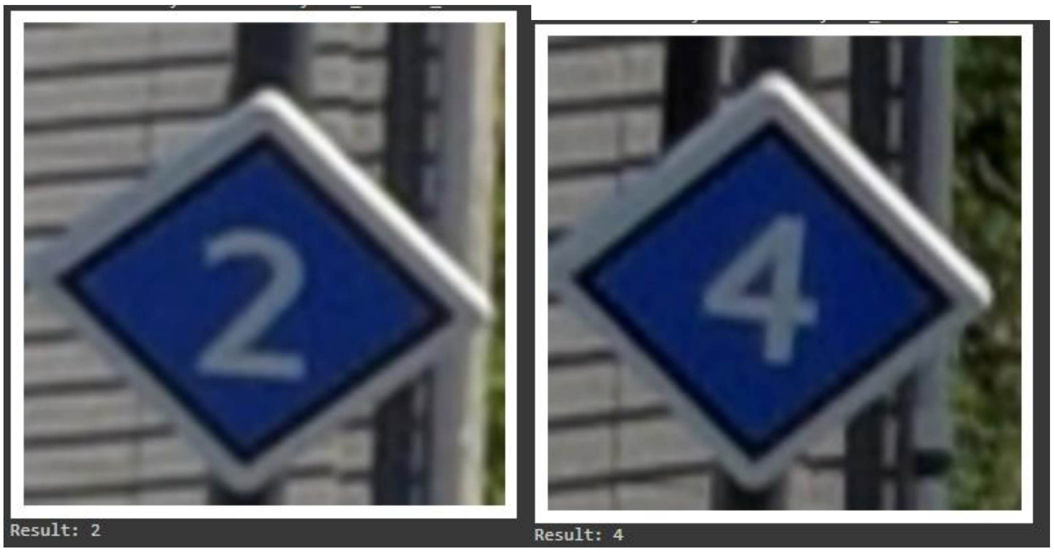


Figure 13 - Examples of results from OCR applied on stop indicators

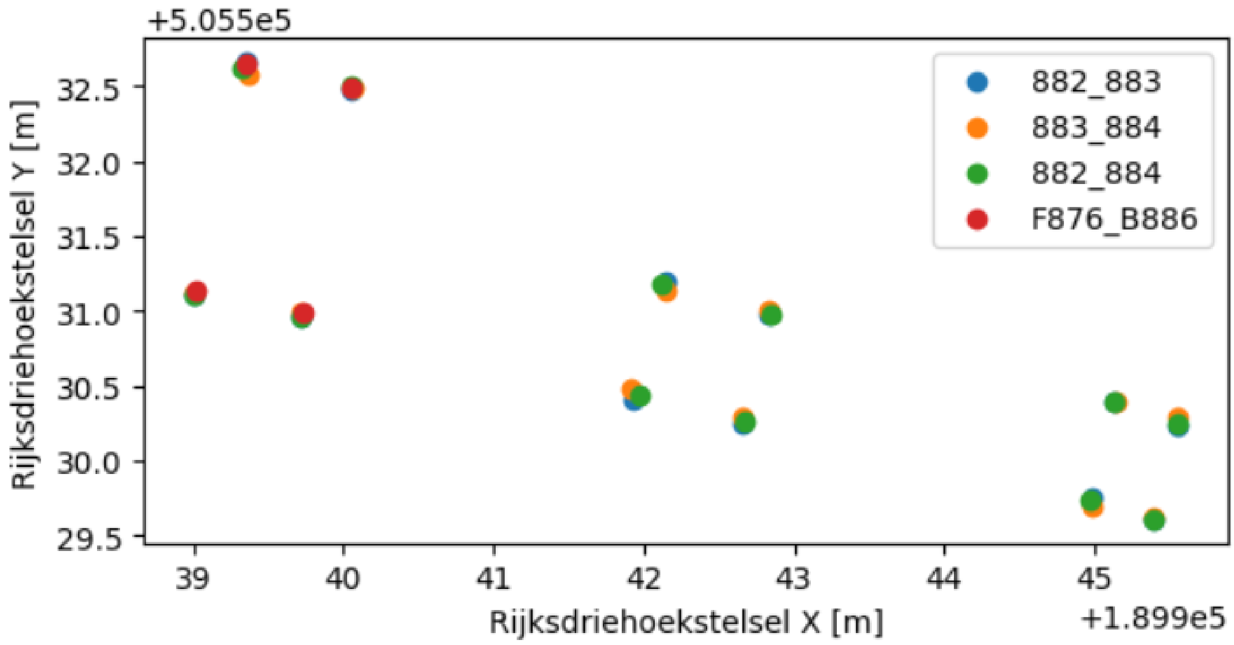


Figure 14 - Location of series of objects from different pairs of images

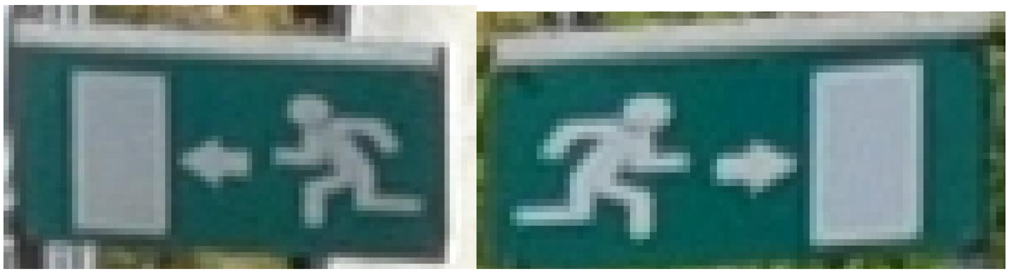


Fig 15 - Example of signs for which there was necessity for reflecting the images by y-axis

References

National triangle coordinates. (2023, October 2). Wikipedia, the free encyclopedia. Retrieved November 10, 2023, from https://nl.wikipedia.org/w/index.php?title=Rijksdriehoeksco%C3%B6rdinaten&oldid=66081956

Keras-OCR documentation. (n.d.). Retrieved from https://keras-ocr.readthedocs.io/en/latest/

pytesseract documentation. (n.d.). Retrieved from https://github.com/tesseract-ocr/tesseract

EasyOCR documentation. (n.d.). Retrieved from https://github.com/JaidedAI/EasyOCR

Application used for labeling the train tracks and signs. (n.d.). Retrieved from https://www.cvat.ai

OpenCV: Introduction to SIFT (Scale-Invariant Feature Transform). (n.d.). Retrieved from https://docs.opencv.org/4.x/da/df5/tutorial_py_sift_intro.html

Slice aided hyper inference (SAHI). (n.d.). Retrieved from https://github.com/obss/sahi/tree/main

Akyon, F. C. (2022, November 13). Sahi: A Vision Library for performing sliced inference on large images/small objects. Medium. https://medium.com/codable/sahi-a-vision-library-for-performing-sliced-inference-on-large-images-small-objects-c8b086af3b80

Yolov8 custom training. (n.d.). Retrieved from https://github.com/deepakat002/yolov8/tree/main

Yolov8 documentation. (n.d.). Retrieved from https://docs.ultralytics.com/

Wikipedia contributors. (2023, 26 oktober). Random sample consensus. Wikipedia.

https://en.wikipedia.org/wiki/Random_sample_consensus#:~:text=Random%20sample%20consensus%20(

RANSAC)%20is,the%20values%20of%20the%20estimates.