



FACULTY OF APPLIED SCIENCES
UNIVERSITY
OF WEST BOHEMIA

DEPARTMENT OF
COMPUTER SCIENCE
AND ENGINEERING

Master's Thesis

Computer Vision Applications in Video Recordings for Traffic Signal Detection and Classification on Czech Railways

Daniel Schnurpfeil



**FACULTY OF APPLIED SCIENCES
UNIVERSITY
OF WEST BOHEMIA**

**DEPARTMENT OF
COMPUTER SCIENCE
AND ENGINEERING**

Master's Thesis

Computer Vision Applications in Video Recordings for Traffic Signal Detection and Classification on Czech Railways

Bc. Daniel Schnurpfeil

Thesis advisor

Ing. Pavel Mautner, Ph.D.

© 2025 Daniel Schnurpfeil.

All rights reserved. No part of this document may be reproduced or transmitted in any form by any means, electronic or mechanical including photocopying, recording or by any information storage and retrieval system, without permission from the copyright holder(s) in writing.

Citation in the bibliography/reference list:

SCHNURPFEIL, Daniel. *Computer Vision Applications in Video Recordings for Traffic Signal Detection and Classification on Czech Railways*. Pilsen, Czech Republic, 2025. Master's Thesis. University of West Bohemia, Faculty of Applied Sciences, Department of Computer Science and Engineering. Thesis advisor Ing. Pavel Mautner, Ph.D.

ZÁPADOČESKÁ UNIVERZITA V PLZNI

Fakulta aplikovaných věd

Akademický rok: 2024/2025

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení: **Bc. Daniel SCHNURPFEL**
Osobní číslo: **A22N0074P**
Studijní program: **N0613A140037 Informatika a její specializace**
Specializace: **Zpracování přirozeného jazyka**
Téma práce: **Využití metod počítačového vidění ve videozáznamech pro detekci a klasifikaci návěstidel na českých železnicích**
Zadávací katedra: **Katedra informatiky a výpočetní techniky**

Zásady pro vypracování

1. Seznamte se s problematikou návěstidel a návěstních znaků, zejména se zaměřením na jejich vizuální charakteristiky a odlišnosti.
2. Prostudujte videa z veřejně dostupných zdrojů (např. YouTube kanál parnici.cz) obsahující železniční návěstidla.
3. Navrhněte a implementujte metody pro získání snímků popřípadě sérií snímků návěstidel/návěstních znaků z dostupných videozáznamů.
4. Navrhněte metody a implementujte řešení pro detekci a klasifikaci světelných návěstidel, případně návěstních znaků.
5. Na dostatečně velké množině dat ověřte funkčnost implementovaných řešení.
6. Zhodnoťte a popište dosažené výsledky.

Rozsah diplomové práce: **doporuč. 50 s. původního textu**
Rozsah grafických prací: **dle potřeby**
Forma zpracování diplomové práce: **tištěná/elektronická**
Jazyk zpracování: **Angličtina**

Seznam doporučené literatury:

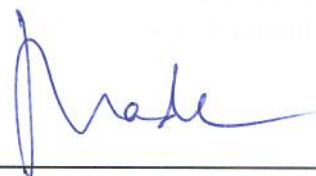
dodá vedoucí diplomové práce

Vedoucí diplomové práce: **Ing. Pavel Mautner, Ph.D.**
Katedra informatiky a výpočetní techniky

Datum zadání diplomové práce: **9. září 2024**
Termín odevzdání diplomové práce: **15. května 2025**



Doc. Ing. Miloš Železný, Ph.D.
děkan



Doc. Ing. Přemysl Brada, MSc., Ph.D.
vedoucí katedry

Declaration

I hereby declare that this Master's Thesis is completely my own work and that I used only the cited sources, literature, and other resources. This thesis has not been used to obtain another or the same academic degree.

I acknowledge that my thesis is subject to the rights and obligations arising from Act No. 121/2000 Coll., the Copyright Act as amended, in particular the fact that the University of West Bohemia has the right to conclude a licence agreement for the use of this thesis as a school work pursuant to Section 60(1) of the Copyright Act.

V Plzni, on 13 May 2025

.....
Daniel Schnurpfeil

The names of products, technologies, services, applications, companies, etc. used in the text may be trademarks or registered trademarks of their respective owners.

During the writing of the thesis, Claude and Mistral LLMs were used to edit parts of the text. After using this tools, the content was checked and edited as needed and the author takes full responsibility for the final content of the thesis.

Abstract

Automated recognition of railway signals is a critical component for enhancing safety in railway transport systems. The thesis focuses on automated visual detection and classification of railway signals within the Czech railway system using a combination of traditional computer vision techniques and modern deep learning architectures. A multistage approach integrates YOLO and RT-DETR models for signal detection, followed by CNNs for state classification. The research introduces a semi-automatic annotation pipeline for train cabin videos that significantly enhances dataset creation efficiency. Experimental results demonstrate that railway signals can be detected with a high F1 score. The developed system can distinguish between critical signal states including stop, go, warning, and various speed adjustment indicators. This work could provide the basis for the brain of driver assistance systems that could reduce human error and increase operational reliability on Czech railways.

Abstrakt

Automatické rozpoznávání železničních signálů je důležitou součástí pro zvýšení bezpečnosti v železničních dopravních systémech. Práce se zaměřuje na automatizovanou vizuální detekci a klasifikaci železničních návěstidel v rámci českého železničního systému s využitím kombinace tradičních technik počítačového vidění a moderních architektur hlubokého učení. Vícestupňový přístup integruje modely YOLO a RT-DETR pro detekci signálů, CNN pro klasifikaci stavu. Výzkum zavádí poloautomatickou anotační pipeline pro videa z vlakové kabiny, která výrazně zvyšuje efektivitu vytváření datových sad. Experimentální výsledky ukazují, že železniční signály lze detekovat s vysokým F1. Vyvinutý systém dokáže rozlišit kritické stavy návěstidel včetně návěstí stůj, jeď, výstraha a různých ukazatelů nastavení rychlosti. Tato práce by mohla poskytovat základ mozku asistenčních systémů pro strojvedoucí, které by mohly omezit lidské chyby a zvýšit provozní spolehlivost na českých železnicích.

Keywords

computer vision • Czech railways • YOLO • RT-DETR • CNN • railway safety

Contents

1	Introduction	4
2	Czech Railways	5
2.1	Railway Signals	6
2.1.1	Fore Signal	7
2.1.2	Single-Light Signals	7
2.1.3	Stop Signal	7
2.1.4	Multiple-Light Signals	8
3	Computer Vision	11
3.1	Traditional Computer Vision	11
3.1.1	Color Thresholding	11
3.1.2	Perceptual Hash Similarity	12
3.2	Neuron	12
3.3	Multilayer Neural Network	13
3.4	Convolution Neural Network	14
3.5	EfficientNet	16
3.6	You Only Look Once	17
3.7	Real-Time Detection Transformer	18
3.8	Metrics	19
3.8.1	F1 Score	19
3.8.2	Confusion Matrix	20
3.8.3	Mean Average Precision	20
3.8.4	Intersection over Union (IoU)	21
4	Related Work	23
4.0.1	France	23
4.0.2	India	23
5	Methodology	25
5.1	Choosing Classes for Railway Lights	26

5.1.1	Basic Detection Approaches	27
5.1.2	Preliminary Classification Schemes	27
5.1.3	Main Classification Approach	27
5.2	Neural Network Architectures Proposal	29
5.3	Semi Automatic Data Annotation Methods	30
5.3.1	Region of Interest Detection	30
5.3.2	Manual Corrections	31
6	Data Analysis	32
6.1	Data Resources	32
6.1.1	Cabview Videos	33
6.1.2	Parníci CZ Videos	34
6.1.3	Pohled z vlaku Videos	35
6.1.4	Discussion	35
6.2	Synthetic Image Dataset	37
6.3	Preliminary Experiments	37
6.3.1	Detection Performance	37
7	Implementation	38
7.1	Dataset Storage	38
7.2	Metadata Structure	38
7.3	Dataset Generation & Reconstruction Scripts	39
7.4	Traditional Computer Vision Components	40
7.5	Neural Network Implementation	40
7.5.1	Detection Models	41
7.5.2	Classification Models	41
7.6	Training Scripts	42
8	Results	43
8.1	Basic Detection Results	43
8.1.1	Railway Light Detection Performance	43
8.1.2	Dual Category Railway Light Detection	44
8.1.3	Multi-Feature Railway Signal Detection	44
8.2	Preliminary Detection & Classification Results	45
8.2.1	Binary State Classification	45
8.2.2	Basic Signal State Classification	45
8.3	Detection & Classification with Environmental Context	47
8.3.1	Single Images Dataset	47
8.3.2	Enhanced Classification with Environmental Context	49
8.4	Classification from ROI	52

8.4.1	Discussion	54
8.5	Test Performance	55
8.5.1	Two-stage Model	55
8.5.2	Single-stage Models	57
9	Conclusion	60
	Bibliography	61
	List of Figures	63
	List of Tables	64

Introduction

1

Railway signaling systems serve as the safety control mechanism for locomotive drivers. They contain a lot of information on speed restrictions and operational permissions. The Czech railways have a specific signaling system consisting of various signal types, such as signals with single or multiple lights, each with specific meanings and operational implications.

Due to the recent modernization of railways in European countries, such as the implementation of the European Train Protection System, traditional signaling systems must now coexist with newer technologies. During this time, automated visual recognition of traditional signals has an advantage over existing safety protocols and could potentially assist in reducing human errors.

To achieve automated visual recognition of railway signals, computer vision approaches used in this thesis range from traditional algorithmic methods to advanced deep learning architectures, such as Convolution Neural Networks (CNNs), EfficientNet, YOLO (You Only Look Once) and Real-Time Detection Transformers (RT-DETR). Each approach has different advantages and disadvantages with regard to computational speed and detection efficiency.

In the methodology, a multistage approach is also introduced for semiautomatic annotation of videos captured from train cabins, which uses both traditional computer vision methods and modern deep learning architectures. The main goal of this thesis is to develop robust models that can detect railway signals in diverse environmental conditions, including varying lighting, weather conditions, and viewing angles.

The results of this thesis could have potential applications in driver assistance systems, training simulators, safety monitoring, and as a supplementary system during the transition to fully automated train control systems. In other words, this work could contribute to the larger goal of improving railway safety and operational efficiency in the Czech Republic and similar signaling systems used in other countries.

Czech Railways

2

Railway transport systems through Czechia have been improved with respect to the modernization of the safety infrastructure. The Czech railway network is approximately 9,500 kilometers[19] long and has many signaling technologies, from legacy mechanical systems to the modern European Train Control System (ETCS).

Statistical analyses of railway incidents within the Czech railway system reveal a lot of signal violations. The visualization below (Figure 2.1) shows the number of train accidents caused by illegal driving through railway signals in 2018 - 2024, with annual incidents between 136 and 183 cases[25]. There is a significant number of in-

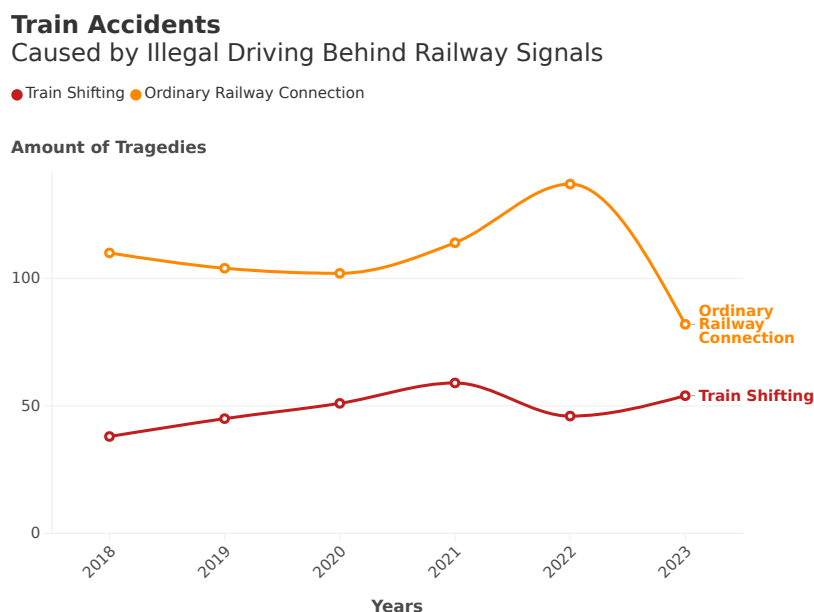


Figure 2.1: Train accidents caused by illegal driving through railway signals, data are taken from [25]

idents occurring during the summer months and transitional seasonal periods. The data indicates a bifurcation between violations occurring during regular train oper-

ations versus shunting operations, with train operations consistently representing approximately 68-75% of total incidents during the measured period.

A critical factor that influences railway safety in Czechia is the heterogeneous nature of signaling infrastructure. Although the main railways are modernized with contemporary European standards, regional and secondary lines continue to operate without advanced train protection systems. This can lead to difficulties for driver adaptation across a lot of infrastructure environments.

The Czech national *Traffic and Signaling Regulations*[Svo24] specify the operational characteristics of rail transport. This regulatory framework defines both visual signal characteristics and operational responses, creating context for computer vision implementation.

2.1 Railway Signals

Railway signals represent a visual communication tool for locomotive drivers. These signals contain specific combinations of lights, shapes, and colors to transmit instructions about speed limits, track availability, and required actions.

The light signals on Czech railways operate through a system of colored lights mounted on standardized signal posts. The most frequent signal colors are red, green, yellow, and white. Each of the colors or color combinations has a different meaning. Red lights typically indicate a stop request, while green lights allow for unlimited movement. The yellow light serves as a warning sign, preparing locomotive drivers for the following limitations. White lights are often in shunting¹ signals or as additional indicators.

The signals combine these colors in various patterns to communicate more complex messages. For example, two vertically positioned yellow lights (Figure 2.3) inform the locomotive driver to reduce speed and expect a stop signal ahead. The position and blinking of the light(s) add another piece of information to the basic colors.

In addition to the light-based system, there are fixed railway signs. Signs and markers display speed limits, distance warnings, and track identification. Their design emphasizes visibility in various weather conditions through reflective materials and high-contrast color schemes. Signal boards often use standardized shapes, such as circles or white triangles, which serve as warning signs. They are not part of the thesis.

¹Shunting in railways is the process of moving trains, wagons within a station to assemble, disassemble, or relocate them for operational purposes.

2.1.1 Fore Signal

In the *Traffic and Signaling Regulations*[Svo24], the fore signal called "předvěst" in Czech is described as follows. This signal depends on the main signal. Typically, the fore signal adds another notification for the locomotive driver before he can see the main signal. As a rule, the fore signals are smaller in size than regular single-light signals and multiple-light signals.

2.1.2 Single-Light Signals

This section is focused on single-light signals and their characteristics as described in the *Traffic and Signaling Regulations*. [Svo24]

There are several distinct single-light signals, each characterized by a specific color and blinking pattern. For example, the go signal called "Volno" in Czech, represented by a steady green light, allows the train to go.

This signal indicates that there is a similar instruction on the following main signal. In contrast, signals such as "Expect speed of 40 km/h" (slowly flashing yellow

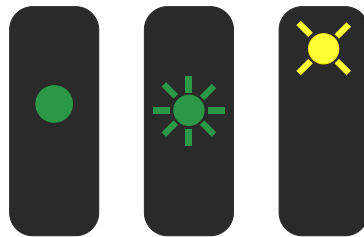


Figure 2.2: "Návěst Volno" (go signal) steady green light on the left; "Očekávejte rychlost 100 km/h" (Expect speed of 100 km/h): rapidly flashing green light in the center; "Návěst Očekávejte rychlost 40 km/h" (Expect speed of 40 km/h): slowly flashing yellow light on the right

light) or "Expect speed of 100 km/h" (rapidly flashing green light) allow the movement of the train while preparing the driver for a speed restriction at the subsequent signal, typically positioned at least at braking distance. These speed-related signals are from 40 km/h to 120 km/h. They are represented by patterns of slow or rapid flashing colors (yellow or green).

2.1.3 Stop Signal

In the regulation *Traffic and Signaling Regulations*[Svo24] there is also the description of the stop signal. It is named "návěst Stůj" in the Czech railway signaling system and has the form of a single red light on the main signal devices. This signal is the most significant safety mechanism in the railway infrastructure.

Based on this signal, the locomotive driver has to stop the locomotive approximately 10 meters in front of the signal device when displaying the stop signal. This signal is also used for shunting operations or special maintenance vehicles. In situations where the main signal is not positioned directly adjacent to the track, the train must stop before reaching the end of the train path indicator.

There are two types of the stop signal. One is the absolute stop and the other is the permissive type. The absolute signal means that when the red light switches to the usual go signal, the locomotive driver can continue. The permissive type allows the locomotive driver to continue in certain cases that are defined in *Traffic and Signaling Regulations*[Svo24] with more details.

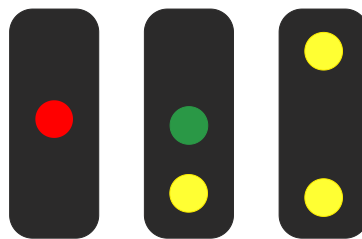


Figure 2.3: "Návěst Stůj" (single red light) on the left; Limit 60 km/h and go in the center; Limit 40 km/h and warning in the right

2.1.4 Multiple-Light Signals

The multiple-light signaling architecture uses vertical light elements where each light combination means a specific state. This system follows a structured logic in which:

1. The lower light element adjusts speed restrictions
2. The upper light element has a similar predictive function as described in Fore Signal section 2.1.1.

The semantics of this system are defined by color coding and flashing patterns. The signal states are:

- **Static yellow lights** - indicating speed restrictions
- **Flashing yellow lights** - indicating the prediction of further speed reductions
- **Green lights (static and flashing)** - permitting a higher speed
- **Numerical indicators** show precise speed thresholds

There are also horizontal illuminated bars that correspond to specific speed thresholds. For example, a horizontal yellow bar indicates a 60 km/h restriction, while a horizontal green bar corresponds to an 80 km/h threshold.

The *Traffic and Signaling Regulations* [Svo24] further describe the relationship between the current and anticipated signal states. That is particularly useful when:

- The predicted speed restriction is lower than the current one (the need to slow down)
- The predicted speed restriction is higher than the current restriction (the need to speed up)

2.1.4.1 Repeater Signals

In *Traffic and Signaling Regulations*[Svo24] also define special states that repeat signals within the Czech railway infrastructure. These repeater signals inform the locomotive driver about the speed limits imposed by subsequent additional white signals located at not very good stopping distances. The architectural realization of these repeater signals is based on a multi-illumination system. The repeater signals



Figure 2.4: "Rychlost 40 km/h a opakování návěsti Výstraha" Speed 40 km/h and repeating the signal Warning

are categorized into several types. The first of them is **the white light with the yellow light above** that allows movement of the train and the maximum speed of the track until the next main signal, but also indicates that the next stop signal is placed at an insufficient braking distance. Another type of repeater signals is represented by Speed-Specific Repeater Signals:

- **Expect Speed 40 km/h** - the white light with the slowly flashing yellow light above indicates the next speed restriction of 40 km/h, 30 km/h, or 50 km/h
- **Expect Speed 60 km/h** - the white light with the rapidly flashing yellow light above indicates the next speed restriction of 60 km/h or 70 km/h

- **Expect Speed 80 km/h** - the white light with the slowly flashing green light above indicates the next speed restriction of 80 km/h
- **Expect Speed 100 km/h** - the white light with the rapidly flashing green light above indicates the next speed restriction of 100 km/h
- **Expect Speed 120 km/h** - the white light with rapidly flashing green light and an illuminated yellow numeral "12" above indicates the next speed restriction of 120 km/h
- **Speed 40 km/h with Various Expected Speeds** - the yellow light, accompanied by the white light above, with varying signal patterns at the top, indicates an immediate 40 km/h restriction and subsequent speed restrictions, depending on the top light configuration, as described in Section 2.1.4.

It is now time to move thematically to the theory of how railway signals could be detected and classified using various sophisticated approaches ranging from traditional computer vision to a real-time transformer.

Computer Vision

3

In the time before neural networks, computer vision (CV) used traditional algorithmic approaches based on image analysis. Traditional computer vision methods could be used for the detection of railway signaling systems as well as modern neural network approaches.

3.1 Traditional Computer Vision

Traditional computer vision techniques focused on feature extraction and algorithmic processing. Unlike neural networks that learn representations automatically, these classical approaches required human experts to design specific algorithms such as **edge detection**, **color segmentation**, shape recognition and pattern matching, which are important for tasks like railway signal detection and classification. Systems designed to detect the various signal types described in section 2.1 (such as fore signals, single-light signals, and multiple-light signals) would traditionally use techniques such as edge detection to detect railway signal housings, color thresholding to isolate illuminated elements, and Hough transforms to detect circular light patterns. Some of the techniques will be described in further sections.

3.1.1 Color Thresholding

Color thresholding is an image segmentation technique that splits an image based on color similarity rather than just intensity values. Unlike traditional grayscale thresholding that operates on a single intensity channel, color thresholding applies selection criteria to **multiple color channels** (typically RGB) and combines them with Boolean operators to extract regions of interest [Kul12]. This approach is particularly effective for natural image segmentation, where color information provides significantly more discriminative power than grayscale alone, allowing for the extraction of specific objects, such as forests (green regions) or the sky (blue regions) from complex backgrounds [Kul12].

3.1.2 Perceptual Hash Similarity

The perceptual hash (pHash) is a perceptual rolling hash of the image that computes the similarity between images based on visual information. When it generates the hash, web-pHash does not compare the pixels itself, but it compares the different representations in the frequency domain of the image [Zau10]. Perceptual Hash then takes the image, converts it to grayscale and resizes it, after which it computes the **discrete cosine transform** (DCT) which retains the low-frequency components that convey the perceptual aspects of the image. According to [Zau10], it produces a fingerprint, the binary hash, by comparing each DCT coefficient to the median value, thus significantly unaffected by minor changes while sensitive to major structural changes. This allows the similarity between two images to be measured as the **Hamming distance** between their hashes, with smaller distances correlating to greater visual similarity.

3.2 Neuron

In the following sections, we will discuss the topic of computer vision artificial neural networks, limiting only to neural networks suitable for classification and detection. In the first section, we will introduce the basic principles of neural networks. In biological terms, it is a nervous cell which can carry the signal or signals further

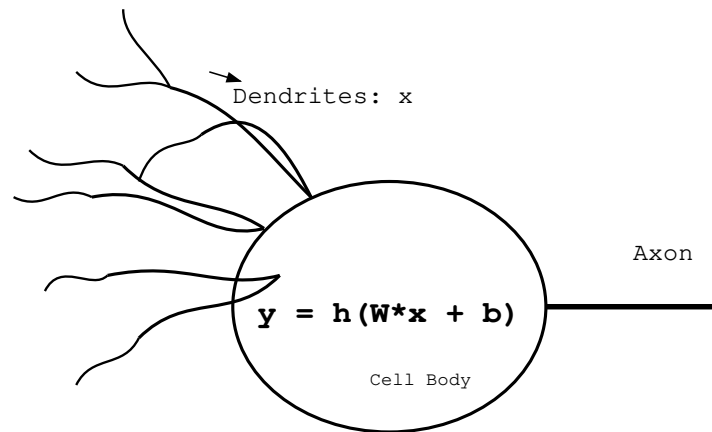


Figure 3.1: Biological model of neuron with mathematical formula

on the basis of the intensity of the input signals. An artificial neuron works on a similar principle, where input signals (dendrites in biological terms) are represented by numerical values. They are represented by \mathbf{x} in the picture 3.1. Then there are weights (\mathbf{W} in the picture 3.1).

The weights are tensors that can learn/save their state. The training process has a few steps:

1. **forward pass:** Input data flows through the network, generating predictions.
2. **error calculation:** The difference between the predictions and the actual targets is measured using a loss function.
3. **backward Pass:** The algorithm (backpropagation) calculates how each parameter participated in the error using the chain rule of calculus to compute gradients.
4. **parameter Updates:** The weights and biases are adjusted in the opposite direction of their gradients, typically using the gradient descent method.

The formula in this state shown in the picture 3.1 plays its role in the forward pass. The neuron is shown on a perceptron, which is the simplest neural network consisting of a single neuron [MSL93].

The **b** is bias. The bias enables a neuron to activate even when all inputs are zero/low or to remain inactive despite receiving large portions of signals. The **h** is the activation function, sometimes called non-linearity, for example it can be a sigmoid function or possibly even a more sophisticated function available nowadays.

The outputs are also tensors, like the input and weights. When the output is considered as the input to another neuron, it is called a neural network afterward. This is because the perceptron itself is very simple. A perceptron can only learn linearly separable patterns, unlike more advanced models such as logistic regression, which can handle probabilistic classification and provide continuous outputs between 0 and 1. That is why a multilayer neural network was found.

3.3 Multilayer Neural Network

Neurons are usually orchestrated in layers. The neuron input from each layer except the first (in feed forward neural networks) is forwarded to each next neuron in each layer if the layers are fully connected as shown in Picture 3.2.

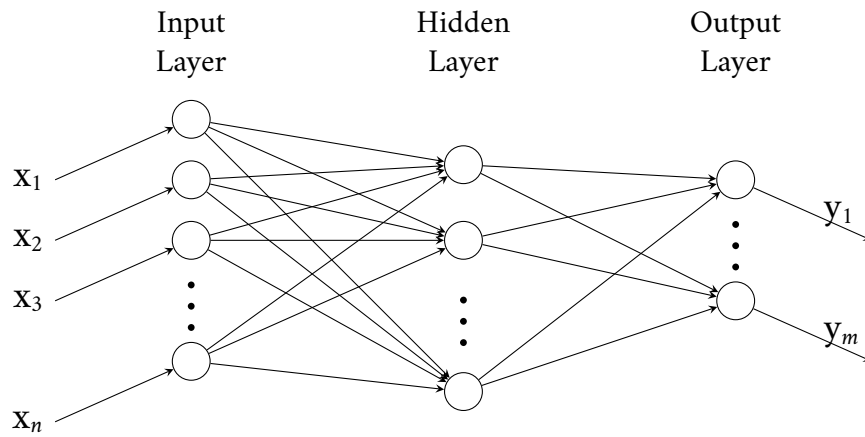


Figure 3.2: Multiple Layer Neural Network

3.4 Convolution Neural Network

The convolution neural network is designed to process multidimensional data [YBH15]. Such data are, for example, color images, which can be represented by, for example, three two-dimensional arrays containing pixel intensities in three color channels (red, green and blue).

In the simplest terms, convolution is a mathematical operation that, in our case, is used to modify an image so that the edges within the image are highlighted, which is important for our objective.[Sch22]

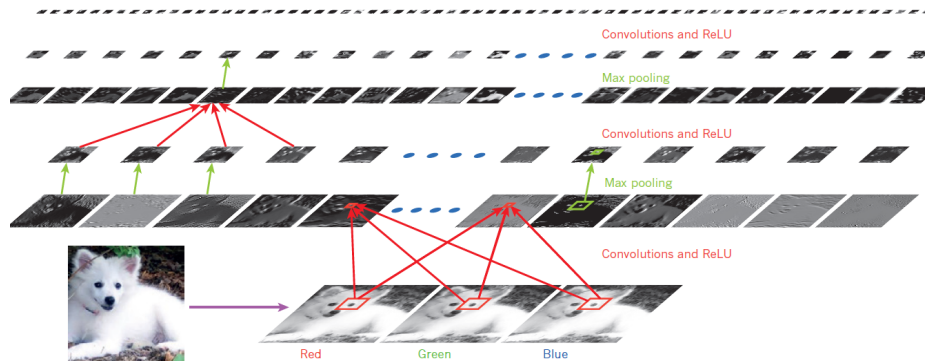


Figure 3.3: CNN Layers (picture is taken from [YBH15])

3.4.0.1 Convolution

At first, it is essential to look at the convolution in more detail. As an example, consider a black and white image represented by the matrix shown on the left in the image 3.4. The values in the matrix represent the brightness intensities of the

pixels. Next, we have the so-called kernel matrix (the convolution mask in Figure 3.4). Both matrices are then processed in the following function:

$$V_{i,j} = (M, N)_{i,j} = \sum_{a=-k}^k \sum_{b=-k}^k M(i-a, j-b) \cdot N(a, b), \quad (3.1)$$

where $V_{i,j}$ is the resulting pixel value at the position of indices i and j , M is the area in the V matrix, and N is the kernel matrix of k rows and k columns. The convolution can be seen in the following figure 3.4. It is clear in the figure 3.4 that for a kernel

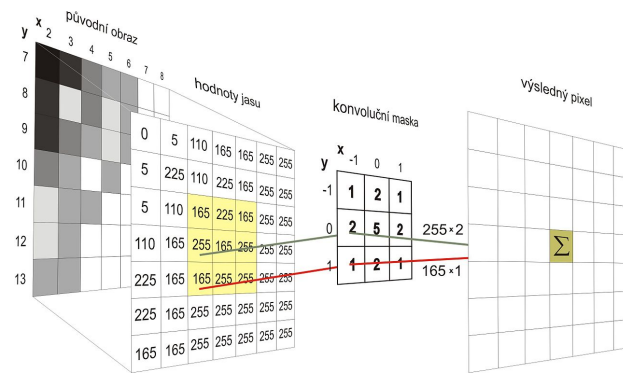


Figure 3.4: Convolution Example (picture is taken from [06])

matrix of three columns and three rows, each index value is multiplied by the index value of a given region of the matrix with the same dimensions as the kernel matrix.

3.4.0.2 Pooling

This method is similar to the convolution described in the previous paragraphs of the 3.4.0.1 section. The method traverses the image matrix by regions and calculates just one pixel using a defined function, for example, by averaging the values in the given regions (average pooling) or by calculating the maximum value in each subregion (max pooling).[Sch22]

3.4.0.3 CNN Architecture

The structure of a convolution neural network is composed of two parts. The first is the part that processes the input image. This part consists of convolution and pooling layers. The result of this part is a vector of features that is used as input to the layered neural network.

3.5 EfficientNet

Between traditional CNN architectures and modern object detectors like YOLO, EfficientNet plays a role in the design of neural networks that influenced subsequent architectures [TL20]. The main improvement of EfficientNet is in its compound scaling method, which addresses fundamental limitations of previous scaling approaches.

Most convolution neural networks were scaled up by relatively by either increasing the network depth (adding more layers) and width (expanding the channel dimensions), or by independently increasing the input resolution. As shown in Figure 3.5, the authors of EfficientNet showed that network dimensions work best together, not separately. Higher resolution images need both deeper grids (for larger receptive fields) and wider grids (to capture fine detail). Scaling these dimensions individually does not work well, but scaling them together does.[TL20]. The composite scaling

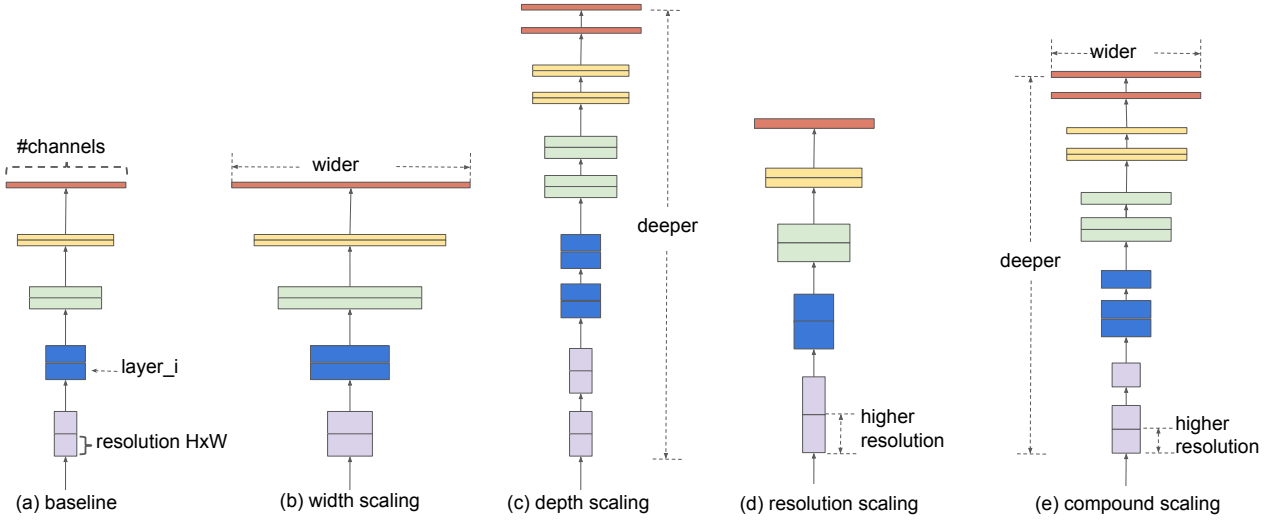


Figure 3.5: (a) baseline; (b)-(d) advanced methods; (e) their proposed compound scaling method (Diagram and label are taken from [TL20])

method has three coefficients, α for depth, β for width, and γ for resolution, which are balanced by the limits $\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$. This ensures that computing resources scale predictably with network size. The scaling formula is applied uniformly:

$$\text{depth: } d = \alpha^\phi \quad \text{width: } w = \beta^\phi \quad \text{resolution: } r = \gamma^\phi \quad (3.2)$$

Where ϕ is a user-defined coefficient that controls the available resources, while α , β , and γ determine the distribution of resources between dimensions. The basic architecture of EfficientNet-B0 was developed based on a search for a neural architecture with optimized accuracy and computational efficiency. Its basic building block is

the mobile inverted bottleneck MBConv, augmented with squeeze-and-excitation optimization. From this basis, neural network types (B0-B7) are generated using a composite scaling method.[TL20] EfficientNet improves efficiency - EfficientNet-B7 achieves 84.3% top-1 accuracy on ImageNet using $8.4\times$ fewer parameters and $6.1\times$ faster execution than comparable [TL20] models. When visualized using activation class maps, composite scaling models have a better focus on relevant regions of the image with greater object detail compared to one-dimensional scaling approaches. This architecture has become suitable for resource-constrained applications where performance efficiency is critical.

3.6 You Only Look Once

YOLO (You Only Look Once) introduced in [Red+16] is the advanced variant of the convolution neural network. Unlike traditional methods that use sliding windows or region suggestions, YOLO processes the entire image in a single evaluation and predicts bounding boxes and class probabilities in parallel through a unified neural network architecture [Red+16]. The system partitions the input images into an $S\times S$ grid, where each grid cell predicts N bounding boxes with associated confidence scores and class probabilities. These predictions are represented as an $S\times S\times(N\times 5+C)$ tensor, where C is the number of classes. The network architecture consists of

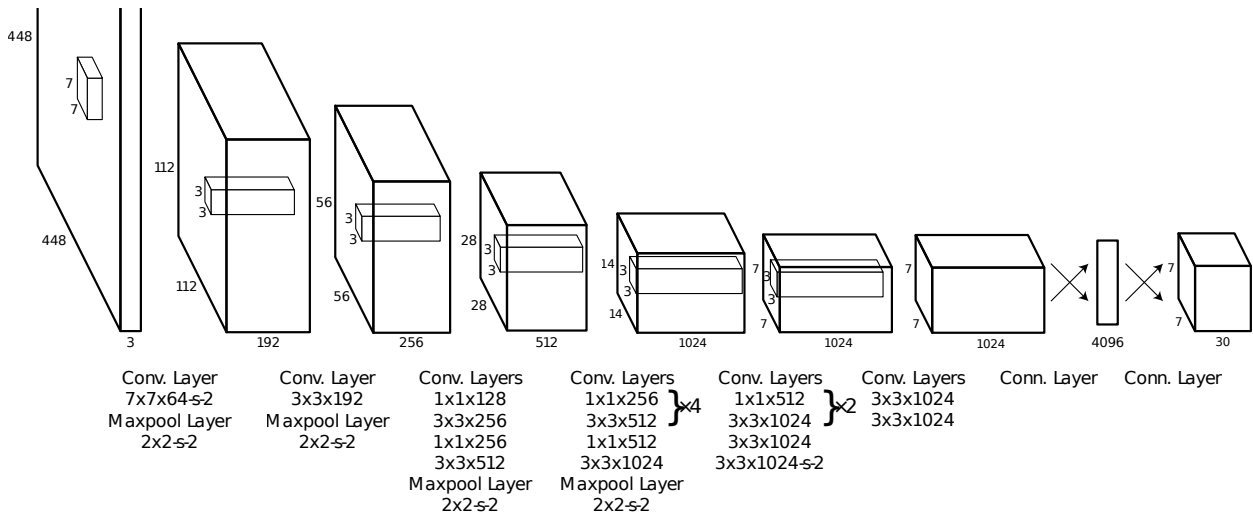


Figure 3.6: One of the first YOLO architectures (Diagram is taken from [Red+16])

24 convolution layers described in previous sections. They are followed by 2 fully connected layers that take inspiration from the GoogLeNet model but 1x1 reduction layers are followed by 3x3 convolution layers. During inference, YOLO processes images at 448x448 resolution, generating approximately 98 bounding boxes per

image. This unified approach is capable of an end-to-end optimization directly on detection performance, the result is an extremely fast processing that speeds up the base YOLO model that operates at 45 frames per second while a faster variant reaches 155 frames per second. This real-time performance, combined with YOLO's ability to detect objects in images, makes it particularly suitable for applications that require robust object detection. Therefore, YOLO has been chosen as one of the main tools in this thesis.

3.7 Real-Time Detection Transformer

The architecture of the Real-Time Detection Transformer (RT-DETR) comes from the End-to-End Object Detection with Transformers paper[Car+20] by Facebook AI research.

The Detection Transformer uses an encoder-decoder architecture, which was originally made for natural language processing tasks. The diagram 3.7 demonstrates its architecture. The Detection Transformer has a convolution backbone to extract visual features from the image, which are then passed to the transformer encoder. The encoder processes these features and creates a representation of the image that is then passed to the decoder. The decoder then uses a set of learned object queries to predict the classes and positions (bounding boxes) of objects in the image. This architecture is not exactly optimal for real-time use cases due to its

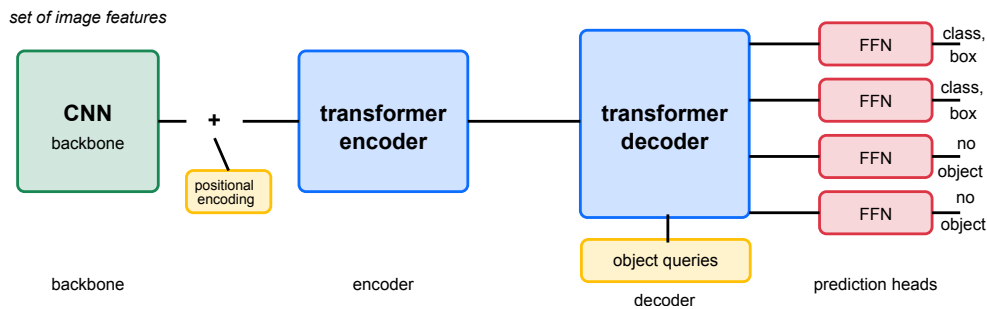


Figure 3.7: Simple diagram of the Detection Transformer architecture

computational complexity and relatively high inference latency. The transformer architecture, while powerful for modeling global interactions between image elements, involves multiple attention mechanisms that require significant computational resources. Self-attention operations scale quadratically with input size, which becomes problematic for high-resolution images or video frames.

Researchers from China[Lv+23] made a few improvements and created the Real-Time Detection Transformer that has an uncertainty-minimal query selection

mechanism that optimizes the initialization of the object query. Unlike previous approaches that rely on classification confidence, this process explicitly performs joint variable classification and localization confidence, which minimizes the uncertainty of the selected features and makes the initial queries of higher quality to the decoder. The researchers from China[Lv+23] also compared their models with the detectors YOLOv5, YOLOv6, YOLOv7 and YOLOv8. Their RT-DETR-R50 achieves **53.1% Average Precision (AP)** on the COCO validation dataset while processing 108 frames per second (FPS) on an NVIDIA T4 GPU. The RT-DETR-R101 configuration reaches 54.3% AP at 74 FPS. These metrics outperform some of the comparable YOLO detectors in both accuracy and computational efficiency. In addition, RT-DETR offers practical advantages with flexible rate tuning options that allow adaptation to different deployment scenarios without the need for retraining by adjusting the decoding layer configuration. Most significantly, RT-DETR eliminates the need for Non-Maximum Suppression (NMS) post-processing¹ which is rather difficult with hyperparameter dependencies in traditional detection frameworks.

3.8 Metrics

The evaluation of machine learning models, particularly in classification tasks, requires robust metrics that can tell more than simple accuracy.

3.8.1 F1 Score

Precision defines the actual positives correctly identified by the model. Recall means how good a model is in finding all relevant instances. The F1 score harmonically balances precision and recall in a single metric, which is useful when class distributions are imbalanced.

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3.3)$$

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \quad (3.4)$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \quad (3.5)$$

¹Non-Maximum Suppression (NMS) post-processing is technique of deleting duplicities in detected regions of interest.

where:

- True Positives (TP) – Instances correctly predicted as positive
- True Negatives (TN) – Instances correctly predicted as negative
- False Positives (FP) – Negative instances incorrectly predicted as positive (Type I error)
- False Negatives (FN) – Positive instances incorrectly predicted as negative (Type II error)

3.8.2 Confusion Matrix

The confusion matrix is a table of predicted versus actual class labels. From this matrix, we can compute precision, which measures the proportion of true positive predictions among all positive predictions, indicating the model's ability to avoid false positives.

Table 3.1: Confusion Matrix Example

Class	Predicted Class	
	Positive	Negative
Actual Positive	True Positives (TP)	False Negatives (FN)
Actual Negative	False Positives (FP)	True Negatives (TN)

3.8.3 Mean Average Precision

Mean Average Precision (mAP) is a broad metric for evaluating object detection systems, in particular advanced models such as those within the Ultralytics YOLO family [Ult25b]. This metric combines detection performance on many detection thresholds and object classes into a single number score. Although basic evaluation methods are readily available, mAP offers a balance between recall (the ability to detect all relevant objects) and precision (the exact spot where they are detected), which has made it the recommended measure in most fields for judging computer vision models [Ult25b].

$$\text{mAP} = \frac{1}{n} \sum_{i=1}^n \text{AP}_i \quad (3.6)$$

$$\text{AP}_i = \frac{1}{n} \sum_{j=1}^n \sum_{j=1}^m (R_j - R_{j-1}) \times P_j \quad (3.7)$$

The mAP is calculated as the average of Average Precision (AP) values across n object classes, where AP for each class is the area under the precision-recall curve,

computed as a sum of precision values P at each recall threshold R , weighted by the change in recall from the previous threshold, with m total recall thresholds considered in the evaluation.

3.8.4 Intersection over Union (IoU)

IoU (Intersection over Union) is a basic metric that is a common task that is used in computer vision. This metric measures how well the predicted boundaries (bounding boxes, for example) match the corresponding boundary of the ground truth object[Ult25a]. IoU is the ratio of the overlap area between the predicted and ground truth boxes to the union area between the predicted and ground truth boxes.

3.8.4.1 Definition & Computation

IoU is calculated as follows, where the intersection area is the area in which the predicted bounding box overlaps with the ground truth bounding box, according to the following equation.

$$\text{IoU} = \frac{\text{Area of Intersection}}{\text{Area of Union}} \quad (3.8)$$

This ratio will give us a score between 0 and 1, with a score of 1 that means there is perfect overlap (the predicted box is equal to the ground truth) and a score of 0 meaning there is no overlap at all[Ult25a].

3.8.4.2 Role in Model Evaluation

This IoU metric is an important metric when evaluating the performance of object detection model such as YOLO. Classification metrics can identify what objects exist in an image, but the IoU states how accurately these objects are localized in a particular image [Ult25a]. Several object detection benchmarks, such as the COCO or PASCAL VOC evaluations, score detections based on IoU thresholds (most often set at 0.5) for whether detections should be considered correct or not.

3.8.4.3 IoU in Advanced Model Training

In addition to being used for evaluation, IoU is now an important part of the training process itself. State-of-the-art architectures for object detection (YOLOv8, YOLOv10, etc.) combine IoU or variants of IoU (GIoU, DIoU, CIoU) in their loss functions. Thus, these losses based on IoU enable the models to have better overlap properties of the predicted boxes taking into account the center distance and the consistency of aspect ratio[Ult25a].

3.8.4.4 **mAP at Different IoU Thresholds**

The mAP metric is typically calculated at specific Intersection over Union (IoU) thresholds. Two common variants are:

- **mAP@.50** (or mAP50): Mean Average Precision calculated at 50% IoU threshold. This means that a detection is considered correct only if the predicted bounding box overlaps with the ground truth by at least 50%.
- **mAP@.50:.95** (or mAP50-95): Mean Average Precision averaged over multiple IoU thresholds, from 50% to 95% with a step size of 5% (i.e., IoU thresholds of .50, .55, .60, ..., .95).

The mAP50-95 is generally considered a stricter metric than mAP50 alone, as it evaluates the model's performance in detecting objects with high localization accuracy.

Related Work

4

In the following chapter, there is a review of the related work performed in foreign countries for railway signal detection and recognition systems.

4.0.1 France

An interesting example of related work can be found in France. In 2022 [Sta+22], a deep learning approach was presented there, using **YOLOv5** for the detection and classification of railway signals, combined with a heuristic method for recognizing blinking signal states. Their system achieved high performance metrics, including 0.976 mean precision (mP@0.5) and 0.959 mean recall (mR@0.5), with a mean average precision (mAP@[0.5, 0.95]) of **0.737**, while maintaining real-time processing capabilities with an inference time of approximately 17.4 milliseconds (about 57 frames per second). Although urban railway systems have reached the Grade of Automation 4 (GoA4), mainline railways face additional challenges due to shared infrastructure, multiple types of trains, higher speeds, and complex signaling systems [Sta+22]. The authors used the **FRSign dataset**, which contains 4,900 trainable objects: 4,171 objects in 2,094 frames for training and 472 objects in 399 frames for validation. Their approach demonstrated robust performance under various environmental conditions, although extremely poor illumination and distant signals remained challenging [Sta+22].

4.0.2 India

Another case of similar approach can be traced to India, where in 2017. [RMR17] they developed a system for railway signal detection and track-specific association. They used pipeline in three steps. The first was a simplified CNN with transfer learning using Inception V3 for track detection, the second was by track-specific ROI selection, and the final step was signal detection and color recognition using Faster R-CNN. The system achieved **0.94 accuracy** over 150km with 247 signals under various environmental conditions. They had problems with signal placement variability, appearance differences, and detection between multiple parallel tracks.

The authors evaluated several alternative approaches, including fixed ROI and rule-based ROI selection, which were ultimately rejected due to limitations in handling spatial variations and track positioning edge cases.

Their implementation supports railway automation and compliance with Positive Train Control regulations by using locomotive-mounted cameras to build a database of signal locations.

Methodology

5

In this chapter, methodological approaches are introduced for data extraction, processing, feature extraction, and model architecture proposals, as well as describing the domain-specific challenges related to the visual interpretation of the railway signal for choosing classification classes. The research methodology in this diploma

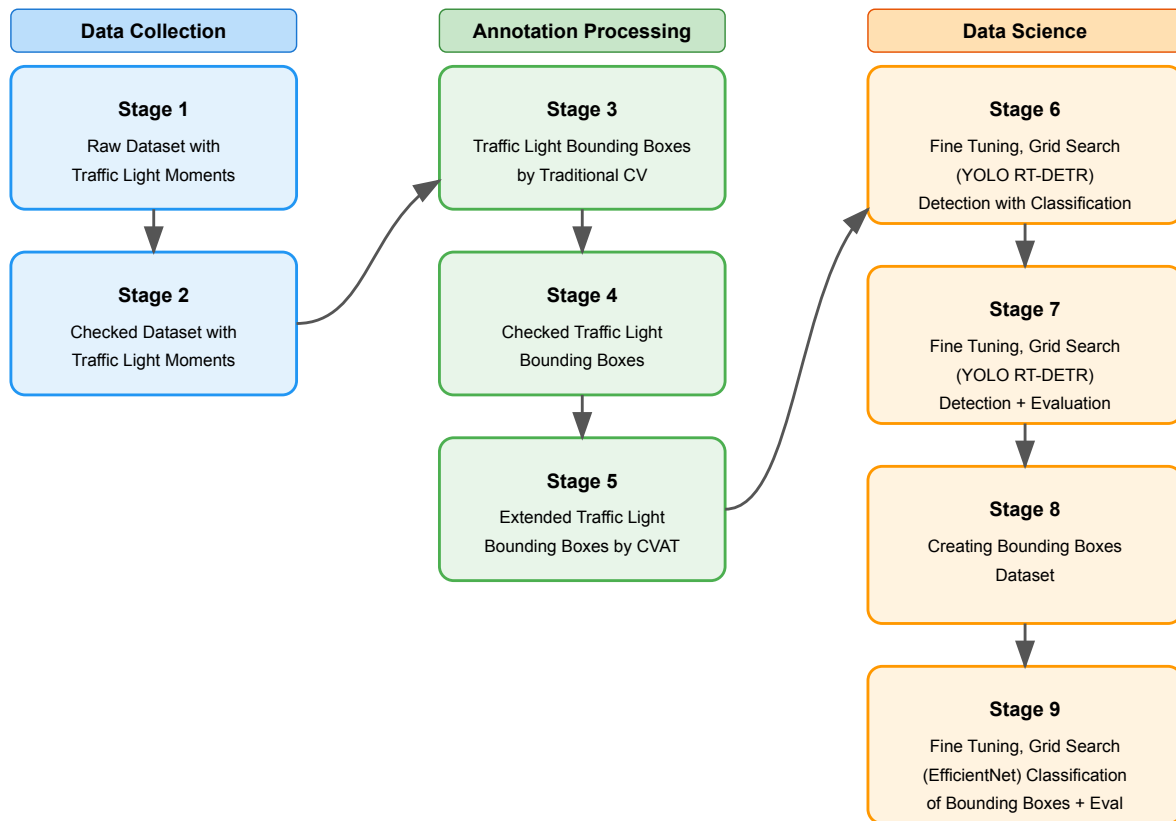


Figure 5.1: Approach for Automatic annotation

this thesis follows a multistage approach to traffic light detection and classification. Initially, raw traffic light data was collected and validated through a verification process.

In the methodology, both traditional computer vision techniques and modern deep learning approaches are used to achieve optimal results.

In the preliminary stages, traditional computer vision methods were applied to establish baseline bounding boxes around traffic lights, which were subsequently verified and extended using the Computer Vision Annotation Tool (CVAT). The core detection methodology involved extensive fine-tuning and hyperparameter optimization through grid search, exploring state-of-the-art models including YOLO and RT-DETR architectures. For precise traffic light state classification, the methodology incorporated EfficientNet models trained on a carefully curated bounding box dataset.

This hybrid approach combining traditional computer vision techniques with advanced deep learning models ensured robust detection and accurate classification of traffic lights under various environmental conditions, providing a solid foundation for the experimental results presented in subsequent sections.

5.1 Choosing Classes for Railway Lights

In this section, there are defined classification categories based on visual analysis, taking into account the trade-off between classification complexity and model performance. The methodology went through several iterations, starting with simpler detection tasks that were then layered with more sophisticated classification tasks. The classes were designed regardless of the signal type (the fore or the main signal) because of their similarities (stop and go) for safety purposes and a wider coverage of the single model. The classification strategy considerations:

- **Safety criticality:** Signal types with the most significant safety implications (stop/go) were prioritized in all classification schemes.
- **Visual distinctiveness:** Classes were defined to maximize visual separation between categories, improving the performance of the model.

Based on visual analysis and preliminary testing, we expect performance trade-offs across our classification schemes.

- **Detection accuracy:** Detection accuracy is expected to decrease as the class count increases, particularly in challenging lighting and weather conditions.
- **Classification precision:** More refined class definitions provide greater operational value but require a higher quality of input data.
- **False positive reduction:** Explicit distraction classes (e.g. road traffic signals) should significantly reduce misclassifications of non-railway signals.

- **Computational demands:** More complex classification schemes require greater computational resources and can potentially affect real-time performance.

5.1.1 Basic Detection Approaches

Initially, minimalist detection approaches are described:

- **railway light detection:** This simplest approach focused only on identifying the presence of railway lights in video frames, without distinguishing their state or orientation.
- **dual category railway light:** The expanded classes that classify between frontal railway lights (directly visible) and those viewed from behind.
- **multi-feature railway signal detection:** Further refinements introduced categories for **active lights (on)**, **inactive lights (off)**, **light from behind**, and **other** railway-related objects that could generate false positives.

5.1.2 Preliminary Classification Schemes

Detection possibilities, as simple state classification:

- **binary state classification:** Classification only between stop signals (single red light) and go signals (single green light).
- **basic signal state scheme:** Classification expanded to include the "twice yellow" signal configuration that indicates speed adjustment and warning states, adding important nuance to operational guidance.

5.1.3 Main Classification Approach

The mature classification model could have six distinct signal states:

- **single red light:** Indicating stop
- **single green light:** Indicating go at normal speed
- **single yellow light:** Caution signal
- **twice yellow (vertical arrangement):** Adjust speed and prepare for potential restrictions
- **upper green, lower yellow combination:** Go with caution, often indicating reduced speed requirement

- **lights turned off:** Nonactive signaling state

The current approach does not include repeater states with white lights as this would exceed the scope of this thesis. However, the class structure has been designed for easy integration of both white-light states by fine-tuning the current models and also blinking-light detection through straightforward heuristic approaches in the future.

5.1.3.1 **Enhanced Classification with Environmental Context**

The final iteration incorporated additional classes to improve reliability by explicitly handling potential sources of confusion.

- **Crossing traffic lights for vehicles:** Similar in appearance but functionally distinct from railway signals
- **Lights viewed from behind:** For robust detection regardless of camera perspective

5.2 Neural Network Architectures Proposal

There are proposed two significantly different architectures, which will be referred to as single-stage and two-stage models. These two approaches have different processing methods and characteristics associated with computational efficiency. Single-

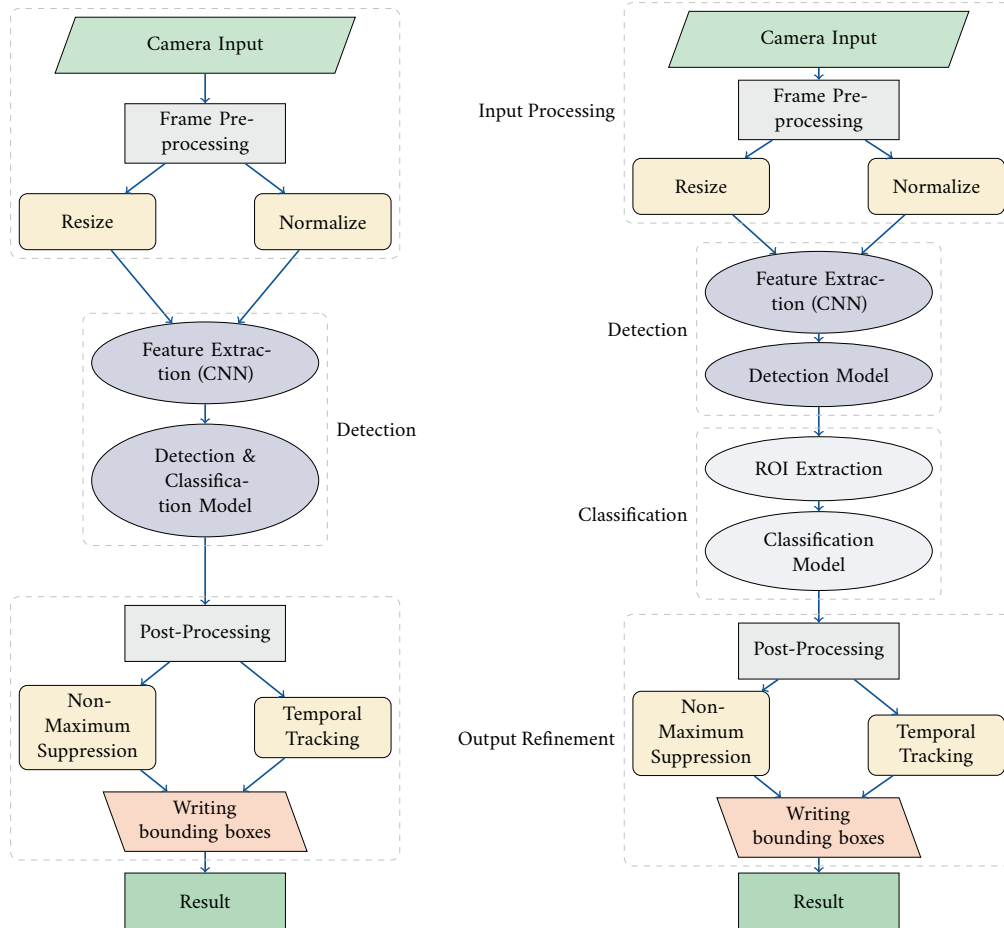


Figure 5.2: Single-stage vs Two-stage Model

stage detection models take advantage of the complete **object localization and classification** process, as a **single pass through** the network architecture predicts bounding box coordinates and class probabilities from input feature maps without intermediate region proposal mechanisms. This simplification of the network architecture has benefits in terms of computational efficiency, making it suitable for real-time inference, which is important for applications that require rapid processing, e.g. autonomous navigation systems. However, such efficiency often leads to performance decay in complex detection tasks that involve small objects or closely grouped images. In contrast, the second approach breaks detection into two sequen-

tial operations. In the first stage, a specialized Region Proposal Network (RPN) is used to propose candidate object locations; in the second stage, these proposals are refined through exact **localization** and **classification** processes.

5.3 Semi Automatic Data Annotation Methods

Due to the fact that the pure **rule-based detection algorithms were unsuccessful** in the detection of railway signals, as stated in the 4.0.2 section, a hybrid approach is proposed. The detection and recognition of railway signals is slightly similar to

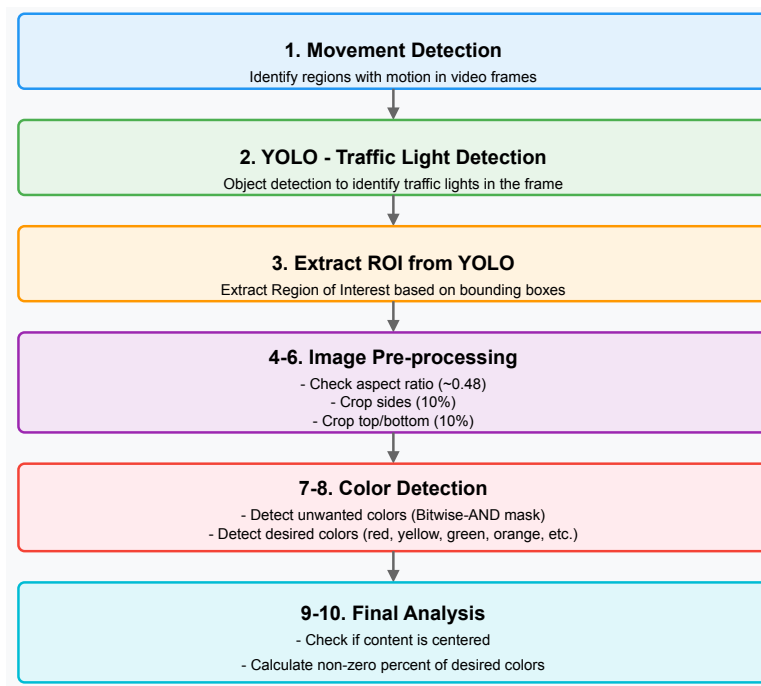


Figure 5.3: Approach for Semi-automatic annotation

the traffic lights on the streets. However, the (street) traffic light is simpler as it has fewer states. However, learned models such as YOLO (described in 3.6 section) that are pre-trained on coco datasets could be useful for railway signal detection. The semi-automatic approach is described in the picture 5.3.

5.3.1 Region of Interest Detection

The Region of Interest (ROI) detection methodology follows a systematic approach to identify and isolate traffic signals within image frames. First, movement detection is applied to identify areas of change in consecutive frames. Subsequently, a YOLO-based object detection algorithm specifically trained for traffic light recognition is

used to localize potential signals. From these detections, the ROI is extracted and

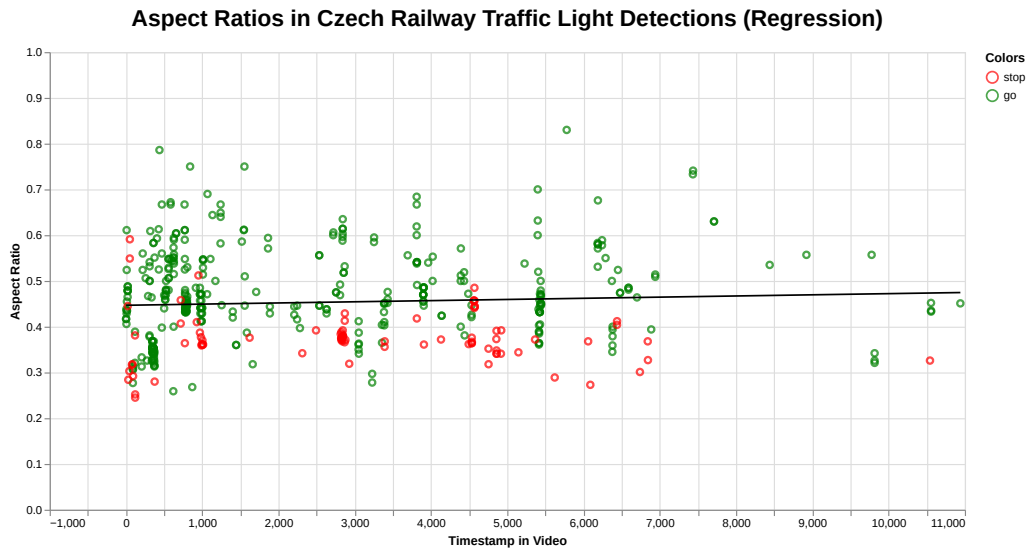


Figure 5.4: Aspect ratio analysis for stop and go states

subjected to geometric validation by examining the aspect ratio, which according to our analysis in Figure 5.4 averages approximately 0.48 between the fore signals and the main signals, where 0 represents a linear shape and 1 represents a square. The extracted ROI undergoes further refinement through proportional cropping, with 10% removed from each side and 10% removed from the top and bottom to eliminate peripheral noise. Color detection is then performed using bit-wise AND operations between masks and the original image, identifying both unwanted colors (red, yellow, green, orange, yellow-orange, black, white) and desired colors within the same spectrum. The spatial distribution of the detected colors is analyzed to verify if the content is properly centered within the ROI. Finally, the percentage of nonzero pixels representing desired colors is calculated to determine the prominence of the traffic signal within the extracted region.

5.3.2 Manual Corrections

The data should be processed by a multiple stage annotation process. Firstly, an automatic annotation pipeline could generate initial labels. Next, it should be manually checked and then corrected, for example, using the Computer Vision Annotation Tool (CVAT). CVAT is an interactive annotation platform for videos and images specifically designed for computer vision applications [CVA23].

In this chapter, the data analysis and methodological approaches used in the development of computer vision systems for the detection and classification of railway signals in the Czech railway are described.

6.1 Data Resources

This section presents an exploratory data analysis of publicly available video resources to train machine learning models in the detection of railway traffic signals. The resources are from the YouTube channels:

- **Parníci CZ** ([link](#))
- **Strojvedouci COM** ([link](#))
- additional – **Pohled z vlaku** ([link](#))

The corpus consists of visual data captured at different frame rates: the **Parníci CZ** and the **Strojvedouci COM** videos are recorded at **60 frames per second**, while the **Pohled z vlaku** videos are captured at **50 frames per second**. All sources are in **full HD** resolution (1920×1080 pixels) due to the trade-offs between video quality and storage size. The corpus has an optimal spatial-temporal density for feature extraction in signal detection algorithms. This technical configuration can use edge detection and color segmentation processes that are useful for railway signal classification tasks while at the same time enabling optical flow computation for motion analysis. The corpus is suitable for both traditional computer vision pipelines and deep learning architectures, such as convolution neural network training through standardized input dimensionality or Real-Time Detection Transformer. All statistics and metadata are stored on **GitHub repository** ([link](#)) in `railway_datasets` folder. The following list consists of notes and explanations for the tables presented below.

1. **Video title** – simplified for readability
2. **Len** – Duration in hours:minutes format
3. **Avg** – Average brightness (scale 0-255), where 0 is black/dark and 255 is maximal illumination
4. **Var** – Temporal variation (how much the brightness changes over time)

6.1.1 Cabview Videos

The **Cabview** videos are part of the corpus of railway footage from the Czech transportation network. Statistical analysis reveals the following technical characteristics pertinent to computer vision applications:

Table 6.1: Statistics from Strojvedouci COM videos at YouTube channel

Video Title	Len	Avg	Var
Cabview z vlaku 3 Rosice n L-Vojtěchov	0:44	117.91	9.17
Cabview z vlaku 2 Žďárec u Skutče-Slatiňany	0:20	117.39	25.31
Cabview z vlaku 6 Pardubice-Hlinsko	1:05	114.53	18.39
Cabview z vlaku 7 Holice-Chrudim	0:46	108.69	18.92
Cabview z vlaku 8 Čáslav-Třemošnice	1:02	108.18	17.20
Cabview z vlaku 14 Pardubice-Holice-Týniště	1:52	96.24	14.69
Cabview 16 Pardubice-Týniště n O-Hradec Králové	1:30	84.46	12.89
Cabview 21 RegioFox Havlíčkův Brod-Pardubice	1:50	94.77	21.60
Cabview 15 Hradec Králové-Chocẽ-Moravany	1:31	94.26	19.43
Cabview 19 Pardubice-Polička duben 2023	1:45	121.70	12.74
Cabview 17 Chrudim-Havlíčkův Brod-Chrudim	3:06	107.52	23.19
Cabview 18 Pardubice-Polička trať 238 261	1:46	94.59	16.34

- **Temporal Extent:** The collection comprises 12 videos with a cumulative duration of **17 hours** 18 minutes 29 seconds, with individual videos ranging from 20 minutes 24 seconds to 3 hours 6 minutes 15 seconds (mean duration: 1 hour 26 minutes 32 seconds).
- **Luminance Distribution:** Videos exhibit mean brightness values ranging from 84.46 to 121.70 (scale 0-255), with an **average brightness of 105.02** ($\sigma = 11.35$) across the videos, providing various illumination conditions for model training.
- **Temporal Variation:** The videos demonstrate considerable variation in frame-to-frame brightness fluctuation ($\mu = 17.49$, $\sigma = 4.47$), with values rang-

ing from 9.17 to 25.31, may need image stabilization pre-processing for consistent feature detection.

- **Information Entropy:** Histogram entropy measurements indicate moderate to high information content ($\mu = 7.63$, $\sigma = 0.12$), with values ranging from 7.40 to 7.78, suggesting a high visual complexity needed for feature extraction tasks.

6.1.2 Parníci CZ Videos

The **Parníci CZ** videos have complementary content with distinct technical properties:

Table 6.2: Statistics from **Parníci CZ** videos at YouTube channel

Video Title	Len	Avg	Var
4K Nýřany - Dioss 3100 Kafemlejnek	0:11	96.60	10.61
4K Týn nad Vltavou - Čičenice 810 RegioMouse	0:32	97.62	13.15
4K Radnice - Plzeň 842 Kvatro	0:44	97.60	20.60
4K Plzeň - Plasy Regioshark 844	0:42	104.65	12.23
4K Bezdrůžice - Plzeň 842 Kvatro	1:03	86.76	14.75
4K Týn n Vltavou - Protivín 814 RegioNova	0:47	130.20	10.27
4K Broumov - Meziměstí 854 Hydra + 954	0:15	109.53	4.75
4K Kolín - Rataje n Sázavou 814 RegioNova	0:53	104.89	15.14
4K Leděčko - Zruč nad Sázavou 814 RegioNova	0:48	94.43	18.68
4K Zruč nad Sázavou - Kutná Hora 814 RegioNova	1:10	111.75	12.20
4K Čáslav - Třemošnice 810 Šukafon	0:28	120.79	7.45
4K Chornice - Velké Opatovice - Skalice	1:57	99.19	9.51
4K Příkosice - Rokycany 814 RegioNova	0:21	121.70	3.74
4K Rokycany - Příkosice 814 RegioNova	0:23	125.55	6.83

- **Temporal Characteristics:** The collection consists of 14 videos with a total duration of **10 hours** 10 minutes 55 seconds, ranging from 10 minutes 37 seconds to 1 hour 56 minutes 46 seconds (mean duration: 43 minutes 38 seconds).
- **Luminance Metrics:** The videos exhibit mean brightness values from 86.76 to 130.20, with an **average of 107.23** ($\sigma = 12.65$), offering slightly higher overall luminance compared to the Cabview corpus.
- **Stability Characteristics:** Temporal variation measurements ($\mu = 11.42$, $\sigma = 4.72$) indicate generally more stable recording conditions than **Cabview** videos, with values ranging from 3.74 to 20.60.

- **Information Content:** Histogram entropy values ($\mu = 7.66$, $\sigma = 0.10$) demonstrate consistent visual complexity in videos, with measurements ranging from 7.41 to 7.80.

6.1.3 Pohled z vlaku Videos

The **Pohled z vlaku** videos offer another valuable perspective with distinctive technical characteristics:

- **Temporal Extent:** The collection comprises 14 videos with a cumulative duration of **22 hours** 37 minutes 14 seconds, with individual videos ranging from 40 minutes to 2 hours 41 minutes (mean duration: 1 hour 33 minutes 57 seconds).
- **Luminance Distribution:** Videos exhibit mean brightness values ranging from 76.32 to 124.13 (scale 0-255), with an **average brightness of 100.60** ($\sigma = 13.30$) across the videos, offering various illumination conditions for robust model training.
- **Temporal Variation:** The videos demonstrate significant variation in frame-to-frame brightness fluctuation ($\mu = 17.79$, $\sigma = 6.38$), with values ranging from 9.96 to 24.91, suggesting dynamic recording conditions that could benefit the validation of signal detection algorithm.
- **Information Entropy:** Histogram entropy measurements indicate a high content information ($\mu = 7.58$, $\sigma = 0.15$), with values ranging from 7.21 to 7.81, reflecting rich visual complexity that provides ample features for machine learning extraction tasks.

6.1.4 Discussion

The following list contains combined statistics computed among the three YouTube channels:

- **Temporal Coverage:** The aggregated corpus has approximately **50 hours** 7 **minutes** of railway operation footage, offering a substantial data volume for model training and validation procedures.
- **Environmental Diversity:** The integrated videos span diverse illumination conditions (**76.32-130.20 brightness range**) and temporal dynamics (3.74-25.31 variation range, excluding outliers), facilitating robust model generalization across varying operational scenarios.

Table 6.3: Statistics from **Pohled z vlaku** videos at YouTube channel

Video Title	Len	Avg	Var
Praha hl nádraží - Rudná u Prahy (link)	0:57	105.85	22.55
České Budějovice - Příbram (link)	1:47	93.41	10.04
Plzeň hl nádraží - Furth im Wald (link)	1:14	94.57	13.79
Praha-Holešovice - České Budějovice (link)	2:07	95.44	24.91
Praha hl n - Plzeň hl n (link)	1:17	76.32	16.22
Praha hl n - Hradec Králové hl n (link)	1:51	114.79	15.58
Ústí nad Labem západ - Kolín (link)	2:16	102.52	9.96
Příbram - Praha hl n (link)	1:36	80.91	21.63
České Budějovice - Plzeň hlavní nádraží (link)	2:00	115.19	16.15
Dečín hl n - Dresden Hbf (link)	0:40	115.19	16.50
Klatovy - Železná Ruda-Alžbětín (link)	1:08	106.16	21.17
Praha hl n - Tanvald (link)	2:41	124.13	21.40
Brno hlavní nádraží - Golčův Jeníkov (link)	2:38	103.82	20.91
Liberec - Stará Paka (link)	1:16	88.27	13.22

- **Frame Extraction Potential:** With varying acquisition rates (**50-60fps**), the videos produce approximately **10.23 million individual frames**, presenting a substantial pool for strategic sampling and annotation protocols.
- **Anomalous Cases:** Three identified outlier videos exhibit atypical characteristics, including **extreme brightness variation (41.28)** and **non-standard frame rates (24-30fps)**, providing valuable edge cases for model robustness evaluation.

Table 6.4: Outliers

Video Title	Len	Avg	Var	FPS
Cabview z vlaku 5 Havlíčkův Brod-Chotěboř	0:20	113.15	8.29	≈ 50
Cabview z vlaku 13 první sníh Pardubice-Hlinsko	2:13	91.72	41.28	30
Cabview 20 Pardubice-Slatiňany-Pardubice	0:51	96.96	25.04	≈ 24

All three data sources are suitable for machine learning applications with complementary characteristics. The **Cabview** videos offer extended duration and higher temporal variation, the **Parníci CZ** videos provide superior resolution and more consistent recording conditions, while the **Pohled z vlaku** videos contribute a significant volume of footage with diverse environmental conditions at a standard 50fps frame rate. The combination of these resources with varying frame rates (50-60fps) and illumination conditions creates a comprehensive dataset for the development of robust railway signal detection algorithms.

6.2 Synthetic Image Dataset

Synthetic image generation was explored as an alternative approach to create a dataset, but faced challenges when using black-forest-labs generators via the Hugging Face API. Despite clear prompts, the models consistently produced traffic lights intended for automobiles rather than railway signals, demonstrating the limitations of current generative AI in domain-specific visual tasks.

6.3 Preliminary Experiments

The preliminary experiments focused on timestamp extraction from railway video footage using object detection techniques. YOLOv5 was selected as the initial model due to its balance of speed and accuracy for real-time applications. The experiments were based on a diverse dataset of Czech railway videos collected from the YouTube channel **Parníci CZ**, all in full HD resolution (see Section 6.1 for more details), but with varying lighting conditions, weather conditions and recording quality.

6.3.1 Detection Performance

The initial results with the pre-trained YOLOv5mu model had quite poor detection possibilities for the general **traffic light** class. The model detected **10895** moments of the **traffic light** class in videos. This was a state, when the movement detection was not applied. This, a fine-tuning is required for the specific characteristics of

Table 6.5: Initial performance metrics for pre-trained YOLOv5 on Czech railway signals

Metric	Moments in videos	Acc on human filt.
YOLOv5m	10895	0.078
YOLOv10m	10404	0.08
YOLOv5m & movement detection	6485	0.13
Filtered by human	848	–

Czech railway signaling systems. One particular note was the difficulty in distinguishing between closely spaced signal aspects in complex junction scenarios.

Implementation

7

This chapter presents details of the implementation of the computer vision system for the detection and classification of railway signals in the Czech railway network. It is based on the methodological approaches described in Section 5, focusing on the practical aspects of system development, data management, and architecture. The whole implementation can be found in **GitHub repository** ([link](#)).

7.1 Dataset Storage

The dataset was organized and stored using a structured cloud. All data assets were maintained in a dedicated **Google Drive** ([link](#)) repository for collaborative development and ensured consistent access to training and testing materials at all stages of the implementation process.

7.2 Metadata Structure

A lightweight JSON metadata format was used to effectively manage detection instances while optimizing storage requirements.

Metadata are stored in the `railway_datasets` folder. Each detection instance was characterized by the following attributes:

- **ID**: Unique identifier for each detection instance
- **aspect ratio**: Width-to-height ratio of the source video
- **video name**: Name of the source video file
- **detection method**: Algorithm used for detection (e.g. "yolov5mu")
- **class**: Object class (e.g. "traffic light")
- **timestamp in video**: Time position in seconds within the source video
- **color**: State of the traffic light (e.g. "stop", "go", "adjust speed and warning")

- **roi coordinates:** Region of interest in normalized format: "x y width height" where x,y is the center of the box
- **ytlink:** YouTube link to the source video

7.3 Dataset Generation & Reconstruction Scripts

Specialized scripts were developed for the creation and reconstruction of datasets. They are in `dataset_scripts` folder:

```
1 convert_cleanup
2     convert_to_coco_json.py
3     ...
4     update_metadata.py
5     yolo_to_roi_format.py
6 explorative_data_analysis
7     dataset_statistics.py
8     ...
9     extract_video_statistics.py
10 images_extraction
11     process_more_videos_with_yolo.py
12     report_results.py
13 reconstruction_scripts
14     annotate_colors_naive.py
15     prepare_augmented_coco_format.py
16     ...
17     reconstruct_dataset.py
18     yolo_minority_class_augmentation.py
19 test_and_preview
20     create_video_sequences.py
21     ...
22     model_evaluation.py
23 visualizations
24     visualization_aspect_ratio_based_on_colors.json
25     visualization_aspect_ratio_based_on_videos.json
```

These tools automated the process of extracting frames from video sources, applying detection algorithms, and generating structured datasets suitable for model training. The scripts can reconstruct training data from metadata, significantly reducing storage requirements while maintaining dataset integrity.

7.4 Traditional Computer Vision Components

Several traditional computer vision algorithms were implemented to support the detection and processing pipeline.

- **Movement Detection:** Implemented in `movement_detection.py`, this module identified regions with motion across consecutive video frames by using the perceptual hash similarity algorithm described in Section 3.1.2 to reduce the search space for subsequent detection processes.
- **Crossing Detection:** The `crossing_detection.py` module identified railway crossings, providing contextual information for signal classification.
- **White Triangle Detection:** Modules such as `detect_white_triangles.py` and `single_white_triangle_detection.py` were developed to detect specific railway signaling elements that give a warning prior to the fore signal or the main signal.
- **Non-Traffic Light Identification:** The `non_traffic_light_identification.py` module filtered out false positives by identifying objects that resembled but were not railway signals.

These components formed a preprocessing pipeline that enhanced the accuracy of the deep learning models by reducing the search space and filtering irrelevant detections. In the project repository, there are more features implemented in `cv_experiment_scripts`:

```
1 crossing_detection.py
2 detect_white_triangles.py
3 generating_railway_signals_rpc.py
4 gmm_clustering.py
5 movement_detection.py
6 non_traffic_light_identifiaction.py
7 process_video_with_naive_ocr.py
8 railway_detection.py
9 single_white_triangle_detection.py
```

7.5 Neural Network Implementation

The neural network implementation followed single-stage and two-stage approaches, as described in the methodology chapter.

7.5.1 Detection Models

Multiple state-of-the-art object detection architectures were implemented and evaluated:

- **YOLOv5 Variants:** Including YOLOv5mu and YOLOv5nu, these models provided a baseline for single-stage detection.
- **YOLOv8 and YOLOv10:** More recent architectures (YOLOv8n, YOLOv10m, YOLOv10n) were also used for fine-tuning.
- **RT-DETR:** Real-Time Detection Transformer architectures (RT-DETR-l, RT-DETR-x) were also used as models for fine-tuning.

Each detection model was adapted to the specific characteristics of the railway signals, with particular attention to the small object size common to the railway signaling systems. The experiment scripts and configurations are stored in `metacentrum_experiments` directory:

```
1 metacentrum_experiments
2 CRL_extended.yaml
3 .
4 .
5 .
6 CTRL_multi_labeled_transfer.yaml
7 fine_tune_yolo.py
8 plan_CRL_extended.sh
9 plan_CRL_single_images_less_balanced.sh
10 plan_multi_labeled.sh
11 plan_rtdetr_CRL_single_images.sh
12 train_on_meta.sh
```

7.5.2 Classification Models

For the classification component, specialized neural networks were implemented:

- **CzechRailwayLightsNet:** A custom neural network architecture implemented in `czech_railway_lights_net.py` and its variants, designed specifically for the classification of Czech railway signals, it is described in more detail in Section 8.4.
- **Combined Model:** The `combined_model.py` implementation integrated detection and classification components in a unified inference pipeline.

The classification models were trained to distinguish between different signal states as defined in the methodology chapter, including stop, go, caution, speed adjustment warnings, and other relevant states.

7.6 Training Scripts

Training scripts were implemented in directory `classification_experiments`:

```
1 CzechRailwayLightClassifier.yaml
2 combined_model.py
3 czech_railway_light_detection_backbone
4     detection_backbone
5         weights
6             best.pt
7 czech_railway_lights_net.pt
8 czech_railway_lights_net.py
9 draft_combined_model_export.py
10 train.py
11 visualize.py
```

The `train.py` script provided a unified interface for training classification models, with support for various hyperparameter configurations. The training scripts integrated TensorBoard logging for real-time performance monitoring, with logs stored in the `logs` directory. The model checkpoints were saved at regular intervals, allowing the selection of optimal model weights based on validation metrics. Several visualization tools were implemented to analyze model performance and internal representations:

- **Model Visualization:** The `visualize.py` script generated architectural visualizations of the neural network models.
- **Feature Maps:** Tools for visualizing intermediate feature maps that provide insight into the internal representations learned by the models.
- **Confusion Matrices:** Automated generation of confusion matrices for evaluating classification performance across different signal states.

These visualization tools were used to observe model performance issues and identify potential improvements to the architecture or training process.

In this chapter, a systematic evaluation of different model configurations and approaches for the detection and classification of the railway signal is presented. The [Met] was used as computational resources to perform almost all hyperparameter grid searches.

Experimentation was performed with various state-of-the-art object detection architectures: YOLOv10m, YOLOv10n, YOLOv5mu, YOLOv8n, RT-DETR-l, RT-DETR-x, and YOLOv5nu. The hyperparameter optimization strategy consisted of different freezing levels (0 and 3), training epochs (15, 30, 60, 80, and 100) and confidence thresholds (0.3, 0.5, and 0.7) to determine the optimal configuration for the detection performance of the railway signal.

8.1 Basic Detection Results

In this section, the results are presented for various detections. They are ordered from basic single-class detection to more sophisticated multi-feature approaches.

8.1.1 Railway Light Detection Performance

There are only detections of railway signals, without classification of their states. All railway signals are grouped into a single class.

- **railway_signal** (Class 0): Any railway signal regardless of its state.

The initial minimalist approach focused on the detection of railway signals achieved the detection rates presented in Table 8.1. The models were evaluated using different configurations, varying in model architecture, training steps, frozen layers, and confidence thresholds. The dataset contains **approximately 800 instances** of railway signals with varying positions throughout the image frame. Most railway signals appear in the central part of the images ($x=0.4-0.6$, $y=0.3-0.4$) with relatively small bounding boxes (width=0.01-0.04, height=0.02-0.10). In the following list, observations from analysis of the results are presented.

Table 8.1: Best Grid Search Results: Railway Signal Detection Performance

Model	Steps	Frozen	Conf	mAP@0.5	Precision	Recall
YOLOv8n	100	0	0.3	0.7994	0.9407	0.8006
YOLOv8n	100	0	0.5	0.7956	0.9476	0.7967
YOLOv5mu	100	0	0.7	0.7688	0.9781	0.7693
YOLOv5nu	30	0	0.3	0.8098	0.9558	0.8089
YOLOv8n	30	0	0.5	0.7783	0.9797	0.7829
YOLOv8n	30	3	0.7	0.7698	0.9826	0.7693
YOLOv8n	80	0	0.5	0.7922	0.9605	0.7920

- **Model Architecture:** YOLOv8n models demonstrated better overall performance in detecting railway signals, achieving the highest mAP scores while requiring significantly less training time compared to other architectures.
- **Frozen Layers:** Models trained with 0 frozen layers (indicated by Frozen = 0) generally outperformed those with 3 frozen layers, suggesting that full adaptation of the model to the specific visual characteristics of the railway signals is beneficial.
- **Confidence Thresholds:** Lower confidence thresholds (0.3) tended to yield better overall detection metrics, while higher thresholds (0.7) produced superior precision at the cost of potentially lower recall.
- **Training Efficiency:** The YOLOv5nu configuration with 30 steps, 0 frozen layers and 0.3 confidence threshold achieved the highest mAP score (0.8098), making it particularly suitable for this railway signal detection task.

8.1.2 Dual Category Railway Light Detection

This is a simplified version of the detection dataset, which contains only two classes: railway signals and other objects. The results were not very satisfactory, with an approximate 0.6 F1 score.

8.1.3 Multi-Feature Railway Signal Detection

This dataset is focused on the detection task with four distinct classes of railway signals.

- **other (Class 0):** Non-signal objects or background elements.
- **railway_signal (Class 1):** Active railway signals displaying any state.
- **signal_back (Class 2):** Railway signals viewed from behind.

- **signal_off (Class 3):** Railway signals that are turned off.

The expanded system achieved the results described in the table 8.2. This approach did not perform very well.

Table 8.2: Multi-Feature Railway Signal Detection Performance

Model	Steps	Frozen	Conf	mAP@0.5	Precision	Recall	Dataset Type
YOLOv5mu	150	0	0.3	0.4738	0.5711	0.5927	Sequences
YOLOv5mu	80	0	0.3	0.4865	0.6183	0.6144	Sequences
YOLOv5mu	120	0	0.3	0.4599	0.5841	0.5829	Sequences
YOLOv5mu	200	3	0.3	0.4435	0.5517	0.5984	Sequences
YOLOv5x	40	5	0.3	0.4845	0.7260	0.4861	Single frames
YOLOv5nu	60	3	0.3	0.5232	0.7015	0.5404	Single frames
YOLOv5nu	60	3	0.5	0.5120	0.7633	0.4426	Single frames
YOLOv10x	40	0	0.3	0.4214	0.6386	0.5069	Single frames
YOLOv10m	60	3	0.3	0.4139	0.6615	0.4755	Single frames
YOLOv10m	40	5	0.3	0.4034	0.6761	0.4239	Single frames

8.2 Preliminary Detection & Classification Results

The initial experiments were not very successful, but their results led to more sophisticated approaches.

8.2.1 Binary State Classification

The basic stop/go classification performed very poorly due to the lack of training data. In transfer learning the results were similar to models trained on the COCO dataset.

- **stop (Class 0):** Railway signals displaying a stop sign (red light).
- **go (Class 1):** Railway signals displaying a go sign (green light).

8.2.2 Basic Signal State Classification

This dataset combines detection and classification tasks, focusing on four main signal states.

- **stop (Class 0):** Railway signals displaying a stop sign (red light).
- **go (Class 1):** Railway signals displaying a go sign (green light).

- **warning_go (Class 2):** Railway signals displaying a combined warning and a go signal.
- **signal_off (Class 3):** Railway signals that are turned off.

The results revealed many issues. The best performing model (YOLOv5mu) achieved only 0.431 mAP50-95, indicating substantial room for improvement. Most models struggled with precision, with the highest value of 0.681 achieved by YOLOv10n.

Table 8.3: Basic Signal State Classification Performance

Model	mAP50-95	Precision
YOLOv5mu	0.431	0.508
YOLOv8n	0.359	0.525
YOLOv10m	0.341	0.652
YOLOv10n	0.325	0.681
YOLOv5nu	0.304	0.446
RT-DETR-L	0.334	0.490

8.2.2.1 Transfer Learning

This dataset is the same as before, but uses transfer learning from a pre-trained model with 80 COCO classes plus the 4 railway-specific classes. Classes 0-79 are standard COCO dataset classes, with railway-specific classes added:

- **stop (Class 80):** Railway signals displaying a stop signal.
- **go (Class 81):** Railway signals displaying a go signal.
- **warning_go (Class 82):** Railway signals displaying a combined warning and a go signal.
- **signal_off (Class 83):** Railway signals that are turned off.

Transfer learning experiments provided results similar to the basic classification approach, suggesting that the pre-trained weights did not provide significant advantages for this specialized domain. The challenging nature of railway signal classification likely requires more domain-specific training data and possibly specialized model architectures tailored to the unique characteristics of railway signaling systems.

8.3 Detection & Classification with Environmental Context

One of the main Czech railway light datasets contains images of railway traffic lights with multiple classification classes. It is a dataset designed to recognize various states of railway signals. These classes are selected on the basis of visual analysis, in Section 5.1.3

- **stop (Class 0):** Single red light indicating a stop signal.
- **go (Class 1):** Single green light indicating a go signal.
- **warning (Class 2):** Single yellow light indicating a warning signal.
- **adjust speed and warning (Class 3):** Two yellow lights displayed together, indicating that the train should adjust speed and prepare for a warning.
- **adjust speed and go (Class 4):** Combination of upper green light and lower yellow light, indicating that the train should adjust speed while proceeding.
- **lights off (Class 5):** Traffic lights that are turned off.
- **crossing light (Class 6):** Signals used specifically for railway crossings.
- **lights behind (Class 7):** Railway signals viewed from behind.

The dataset has a few variants. They are stored on the cloud mentioned in Section 7.2. The first presented is a single image dataset.

8.3.1 Single Images Dataset

This dataset contains only one frame for each detection moment and is not extended by the CVAT tool (described in Section 5) tool. The class distribution is the following:

- **stop (Class 0):** over 80 samples
- **go (Class 1):** around 50 samples
- **warning (Class 2):** around 20 samples
- **adjust speed and warning (Class 3):** around 20 samples
- **adjust speed and go (Class 4):** around 35 samples
- **lights off (Class 5):** around 65 samples
- **crossing light (Class 6):** less than 10 samples

- **lights behind (Class 7)**: around 5 samples

The values are from `experiment_results/yolo/CRL_single_image/labels.jpg` on the github project repository. The experiments evaluated multiple object de-

Table 8.4: Grid search Results

Model	Epochs	Frozen	Conf	IoU	mAP50	mAP50-95	Precision	Recall
YOLOv8n	100	0	0.3	0.5	0.5694	<u>0.4876</u>	0.5456	0.5569
YOLOv8n	100	3	0.3	0.5	0.5694	<u>0.4876</u>	0.5456	0.5569
YOLOv10m	120	0	0.3	0.5	0.5989	0.4846	0.8445	0.5849
YOLOv10m	120	0	0.7	0.5	0.5989	0.4846	0.8445	0.5849
YOLOv10m	120	3	0.3	0.5	0.5740	0.4584	0.8753	0.5408
YOLOv10n	150	0	0.3	0.5	0.5143	0.4195	0.7211	0.5306
YOLOv10n	120	0	0.7	0.5	0.5193	0.4155	0.7500	0.4712
YOLOv10n	120	3	0.7	0.5	0.5193	0.4155	0.7500	0.4712
YOLOv5mu	150	0	0.5	0.5	<u>0.5884</u>	0.5015	0.6162	<u>0.5653</u>
YOLOv5mu	150	0	0.3	0.5	0.5860	<u>0.4977</u>	0.6060	0.5660
YOLOv5mu	120	3	0.7	0.5	0.5591	0.4730	0.6428	0.5009
YOLOv5nu	120	0	0.5	0.5	0.5139	0.4367	0.5793	0.4526
YOLOv5nu	120	0	0.7	0.5	0.5080	0.4312	0.6050	0.4252
RT-DETR-L	100	0	0.3	0.5	0.5570	0.4544	<u>0.8636</u>	0.5561
RT-DETR-L	100	3	0.3	0.5	0.5403	0.4448	0.7360	0.5482
RT-DETR-X	100	3	0.3	0.5	0.5141	0.4202	0.6719	0.5022

tection models on the Czech Railway Lights Dataset with varying configurations. **YOLOv10m** (120 epochs, 0 frozen layers) achieved the highest mAP50 score (**0.5989**) and recall (**0.5849**), while **YOLOv5mu** (150 epochs, 0.5 confidence) delivered the best mAP50-95 performance (**0.5015**). **YOLOv10m** with 3 frozen layers demonstrated superior precision (**0.8753**), closely followed by **RT-DETR-L** (0.8636). The results suggest that the frozen layers were generally preserved or slightly reduced in performance while potentially improving training efficiency. Higher confidence thresholds (0.7) maintained precision but sometimes reduced recall compared to lower thresholds (0.3). In general, the configurations **YOLOv10m** and **YOLOv5mu** emerged as optimal choices for the railway signal detection task, with the selection depending on whether precision, recall, or balanced performance are prioritized.

8.3.1.1 More Balanced Variant

There is also a variant that contains a subset of the classes in the complete dataset, with a less unbalanced distribution of samples between classes. This dataset has a subset of the classes from the complete dataset, with a more balanced distribution of samples among the classes. The dataset is structured with dedicated train

and validation image directories in Google drive storage mentioned in Section 7.2. Initial experiments with this variant produced **unsatisfactory results** that were not included in the final analysis. This variant contains six primary railway signal classes, excluding the less common crossing lights and behind-view signals:

- **stop (Class 0):** Single red light indicating a stop signal.
- **go (Class 1):** Single green light indicating a go signal.
- **warning (Class 2):** Single yellow light indicating a warning signal.
- **adjust speed and warning (Class 3):** Two yellow lights together.
- **adjust speed and go (Class 4):** Combination of upper green light and lower yellow light.
- **lights off (Class 5):** Traffic lights that are turned off.

8.3.2 Enhanced Classification with Environmental Context

This dataset contains multiple frames for each detection and is extended by the CVAT (described in Section 4.7). The class distribution is in the following list. The results are taken from `experiment_results/yolo/CRL_extended/labels.jpg` on the github project repository.

- **stop (Class 0):** over 220 samples
- **go (Class 1):** over 210 samples
- **warning (Class 2):** around 25 samples
- **adjust speed and warning (Class 3):** around 75 samples
- **adjust speed and go (Class 4):** around 100 samples
- **lights off (Class 5):** around 130 samples
- **crossing light (Class 6):** 5 samples
- **lights behind (Class 7):** 3 samples

Based on the grid search, RT-DETR-L has quite good performance with the highest precision 88.2% and 77.5% F1 score, thus establishing itself as the most balanced model among the tested. YOLOv10m appeared to be a close competitor with strong precision capabilities 87.0% and a commendable F1 score 76.5%. Although

YOLOv5mu achieved the highest mAP 62.2% and recall 70.2%. However, the precision 77.0% was identified to be lower than the top models, resulting in a solid but comparatively lower F1 score 73.5%.

Based on the collected data, it can be concluded that RT-DETR-L would be optimally suited for applications where high precision is prioritized. In contrast, YOLOv5mu might be recommended in scenarios where higher recall is required and a certain threshold of false positives can be tolerated. The augmentation strategy incorporated the following parameters:

- **HSV Color Space Modifications:**

- Hue augmentation was set to 0.015
- Saturation augmentation was configured at 0.07
- Value augmentation was established at 0.04

- **Geometric Transformations:**

- Rotation augmentation was minimally applied at 0.001 degrees
- Translation augmentation was implemented at 0.1
- Scale augmentation was set to 0.25
- Shear augmentation was configured at 0.05
- Perspective augmentation was not applied (value of 0.0)

Table 8.5: Best Grid search Results

Model	Epochs	Frozen	Conf	mAP50-95	Precision	Recall	F1	Aug.
YOLOv8n	80	0	0.3	0.6093	0.7447	0.6470	0.6925	Yes
YOLOv10m	80	0	0.3	0.5836	0.8704	0.6818	0.7646	No
YOLOv10n	100	0	0.5	0.5221	0.8534	0.6106	0.7122	No
YOLOv5mu	100	0	0.3	0.6223	0.7701	0.7022	0.7346	No
YOLOv5nu	100	3	0.3	0.6075	0.7282	0.6902	0.7087	Yes
RT-DETR-L	60	0	0.3	0.5897	0.8823	0.6902	0.7747	Yes

It was observed that model augmentation generally resulted in performance improvements in various metrics. This enhancement was particularly notable in RT-DETR-L, where an exceptional precision-recall balance was achieved with augmentation enabled. The models with augmentation (YOLOv8n, YOLOv5nu, and RT-DETR-L) demonstrated more robust performance characteristics compared to their non-augmented counterparts.

8.3.2.1 Extended by Pohled z Vlaku Videos

This variant contains multiple frames for each detection and is extended by the CVAT (described in Section 4.7). and **Pohled z Vlaku** videos. The class distribution is in the following list. The results are taken from:

experiment_results/yolo/CRL_extended_v2//labels.jpg*
on the github project repository.

- **stop (Class 0)**: over 590 samples
- **go (Class 1)**: over 380 samples
- **warning (Class 2)**: around 150 samples
- **adjust speed and warning (Class 3)**: around 120 samples
- **adjust speed and go (Class 4)**: around 200 samples
- **lights off (Class 5)**: around 590 samples
- **crossing light (Class 6)**: - samples
- **lights behind (Class 7)**: - samples

From the results in the table 8.6, YOLOv5nu had an optimal detection quality with the highest mAP50-95 (0.7243). The basic detection capability was led by YOLOv10n (mAP50: 0.8626), although the localization precision was found to be lacking. YOLOv5mu had the best precision performance (0.9459), making false positives significantly minimized. Maximum recall values (0.8049) were obtained with RT-DETR-L, although at the cost of 2-3 times longer training durations. The learning rates (0.001) were consistently associated with better performance.

Table 8.6: Grid Search Results

Model	Epochs	Frozen	Conf	Lr	mAP50	mAP50-95	Precision	Recall	F1
YOLOv5nu	60	0	0.5	0.001	0.8317	0.7174	0.8991	0.7293	0.8055
YOLOv5nu	100	0	0.5	0.0001	0.8451	0.7185	0.8985	0.7703	0.8295
YOLOv8n	80	0	0.5	0.001	0.8403	0.7160	0.9204	0.7377	0.8191
YOLOv5mu	60	0	0.5	0.001	0.8422	0.7192	0.9254	0.7316	0.8174
YOLOv5mu	80	0	0.5	0.001	0.8456	0.7145	0.9450	0.7368	0.8280
YOLOv8n	100	0	0.5	0.001	0.8334	0.7127	0.8855	0.7690	0.8231
YOLOv5nu	30	0	0.5	0.001	0.8174	0.6973	0.9072	0.6982	0.7890
YOLOv5mu	100	0	0.5	0.001	0.8277	0.7112	0.9459	0.6904	0.7983
YOLOv8n	60	0	0.5	0.001	0.8293	0.7130	0.9260	0.7186	0.8093
YOLOv5nu	80	0	0.5	0.001	0.8375	0.7093	0.9129	0.7332	0.8136
YOLOv5nu	100	0	0.5	0.001	<u>0.8560</u>	0.7243	0.9113	<u>0.7934</u>	0.8486
YOLOv5nu	80	0	0.5	0.0001	0.8142	0.6961	0.9192	0.7030	0.7969
YOLOv5mu	30	0	0.5	0.001	0.8271	0.6853	0.9298	0.7067	0.8035
YOLOv10n	100	0	0.5	0.001	0.8626	0.6891	0.8667	0.7586	0.8090
YOLOv8n	80	0	0.5	0.0001	0.8076	0.6866	0.8795	0.7119	0.7871
RT-DETR-L	60	0	0.5	0.001	0.8230	0.6793	0.8734	0.8049	0.8378
YOLOv8n	100	0	0.5	0.0001	0.8057	0.6833	0.8922	0.6944	0.7807
YOLOv10n	80	0	0.5	0.001	0.8242	0.6641	0.8580	0.7185	0.7819
YOLOv8n	30	0	0.5	0.001	0.7774	0.6510	0.8674	0.6668	0.7540
RT-DETR-L	100	0	0.5	0.001	0.7999	0.6424	0.8987	0.7911	<u>0.8414</u>
YOLOv10n	60	0	0.5	0.001	0.7949	0.6390	0.8015	0.7145	0.7553
RT-DETR-L	100	0	0.5	0.0001	0.7952	0.6363	0.8782	0.7606	0.8152
RT-DETR-L	80	0	0.5	0.0001	0.7527	0.6259	0.8881	0.7614	0.8199
RT-DETR-L	30	0	0.5	0.001	0.7510	0.6084	0.8830	0.7528	0.8130
YOLOv5nu	30	0	0.5	0.01	0.6695	0.5668	0.7319	0.5872	0.6517
YOLOv10n	60	0	0.5	0.01	0.6982	0.5587	0.7141	0.6262	0.6671
YOLOv10n	30	0	0.5	0.01	0.6679	0.5369	0.8466	0.5580	0.6730

8.4 Classification from ROI

In this section, the results of classifying railway signals from their Regions of Interest (ROI) are presented. Train pictures have different aspect ratios, which presents additional challenges for classification. The dataset used for this classification task has the following distribution:

- **stop**: 401 samples
- **go**: 399 samples
- **warning**: 404 samples
- **adjust speed and warning**: 337 samples

- **adjust speed and go:** 387 samples
- **lights off:** 398 samples

The classification model, CzechRailwayLightNet, uses a convolution neural network architecture with the following specifications:

- **Input image size:** 72×34 pixels with 3 channels
- **Convolution layers with channels:** [64, 128, 256]
- **Kernel size:** 3×3
- **Stride:** 2×2
- **Hidden layer size:** 128 neurons
- **Dropout rate:** 0.2

Several approaches were tested to address the class imbalance and aspect ratio variations:

- **Original Dataset:** The dataset with class imbalance (stop: 361, go: 283, warning: 60, adjust speed and warning: 104, adjust speed and go: 138, lights off: 268).
- **Cropped Dataset:** Some samples were removed to reduce class imbalance, resulting in a more balanced distribution (stop: 286, go: 283, warning: 60, adjust speed and warning: 104, adjust speed and go: 138, lights off: 268).
- **Semi-Augmented Dataset:** Targeted data augmentation was applied to classes:
 - Class 2 (warning): Increased from 60 to 240 samples through noise addition, darker images, and brightness/contrast adjustments
 - Class 3 (adjust speed and warning): Increased from 104 to 208 samples through noise augmentation
 - Class 4 (adjust speed and go): Increased from 138 to 276 samples through noise augmentation.
- **Extended Dataset:** An approach combining augmentation techniques with additional data collection to achieve near-perfect balance across all classes.

All initial experiments used 20 training epochs with a validation set size of 15%. The extended dataset experiments used longer training (30-90 epochs) with a more careful hyperparameter tuning.

Table 8.7: Dataset Results Comparison

Experiment	Epochs	Loss	Prec.	F1
Cropped	20	0.162	0.949	0.947
Semi-Augmented	20	0.220	0.963	0.962
Semi-Aug. (LR=0.0001)	20	0.269	0.909	0.906
Extended Dataset	90	0.264	0.952	0.951
Extended (LR=0.0001)	30	0.104	0.977	0.977

8.4.1 Discussion

The original dataset showed strong performance (95.1% F1 score) despite class imbalance, but the targeted augmentation to balance classes further improved the results to 96.2%. Across most experiments, classification errors occurred primarily between:

- **adjust speed and warning** (class 3) and **adjust speed and go** (class 4), which share similar visual features
- **stop** (class 0) and **lights off** (class 5), particularly in low-light conditions
- **go** (class 1) and **lights off** (class 5), especially in experiments with lower learning rates

Targeted augmentation significantly improved the recognition of underrepresented classes. The **warning** class (2) F1 score improved from approximately 82%, for classes 3 and 4 reduced confusion between these similar signal types.

For **the semi-augmented dataset**, reducing the learning rate from 0.001 to 0.0001 degraded the performance by approximately 5.6 percentage points (from 96.2% to 90.6%).

However, for **the extended dataset**, a lower learning rate of 0.0001 combined with longer training (30 epochs) achieved the best overall performance (97.7% F1 score). The most balanced dataset combined with optimized hyper-parameters (learning rate=0.0001, weight decay=0.001, 30 epochs) produced the highest F1 score (97.7%) and the lowest loss (0.104), demonstrating the benefits of data balancing and careful parameter tuning.

The model with the best performance **extended with a lower learning rate** shows minimal confusion between classes. The most common remaining errors were between **go** (class 1) and **lights off** (class 5), with only two mistakes out of 60 samples achieving a 97.7% F1 score, representing a substantial improvement over the original imbalanced dataset.

The confusion matrices indicate that the remaining errors are concentrated in visually similar signal states or in challenging lighting conditions. Future work could focus on enhanced feature extraction for these specific cases.

8.5 Test Performance

The test dataset consists of 404 samples distributed in six classes. The distribution is as follows:

Table 8.8: Distribution of samples across test classes

Class	Count	Percentage
stop	126	31.2%
go	53	13.1%
warning	30	7.4%
adjust speed and warning	26	6.4%
adjust speed and go	38	9.4%
lights off	131	32.4%
Total	404	100%

8.5.1 Two-stage Model

This model integrates the detection **backbone with the classification head** in a two-stage pipeline. The detection stage detects railway signals in the input image, and the classification stage determines the signal state for each detected instance. The model refers to the architecture described in the scheme in Section 5.2.

The detection backbone demonstrated strong performance in locating railway signals (Table 8.9). These metrics indicate that the detection model reliably identifies railway signals with high precision and recall, successfully locating 95.96% of the signals in the test dataset with only 25 false positive detections. The classification

Table 8.9: Overall detection performance metrics

Metric	Value
Precision	0.9417
Recall	0.9596
F1 Score	0.9506
True Positives	404
False Positives	25
False Negatives	17

head showed very poor performance in determining signal states (Table 8.10). The

Table 8.10: Overall classification performance metrics

Metric	Value
Precision	0.6171
Recall	0.4975
F1 Score	0.4673

per-class performance analysis revealed several findings. The model achieves the highest precision for the **stop** class (0.8214), but with a very low recall (0.1797), indicating that while it rarely mistakes other signals for a **stop** signal, it fails to identify many actual **stop** signals. The **lights off** class has the highest recall (0.7591) but moderate precision (0.4444), suggesting that the model tends to overfit signals as **lights off**. The **warning** class showed notably poor precision (0.2167), indicating significant confusion with other signal types. The **go** class displays the most balanced performance with an F1 score of 0.5983.

The combined model demonstrates strong detection capabilities but moderate classification performance. The high detection metrics (F1 = 0.9506) indicate that the model reliably identifies the presence and location of railway signals in the images. However, the classification metrics (F1 = 0.4673) suggest that there is room for improvement in determining the specific signal state. Of particular concern is the low recall for **stop** signals (0.1797), which poses a safety risk in practical applications. This poor performance may be attributed to several factors:

- class imbalance in the training dataset
- limited resolution of cropped signal regions for classification
- the variability in lighting conditions

The variability affected the appearance of the red signal indicators, causing potential confusion between visually similar signal states. Strong detection performance provides a solid foundation for the system, but the classification component requires further refinement to achieve the reliability necessary for safety-critical railway applications.

8.5.1.1 Detection Component

From the best detection models tested in the table 8.1, **YOLOv5nu** was selected as the detection backbone. Its architecture has the configuration described in table 8.11. The detection model was trained with data augmentation techniques:

- random horizontal flips (probability 0.5)
- translation (0.1)

Table 8.11: Detection backbone test configuration parameters

Parameter	Value
Model	YOLOv5nu
Image size	1920x1080 pixels
Batch size	16
Learning rate (initial)	0.01
Learning rate (final)	0.01
Momentum	0.937
Weight decay	0.0005
Confidence threshold	0.5
Intersection over union	0.4

- scaling (0.5)

8.5.1.2 Classification Component

A custom **CzechRailwayLightNet** architecture was implemented as the classification head, with a structure described in table 8.12. The classification model was

Table 8.12: Classification head configuration parameters

Parameter	Value
Input size	72×34 pixels
Convolution channels	[64, 128, 256]
Kernel size	3×3
Stride	2×2
Hidden size	128
Dropout	0.2
Classes	6

designed to process the cropped signal regions from the detection backbone and classify them into one of the six defined signal states.

8.5.2 Single-stage Models

In this study, single-stage detection models were evaluated due to their computational efficiency and real-time processing capabilities. These architectures, which predict bounding boxes and class probabilities in a single forward pass, were selected for their potential application in traffic signal recognition systems where latency is critical. Two representative models, YOLOv8n and RT-DETR, were trained and tested on the custom traffic signal dataset. Performance was measured using standard metrics: precision, recall, and F1 score in all signal classes. Class imbalance

effects were observed in both models, with particular attention paid to the trade-offs between detection accuracy and computational demands.

8.5.2.1 YOLOv8n

The YOLOv8n model was evaluated after training in the test dataset for 120 epochs with a confidence threshold of 0.5 and the IoU threshold of 0.4. The model achieved an overall precision of 0.9370, a recall of 0.8836, and an F1 score of 0.9095, demonstrating strong performance in the test set. The dataset has a notable class imbalance, with the **stop** and **lights off** classes representing more than 63% of the samples. This imbalance appears to have affected the model's performance across classes.

- **High-prevalence classes:** The **stop** class (31.2% of the samples) achieved the highest recall (0.9531) but relatively lower precision (0.8026), indicating that the model tends to overfit this class. The **lights off** class (32.4% of the samples) showed the poorest overall performance with an F1 score of 0.7510.
- **Low-prevalence classes:** Despite their lower representation, classes like **go** and **adjust speed** and **warning** achieved commendable precision scores of 0.9778 and 1.0000 respectively, though with lower recall values.

Precision and recall have some trade-offs in the different classes. The **adjust speed and warning** class achieved perfect precision (1.0000) but moderate recall (0.7407), indicating that while all its positive predictions were correct, the model missed approximately 26% of instances from this class. The **stop** class demonstrated the opposite trend with high recall (0.9531) but lower precision (0.8026), suggesting that the model rarely misses the **stop** signal, but occasionally generates false positives. The error distribution has the following problems. The **lights off** class had

Table 8.13: Detection performance metrics by class

Class	Precision	Recall	F1 Score	TP	FP	FN
stop	0.8026	0.9531	0.8714	122	30	6
go	0.9778	0.7719	0.8627	44	1	13
warning	0.9200	0.7419	0.8214	23	2	8
adjust speed and warning	1.0000	0.7407	0.8511	20	0	7
adjust speed and go	0.9677	0.7317	0.8333	30	1	11
lights off	0.7903	0.7153	0.7510	98	26	39
Overall	0.9370	0.8836	0.9095	372	25	49

the highest number of false negatives (39) and a significant number of false positives (26), indicating a particular difficulty in correctly identifying this class. This may be due to its visual similarity to other classes or challenging lighting conditions in

the test samples. The **stop** class contributed significantly to false positives (30), the model may be overfitting features associated with **stop** signals.

8.5.2.2 RT-DETR

The RT-DETR model was evaluated after training on the test dataset for 60 epochs with a confidence threshold of 0.5 and an IoU threshold of 0.4. The model achieved overall precision of 0.9589, recall of 0.7767, and an F1 score of 0.8583, indicating robust detection capabilities despite certain class-specific challenges. Class imbalance effects were observed in the performance of the model in different categories.

- **Highest performance:** The **warning** class demonstrated balanced precision and recall values (both 0.9032), resulting in the highest F1 score (0.9032) among all classes. Perfect precision (1.0000) was achieved for the **adjust speed and warning** class, although at the cost of lower recall (0.6667).
- **Problematic classes:** Significant difficulties were encountered with the **lights off** class, where only 48.91% of the instances were correctly identified (recall of 0.4891), resulting in 70 false negatives and the lowest F1 score (0.6291) despite relatively high precision (0.8816).

A precision and recall trade-off was evident in different classes. The **adjust speed and warning** class was never falsely predicted (precision of 1.0000), but 33.33% of its instances were missed. In contrast, the **go** class exhibited the highest recall (0.8947) but lower precision (0.8361), indicating occasional false positives. The analysis revealed characteristic error patterns. The **lights off** class contributed significantly to the overall false-negative count (70 out of 94), suggesting particular challenges in its recognition that may be attributed to visual ambiguity or environmental factors. The **stop** class generated the highest number of false positives (17), which could potentially indicate overgeneralization of its visual features with other signal types.

Table 8.14: Detection performance metrics by class for RT-DETR

Class	Precision	Recall	F1 Score	TP	FP	FN
stop	0.8607	0.8203	0.8400	105	17	23
go	0.8361	0.8947	0.8644	51	10	6
warning	0.9032	0.9032	0.9032	28	3	3
adjust speed and warning	1.0000	0.6667	0.8000	18	0	9
adjust speed and go	0.9394	0.7561	0.8378	31	2	10
lights off	0.8816	0.4891	0.6291	67	9	70
Overall	0.9589	0.7767	0.8583	327	14	94

Conclusion

9

In this diploma thesis, automated visual recognition of railway signals within the Czech railway system was researched. A multistage approach was implemented where traditional computer vision techniques were combined with modern architectures such as CNNs, EfficientNet, YOLO, and RT-DETR, each being evaluated for detection and classification metrics.

The research methodology and implementation also had a semi-automatic annotation process for train cabin videos, through which training data was generated. The experimental results showed that railway signals could be detected and classified with high-performance metrics using the proposed models. Notable performance was observed in the YOLOv5 and YOLOv8 architectures, which demonstrated strong mean Average Precision (mAP50-95 0.72 valid) and F1 score 0.9 (test), which makes them suitable for locomotive driver assistance systems. The integration of custom CNN further enhanced the system. The findings have several implications for the safety of railways in the Czech Republic and possibly also in regions with similar signaling systems. Through the automation of railway signal recognition, human error can be reduced while overall operational reliability can be improved. The developed models can serve as a tool for locomotive drivers to warn them of incoming restrictions. Several limitations of the current approach were identified during the research. Detection accuracy was found to be affected by challenging illumination conditions, as also noted by Staino et al. [Sta+22]. Furthermore, detection performance was poor when signals were placed against complex backgrounds, particularly against trees in forested areas where the color contrast was lower.

In future research directions, the detection of flashing lights and repeating signals with white light (Fig. 2.4) could be included, which goes beyond the scope of this thesis and requires a more complex analysis and additional data collection. The integration of the proposed system into onboard cameras also presents challenges that must be addressed, particularly the need for accelerated inference speeds to adapt to the high velocity of moving trains. By addressing these limitations while balancing technical innovation with practical application, railway safety can be further enhanced through solutions that effectively complement human expertise.

Bibliography

- [Car+20] CARION, Nicolas et al. *End-to-End Object Detection with Transformers*. 2020. Available from arXiv: 2005.12872 [cs.CV].
- [YBH15] YANN; BENGIO, Yoshua; HINTON, Geoffrey. *Deep learning*. cs.toronto.edu, 2015. Available also from: <https://www.cs.toronto.edu/~hinton/absps/NatureDeepReview.pdf>.
- [06] wikimedia.org, 2006. Available also from: https://upload.wikimedia.org/wikipedia/commons/thumb/c/c5/Konvoluce_2rozm_diskretni.jpg/1280px-Konvoluce_2rozm_diskretni.jpg.
- [CVA23] CVAT.AI CORPORATION. *Computer Vision Annotation Tool (CVAT)*. 2023. Version 2.25.0. Available also from: <https://github.com/cvat-ai/cvat>.
- [19] *Europe: rail network density by country size 2019*. 2019. Available also from: <https://www.statista.com/statistics/1243209/europe-rail-network-density-per-country-size/>.
- [Kul12] KULKARNI, Nilima. Color Thresholding Method for Image Segmentation of Natural Images. *International Journal of Image, Graphics and Signal Processing*. 2012, vol. 4. Available from DOI: 10.5815/ijigsp.2012.01.04.
- [Lv+23] LV, Wenyu et al. *DETRs Beat YOLOs on Real-time Object Detection*. 2023. Available from arXiv: 2304.08069 [cs.CV].
- [MSL93] MAŘÍK, Vladimír; STĚPÁNKOVÁ, Olga; LAŽANSKÝ, Jiří. *Umělá inteligence*. Praha, CZ: Academia, 1993. ISBN 80-200-0496-3.
- [Met] METACENTRUM. *National Grid Infrastructure MetaCentrum* [Available at <https://www.metacentrum.cz/>]. [N.d.]. Supported by the CESNET e-infrastructure (LM2023051) funded by the Ministry of Education, Youth and Sports of the Czech Republic.

- [Sch22] SCHNURPFEIL, Daniel. *Architectural Style Classification on Snapshots of Buildings*. ZCU, 2022. Available also from: <https://naos-be.zcu.cz/server/api/core/bitstreams/b699dd9d-fb36-436b-a9d4-1bdb7d6e13c3/content>.
- [Red+16] REDMON, Joseph; DIVVALA, Santosh; GIRSHICK, Ross; FARHADI, Ali. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. Available from arXiv: 1506.02640 [cs.CV].
- [RMR17] RITIKA, S; MITTAL, Shruti; RAO, Dattaraj. *Railway Track Specific Traffic Signal Selection Using Deep Learning*. 2017. Available from arXiv: 1712.06107 [cs.CV].
- [Sta+22] STAINO, Andrea; SUWALKA, Akshat; MITRA, Pabitra; BASU, Biswajit. Real-Time Detection and Recognition of Railway Traffic Signals Using Deep Learning. *Journal of Big Data Analytics in Transportation*. 2022, vol. 4, pp. 57–71. Available from DOI: 10.1007/s42421-022-00054-7.
- [25] *Statistiky mimořádných událostí | Drážní inspekce* [Online]. 2025. Available also from: <https://di.gov.cz/mimoradne-udalosti/statistiky-mimoradnych-udalosti>.
- [Svo24] SVOBODA, Jiří. *Dopravní a návěstní předpis pro tratě nevybavené evropským vlakovým zabezpečovačem* [Online]. 2024. Available also from: <https://provoz.spravazeleznic.cz/Portal/Show.aspx?oid=2179979>.
- [TL20] TAN, Mingxing; LE, Quoc V. *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. 2020. Available from arXiv: 1905.11946 [cs.LG].
- [Ult25a] ULTRALYTICS. *Intersection over Union (IoU)*. 2025. Available also from: <https://www.ultralytics.com/glossary/intersection-over-union-iou>. Accessed: 2025-04-15.
- [Ult25b] ULTRALYTICS. *Mean Average Precision (mAP)*. 2025. Available also from: <https://www.ultralytics.com/glossary/mean-average-precision-map>. Accessed: 2025-04-15.
- [Zau10] ZAUNER, Christoph. Implementation and Benchmarking of Perceptual Image Hash Functions. In: 2010. Available also from: <https://api.semanticscholar.org/CorpusID:17075066>.

List of Figures

2.1	Train accidents caused by illegal driving through railway signals, data are taken from [25]	5
2.2	"Návěst Volno" (go signal) steady green light on the left; "Očekávejte rychlost 100 km/h" (Expect speed of 100 km/h): rapidly flashing green light in the center; "Návěst Očekávejte rychlost 40 km/h" (Expect speed of 40 km/h): slowly flashing yellow light on the right	7
2.3	"Návěst Stůj" (single red light) on the left; Limit 60 km/h and go in the center; Limit 40 km/h and warning in the right	8
2.4	"Rychlost 40 km/h a opakování návěsti Výstraha" Speed 40 km/h and repeating the signal Warning	9
3.1	Biological model of neuron with mathematical formula	12
3.2	Multiple Layer Neural Network	14
3.3	CNN Layers (picture is taken from [YBH15])	14
3.4	Convolution Example (picture is taken from [06])	15
3.5	(a) baseline; (b)-(d) advanced methods; (e) their proposed compound scaling method (Diagram and label are taken from [TL20])	16
3.6	One of the first YOLO architectures (Diagram is taken from [Red+16])	17
3.7	Simple diagram of the Detection Transformer architecture	18
5.1	Approach for Automatic annotation	25
5.2	Single-stage vs Two-stage Model	29
5.3	Approach for Semi-automatic annotation	30
5.4	Aspect ratio analysis for stop and go states	31

List of Tables

3.1	Confusion Matrix Example	20
6.1	Statistics from Strojvedouci COM videos at YouTube channel	33
6.2	Statistics from Parníci CZ videos at YouTube channel	34
6.3	Statistics from Pohled z vlaku videos at YouTube channel	36
6.4	Outliers	36
6.5	Initial performance metrics for pre-trained YOLOv5 on Czech railway signals	37
8.1	Best Grid Search Results: Railway Signal Detection Performance . . .	44
8.2	Multi-Feature Railway Signal Detection Performance	45
8.3	Basic Signal State Classification Performance	46
8.4	Grid search Results	48
8.5	Best Grid search Results	50
8.6	Grid Search Results	52
8.7	Dataset Results Comparison	54
8.8	Distribution of samples across test classes	55
8.9	Overall detection performance metrics	55
8.10	Overall classification performance metrics	56
8.11	Detection backbone test configuration parameters	57
8.12	Classification head configuration parameters	57
8.13	Detection performance metrics by class	58
8.14	Detection performance metrics by class for RT-DETR	59

1101001
101011000011100010 1100001
101011010101 1100001



11010011101101001
01100001 1100001
111000101011 1100001