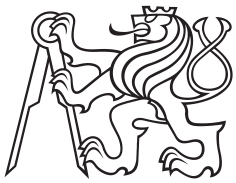
Bachelor Thesis



Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

Explainable Object Detectors for Autonomous Driving

Jan Koča

Supervisor: Ing. Lukáš Neumann, Ph.D.
Study Program: Cybernetics and Robotics
May 2024

BACHELOR‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Koča Jan** Personal ID number: **499023**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Cybernetics and Robotics**

II. Bachelor’s thesis details

Bachelor’s thesis title in English:

Explainable object detectors for autonomous driving

Bachelor’s thesis title in Czech:

Analýza rozhodování detektorů objektů pro autonomní automobily

Guidelines:

1. Familiarise yourself with object detector methods [1, 2] and methods for explaining/visualizing decisions of modern deep neural networks [3, 4]
2. Create an algorithm that exploits the aforementioned methods [3,4] to visualize how image regions that the network uses to make its decision change for a single object in a sequence of frames
3. Test the proposed algorithm on a standard autonomous driving dataset, such as KITTI [5]
4. Analyse, qualitatively and quantitatively, which regions of the image contribute to network’s decision and how do the regions change over time (within a single sequence that captures a single object)
5. Optional: Propose, how the gained insights might improve object detection accuracy

Bibliography / sources:

[1] He, Kaiming, et al. "Mask R-CNN." ICCV 2017.
[2] Wu et al., “Detectron 2”, online: https://github.com/facebookresearch/detectron2
[3] Selvaraju, Ramprasaath R., et al. "Grad-cam: Visual explanations from deep networks via gradient-based localization." ICCV 2017
[4] Ramaswamy, Harish Guruprasad. "Ablation-CAM: Visual explanations for deep convolutional network via gradient-free localization." WACV 2020.
[5] G Andreas et al. "Vision meets robotics: The KITTI dataset." The International Journal of Robotics Research 32.11 (2013): 1231-1237

Name and workplace of bachelor’s thesis supervisor:

Ing. Lukáš Neumann, Ph.D. Visual Recognition Group FEE

Name and workplace of second bachelor’s thesis supervisor or consultant:

Date of bachelor’s thesis assignment: **30.01.2024** Deadline for bachelor thesis submission: **24.05.2024**

Assignment valid until: **21.09.2025**

Ing. Lukáš Neumann, Ph.D.	prof. Dr. Ing. Jan Kybic	prof. Mgr. Petr Páta, Ph.D.
Supervisor’s signature	Head of department’s signature	Dean’s signature

III. Assignment receipt

The student acknowledges that the bachelor’s thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor’s thesis, the author must state the names of consultants and include a list of references.

. | Date of assignment receipt

Student’s signature

Acknowledgements

I would like to extend my sincere appreciation to all who have generously offered their support throughout my academic journey and who have contributed to the process of composing this thesis.

Above all, I would like to express my genuine gratitude to my supervisor, Ing. Lukáš Neumann, Ph.D., for their prompt support, insightful feedback, and boundless patience. Their expert guidance has been crucial from the inception to the final stage of this thesis.

I am also profoundly grateful to my colleagues and lecturers at CTU, whose expertise and assistance have enhanced my understanding of the subject matter. Their constructive criticism and thoughtful discussions have greatly contributed to the improvement of this thesis.

To my friends and family, I owe a debt of gratitude for their unrelenting encouragement, understanding, and patience. Their love, support, and belief in my abilities have provided me with stable foundations for my academic pursuits.

Lastly, I would like to express my heartfelt appreciation to all those who may not be mentioned here but have contributed in various ways to the completion of this thesis.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prague, May 24, 2024

_____

Signature

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 24. května 2024

_____

podpis autora práce

Abstract

The primary objective of this thesis is to explore the potential of employing Explainable AI (XAI) methods to gain insights into the decision-making processes of state-of-the-art object detectors for sequential data, with a focus on applications in autonomous driving.

The research begins by reviewing historical and modern object detection approaches and identifying suitable XAI methods that are able to interpret decisions made by deep neural networks on image data, particularly those that highlight relevant parts of the input image.

Subsequently, an appropriate dataset relevant to autonomous driving, containing sequential data, is identified. A suitable object detection model is selected and integrated with the chosen XAI method to explain the model's decisions on the dataset.

Finally, the thesis involves a quantitative and qualitative analysis of the XAI outputs for the model's decisions, aiming to uncover patterns and dependencies within the explanations.

Keywords: object detection, explainable artificial intelligence, autonomous driving, deep learning

Supervisor: Ing. Lukáš Neumann, Ph.D.

Abstrakt

Hlavním cílem této práce je prozkoumat možnosti využití metod vysvětlitelné umělé inteligence (XAI) k získání vhledu do rozhodovacích procesů nejmodernějších detektorů objektů pro sekvenční data se zaměřením na aplikace v autonomním řízení.

Výzkum začíná přehledem historických a moderních přístupů k detekci objektů a identifikací vhodných metod XAI, které jsou schopny interpretovat rozhodnutí hlubokých neuronových sítí na obrazových datech, zejména těch, které zvýrazňují relevantní části vstupního obrazu.

Následně je identifikován vhodný soubor dat relevantní pro autonomní řízení, který obsahuje sekvenční data. Je vybrán vhodný detektor objektů, který je následně integrován se zvolenou metodou XAI, jež dokáže poodhalit zdroje jednotlivých rozhodnutí modelu při aplikaci na datové sadě.

Nakonec práce zahrnuje kvantitativní a kvalitativní analýzu výstupů XAI pro jednotlivá rozhodnutí detektoru, jejímž cílem je odhalit vzorce a závislosti v těchto výstupech a tím i v rozhodování daného detektoru.

Klíčová slova: detekce objektů, vysvětlitelná umělá inteligence, autonomní řízení, hluboké učení

Překlad názvu: Analýza rozhodování detektorů objektů pro autonomní automobily

Contents

Chapter 1

Introduction

1.1 Motivation

Autonomous driving has already a well-established and rich history and is often regarded as the future of transportation. Nevertheless, it is still not fully accepted by today's society.

Despite its potential to reduce traffic accidents and address issues such as infrastructure overload and congestion, people are still uncomfortable with self-driving cars on the roads, let alone riding in them themselves. Many of these fears originate from a general skepticism towards new technologies for which people lack prior experience. But there are also other factors that are equally significant, such as the unclear decision-making process of the controlling AI agent or insufficient explanations of cases where the AI failed to make the right decision across various conditions and circumstances, thereby appearing unreliable.

These factors can potentially be addressed by a sub-field of artificial intelligence studies called **explainable AI**, also known as **XAI**, which presents methods and algorithms that partially uncover the mentioned uncertainty and provide relatively understandable insights into AI models that might otherwise seem like black boxes.

By leveraging the possibilities offered by XAI in autonomous driving, this field may see at least a modest improvement in much-needed trust. Another significant application of this practice is that researchers can utilize it to explain the circumstances of AI agents' incorrect predictions and, based on that, adjust the engineering process of the AI model. For example, this could involve providing more data of the exposed specific cases during the learning phase.

Neural network-driven **object detectors** play a crucial role in autonomous car technology, as the behavior and reactions of self-driving cars strongly rely on precise identification and localization of objects in their surroundings. While various sources, including LIDAR or GPS, contribute to the perception of the surrounding world, cameras producing RGB images are the primary means of perception, making 2D object detection essential.

Given this premise, it is reasonable to apply relevant XAI methods to object detectors to utilize the advantages they offer for autonomous driving.

1.2 Key Objectives

It is essential to clearly outline the key objectives of this thesis as they serve as the foundational framework that guides the research effort, especially considering the broad scope of the original assignment.

The key objectives, listed sequentially, are as follows:

1. Gather information on existing object detection approaches, focus primarily on modern deep-learning-based detectors. Additionally, research the explainable AI methods for modern image-processing convolutional neural networks, particularly object detectors. Focus especially on methods that are grounded in highlighting the relevant parts of the input image.
2. Explore available datasets relevant to autonomous driving and determine a suitable option for fulfilling the subsequent objectives. An important criterion to consider is the presence of sequential data within the dataset, as this requirement will be crucial for achieving the other goals outlined in this thesis.
3. Given the chosen dataset and the obtained information about object detection algorithms, identify important components and parameters of the deep-learning-based detectors, and based on that choose a suitable model to proceed with.
4. Considering the outputs of previously outlined objectives, select a state-of-the-art XAI method for the subsequent analysis.
5. With the chosen dataset and object detector in mind, select a target object class, such as pedestrians or cars, and identify instances of this class within the dataset. Utilize the sequential nature of the data and track the same instances across the dataset. For example, track each pedestrian over all images. Finally, determine the criteria for satisfactory instances and appropriately filter them.
6. Develop an algorithm that implements the selected XAI method and visualizes explanations of the decisions made by the chosen object detector for a specified object. Apply this algorithm on the filtered instances of the nominated object class within the dataset. (This step is not further reflected in this thesis as it does not delve into specifics of the implementation. However, the results of this step are shown in the subsequent parts.)
7. Define metrics capable of capturing changes in explanations of the decisions made by the object detector for the object instances over time. Additionally, define other supportive metrics if necessary. The primary objective of the metrics should be to uncover potential patterns and dependencies within the explanations.

8. Analyze the results both qualitatively and quantitatively. Describe how the explanation generally changes over time within a single sequence that captures an autonomous-driving-related object.

9. (Optional) Propose, how the gained insight might improve accuracy of object detection.

1.3 Thesis Outline

The thesis follows this structure:

Chapter 2 delves into research on object detection and explainable AI, focusing on the essential information about the detector and XAI method used in the subsequent experiments and analysis.

Chapter 3 details the extraction of the data needed for the analysis. It describes the selected dataset, the deep learning detector used in this work, the selection process of the XAI method, its adjustments, parameter settings, and proposes suitable metrics for the outputs of the conducted experiments.

Chapter 4 involves the analysis and interpretation of the outputs from the previous chapter.

Chapter 2

Theoretical Background

This chapter provides a comprehensive overview of key concepts in object detection and explainable AI this thesis builds upon. The chapter directly addresses the first objective outlined in section 1.2.

In section 2.1, the general principles of object detection are outlined, covering both historical and modern approaches. The discussion primarily centers on modern deep learning detectors, with a focus on two-stage detectors and a brief mention of one-stage detectors. Section 2.2 delves into explainable AI, detailing the common technique of pixel attribution maps to visualize explanations of computer vision neural networks.

2.1 Object Detection

Object detection, a fundamental task in computer vision alongside image classification, segmentation, and others, involves locating and identifying objects within digital images.

The technology of object detection is currently utilized across a wide spectrum of fields. It finds applications in sectors such as healthcare, security, marketing, agriculture, autonomous driving, and many others. In these fields, object detection systems are deployed to identify various entities, including people, animals, traffic signs, medical features, and more.

Although 3D object detection also exists, it is not relevant for this thesis.

2.1.1 Essence of Object Detection

Object detection can basically be seen as an extension of object classification, incorporating the additional task of object localization.

The fundamental premise underlying both detection and classification is that each type of object possesses specific distinguishing features. For instance, a basketball is typically characterized by its round shape and orange color, while a polar bear is usually identified by its white fur and distinctive body parts such as a head and legs.

Geometric features within images are recognized by the object detectors at various levels. At a low level, detectors may identify basic geometric shapes such as edges and corners. Meanwhile, at a higher level, detectors

may recognize complex structures such as body parts. Color information is considered by analyzing geometric features within each color channel of an RGB image and subsequently integrating all partial outputs.

Various ways of extracting these features exist, and each detection method implements them differently. Once the features are obtained, they are subjected to an object classification process.

Localization of an object usually comprises determining a bounding box, a rectangle with sides parallel to the image axes, that tightly encloses the object in the image. Typically, this bounding box is represented either by the coordinates of its top-left and bottom-right corners in the image or by providing the center (or one corner) coordinates and dimensions of the bounding box. Again, the way this bounding box is acquired varies across different object detection methods.

The typical object detection pipeline involves feeding an image into a detector designed to recognize specific object types, known as object classes (e.g., cars, people). The detector then scans the image to locate instances of these classes (e.g., specific cars), producing output consisting of bounding box coordinates, class labels, and confidence scores. These scores indicate the detector's level of certainty for each detected object. Since the detector often produces numerous suggestions, a common practice is to filter out less certain detections using a confidence score threshold.

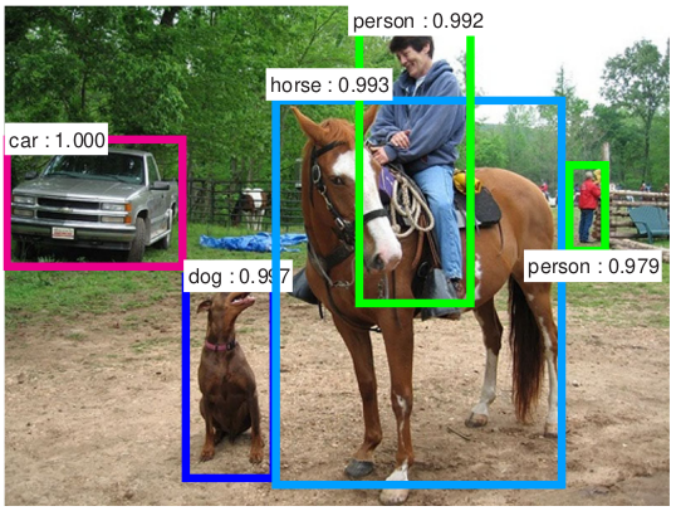


Figure 2.1: Example output of object detection process. [1]

Two approaches to object detection are widely recognized: **traditional methods** and **deep-learning-based methods**.

2.1.2 Traditional Object Detection

Historically, methods falling under the traditional, non-deep-learning approach were the first relatively successful object detectors.

For object localization, these methods commonly employed techniques like sliding windows or concepts similar to it. In the sliding window method, the algorithm places a window of multiple scales over various locations in the image. Each of these image regions is then searched for specific features, using methods such as analyzing pixel intensity gradients or applying predefined sets of kernels. The extracted features are subsequently processed through a classification step, often utilizing algorithms like *Support Vector Machine* [2] (SVM).

Examples of these methods include the *Viola-Jones* [3] algorithm, introduced in 2001 by Paul Viola and Michael Jones, and the *Histogram of Oriented Gradients* [4] (HOG) algorithm, originally proposed in 2005 by N. Dalal and B. Triggs.

While these methods achieved significant milestones, including face and person recognition, and underwent considerable refinement over the years, they exhibited notable drawbacks and limitations. These limitations include binary classification (recognizing only one object class) or the inability to detect slightly more complex scenarios, such as side views of faces for the case of the face detectors. Consequently, they were eventually outperformed by the deep learning approach.

2.1.3 Object Detection in Deep Learning

The vast majority of today's object detection applications leverage deep-learning-based algorithms, harnessing the significant capabilities of deep neural networks.

These methods effectively address the limitations of traditional approaches. For instance, they demonstrate robustness in complex scenarios, while also possessing the capability to distinguish multiple object classes. However, they also bring forth new challenges and drawbacks, such as high computational requirements for training and the need for a large amount of annotated data. This annotation process typically requires extensive labor and financial resources.

Two primary types of object detectors utilizing neural networks are commonly recognized: **two-stage** (multi-shot) and **one-stage** (single-shot).

This classification is marked by two important milestones in the history of object detection. The first is the invention of the **region-based convolutional neural network (R-CNN)** by Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik (originally) in 2013 [5], which served as the foundation for subsequent two-stage detectors. The second milestone is the release of the work *You Only Look Once: Unified, Real-Time Object Detection* by Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi in 2016 [6], which introduced one of the first widely recognized one-stage detectors called **YOLO**.

2.1.4 Two-Stage Detectors

Two-stage detectors, as their designation suggests, operate in two distinct stages: the region proposal stage for localization and the subsequent classification stage. Each of the stages is handled by its own distinct algorithm.

There are many profound algorithms in this category but most of them are based on the R-CNN. Also, it is worth noting that the term 'region' is frequently used throughout this section, typically referring to rectangular areas in images.

2.1.4.1 R-CNN:

The R-CNN (*Region-Based Convolutional Neural Network*) [5] pioneered the development of two-stage algorithms by introducing the foundational concept that led to the creation of an entire family of related methods.

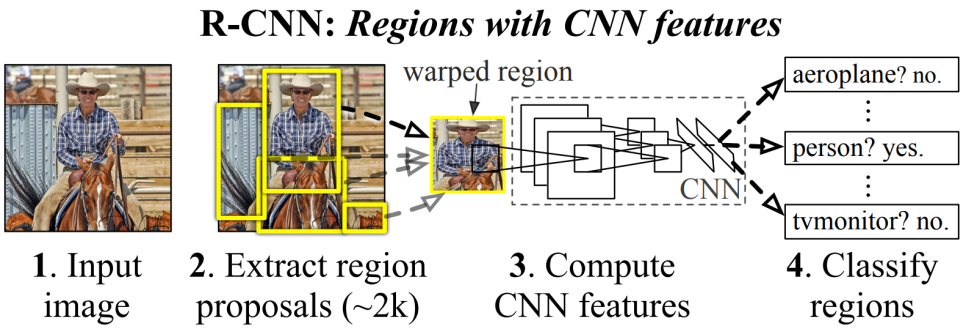


Figure 2.2: Pipeline of the R-CNN algorithm. [5]

The first step in object detection, localization, often presents a bottleneck for the efficiency of object detection algorithms, particularly when imprecise, leading to the generation of numerous regions for object search. One of the simplest yet somewhat naive approaches is the sliding window algorithm, also known as exhaustive search. However, the R-CNN architecture improves the efficiency by adopting a technique known as **Selective Search** [7].

The selective search typically involves three main steps:

1. Segment the image based on pixel intensity using a graph-based segmentation method.
2. Combine smaller, similar (amorphous) regions into larger ones recursively, employing the following algorithm:
   a. From the set of regions, select the two that are most similar.
   b. Merge them into a single, larger region.
   c. Repeat the above steps. Through this iterative process, larger segments are formed and appended to the list of identified regions. The process concludes upon reaching predefined thresholds for maximum region size, number of regions, and similarity.

Various methods exist to assess region similarity, including color, texture, size, or fill similarity, as proposed in the original paper.

3. Utilize the identified regions to generate region proposals by assigning them bounding boxes.

Figure 2.3: Process of extracting region proposals using the Selective Search algorithm. The first image in the top row illustrates the initial segmentation step of the algorithm, while the other two images depict the subsequent merging of regions. The bottom row images illustrate the final step, where each region is assigned a bounding box. Each image depicts the result corresponding to the segmentation state shown in the respective image in the top row. It is important to note that the color specification of the bounding boxes is irrelevant for this explanation. [7]

As a result, approximately 2000 region proposals (also called object proposals) are generated using this approach.

The second stage involves warping the proposed regions into a square shape. These regions are then inputted into a feature-extracting convolutional neural network (CNN). The feature maps obtained from the CNN's output are subsequently evaluated by linear *Support Vector Machines* [2] (SVMs), each trained to recognize a specific class. The resulting score indicates the probability of the class being present within the region.

Each proposed region is evaluated for every class. Using these scores, a greedy *Non-Maximum Suppression* technique is employed independently for each class. This technique rejects a region if it has an *Intersection-over-Union* (IoU) overlap with a higher-scoring unsuppressed region that exceeds a predetermined threshold.

In subsequent versions, a simple linear regression model was integrated into the algorithm to mitigate localization errors. This model returns four offset values used to adjust the bounding box coordinates.

The algorithm encounters several challenges. Its deployment for real-time applications is hindered by the considerable time required for detection, often taking tens of seconds per image. Additionally, the training time is prolonged for the same reason. Moreover, the fixed nature of the selective search component restricts this part of the algorithm from learning from the provided data.

These challenges have been addressed by several subsequent methods, including *SPPnet* [8], *Fast R-CNN* [9], *Faster R-CNN* [1], *Mask R-CNN* [10], *Feature Pyramid Network* [11] (FPN), and others. Some of these methods will be explored further in this section, selected based on their relevance to this thesis.

2.1.4.2 Fast R-CNN:

The Fast R-CNN [9] method succeeds the R-CNN directly, preserving its fundamental concept while introducing significant alterations in the second stage of the algorithm.

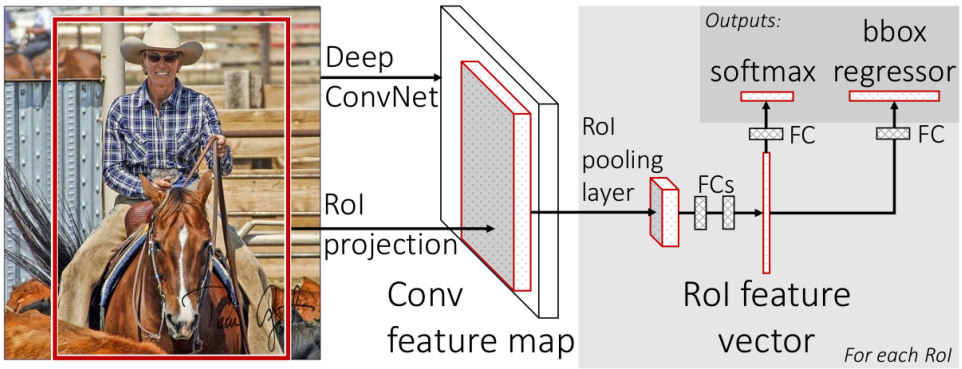


Figure 2.4: Pipeline of the Fast R-CNN algorithm. [9]

The algorithm processes the analyzed image along with a set of object proposals, obtained for example by the Selective Search. Initially, the image undergoes several convolutional and max-pooling layers to generate a feature map. Subsequently, for each object proposal, a *Region of Interest (RoI) pooling layer* [9] extracts a fixed-length feature vector from the feature map. These feature vectors are then fed into a sequence of fully connected layers, which branch into two output layers.

The first output layer produces probability estimates across defined number of object classes, including a catch-all "background" class. Meanwhile, the second output layer generates four numbers encoding refined bounding-box positions for each class.

The RoI pooling layer converts the features within any valid region of interest into a compact feature map with a fixed spatial extent, where the dimensions are determined by layer hyperparameters that are set when the neural network is designed.

This approach significantly reduces the detection time to mere seconds, and training time is also substantially decreased. The primary reason for this improvement is the elimination of the need to employ the CNN to extract the feature map from each object proposal individually. Instead, this process is performed only once for the entire image, and the features for each object proposal are subsequently extracted from the resulting feature map.

When the generation of region proposals is excluded from the detection process (i.e., they are pre-generated), the detection time decreases to under a second. This suggests that the next bottleneck to address is the region proposal generation.

2.1.4.3 Faster R-CNN:

Faster R-CNN [1] is an algorithm built upon the foundation of Fast R-CNN, specifically designed to address its aforementioned bottleneck - region proposal generation.

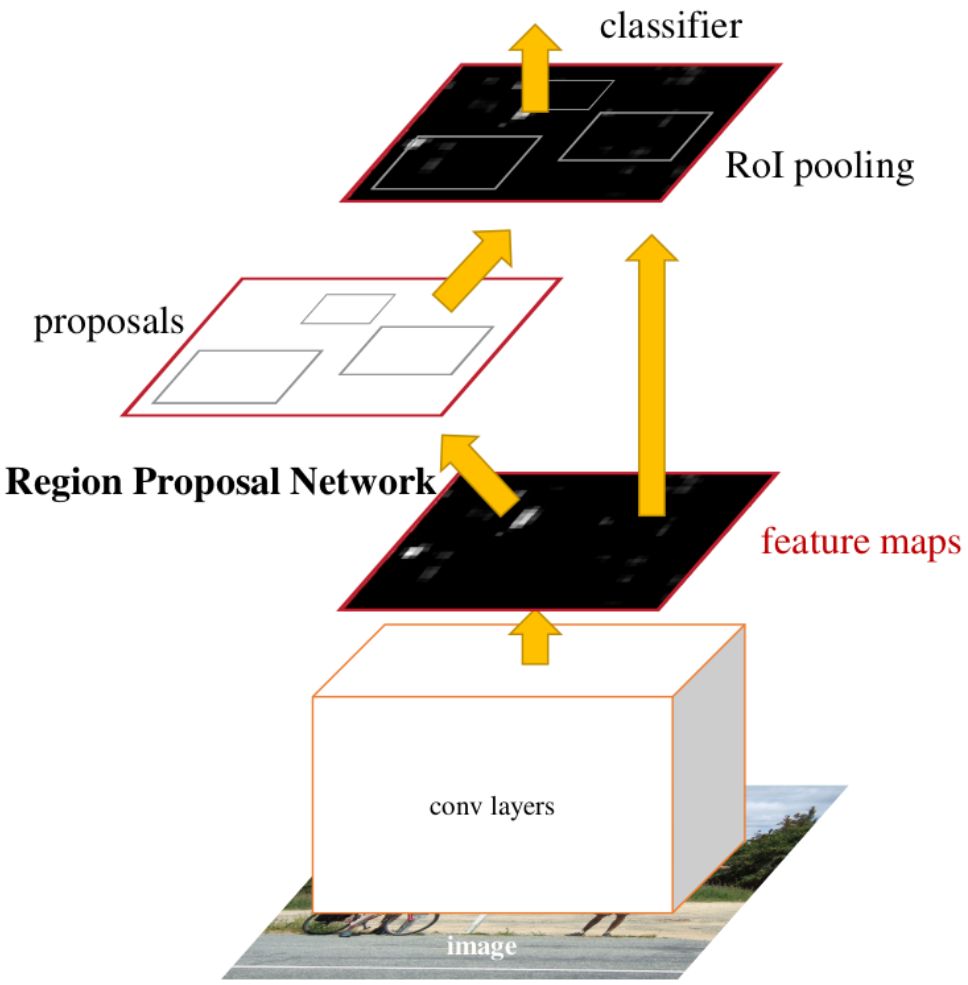


Figure 2.5: Pipeline of the Faster R-CNN algorithm. [1]

The pipeline of this algorithm remains consistent with its previous version, with the addition of a *Region Proposal Network* [1] (RPN) that supplies the region proposals. Notably, the RPN shares convolutional layers with the feature-extracting network, enhancing overall efficiency.

The RPN initially identifies points (usually all) on the multi-dimensional feature map generated by the feature-extracting network. At each of these points, k region proposals are predicted. These proposals are parameterized relative to k reference boxes, known as anchors. An anchor is centered at the defined point and possesses a scale and aspect ratio.

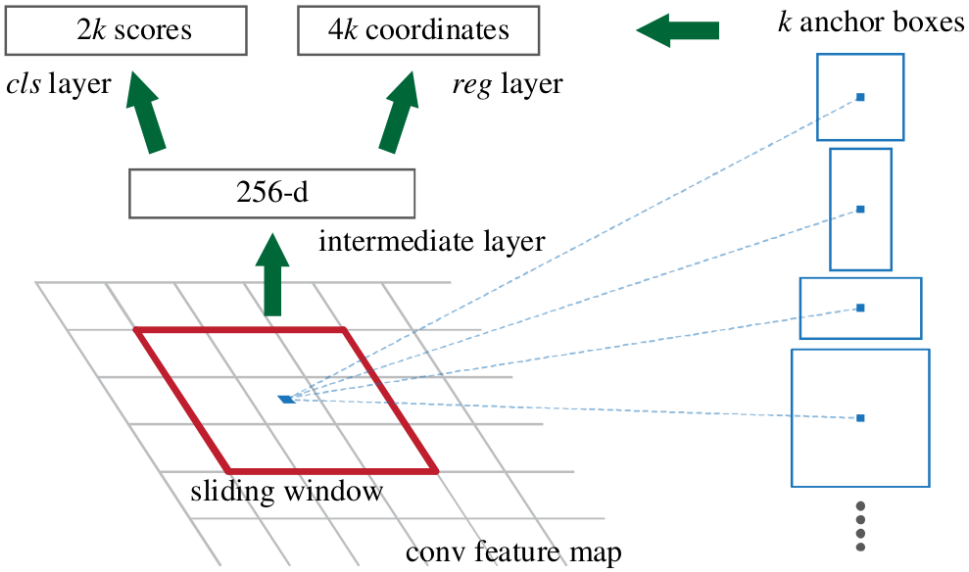


Figure 2.6: Anchor extraction and processing at one of the selected points of the feature map. The '*cls* layer' represents the classification part while the '*reg* layer' represents the regression part. Notably, the parameters depicted are implementation-specific; in this case: '256-d' refers to the feature map's dimensionality, '2k scores' represents the classification output with two classes, and '4k coordinates' denotes the regression output. [1]

Comprising of two components, the RPN includes a classification part and a regression part. The classification part evaluates each anchor and yields a probability that the anchor contains an object. Meanwhile, the regression part outputs four numbers encoding the coordinates of the corresponding region proposal. For a feature map of size $H \times W$, there are a total of $H \cdot W \cdot k$ anchors. Subsequently, the region proposals can be filtered based on the probability score or attributes of the proposal box itself, such as whether it extends beyond the boundaries of the image. After this point, the procedure closely follows the approach of Fast R-CNN.

These improvements have resulted in a considerable decrease in detection time, now measurable in hundreds of milliseconds. Consequently, the Faster R-CNN algorithm is widely employed in numerous real-time applications.

2.1.5 One-Stage Detectors

One-stage detectors, in contrast to two-stage detectors, tackle both object detection tasks, localization and classification, in a single stage by one neural network.

Single-shot detectors are typically faster but may trade off some accuracy compared to multi-shot detectors. Additionally, they often offer easier training due to their design, allowing for end-to-end training, which is not always feasible for detectors in the other group.

Numerous notable one-stage algorithms exist: *You Only Look Once* [6] (YOLO), *Single Shot MultiBox Detector* [12] (SSD), *RetinaNet* [13], and various iterations of YOLO (*YOLOv3* [14] to *YOLOv9* [15]) to name a few.

While this approach is an important part of the modern object detection, this section will solely elaborate on the original YOLO algorithm, considering the broad and complex nature of the topic and the context of this work.

2.1.5.1 YOLO:

The YOLO [6] detector serves as the equivalent in one-stage detectors to what R-CNN represents for two-stage detectors—a pioneering force in its paradigm.

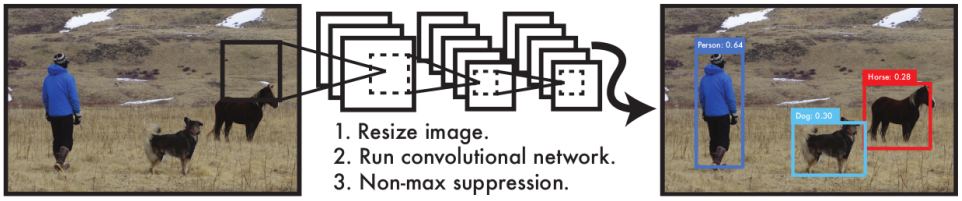


Figure 2.7: Outline of the YOLO algorithm pipeline. [6]

This approach consolidates various components of object detection into a single neural network. It takes an image as input and simultaneously predicts bounding boxes across all classes. This enables the neural network to utilize global context, even the other predictions, for every single object detection.

The initial step of the algorithm involves partitioning the input image into a grid of $S \times S$ cells. Within each grid cell, the neural network predicts B bounding boxes and C conditional class probabilities. Notably, only one set of class probabilities is predicted per grid cell, regardless of the number of boxes. Each bounding box prediction includes five parameters: x, y, w, h, and confidence. The (x, y) coordinates denote the box's center relative to the grid cell bounds, while the width and height are relative to the entire image. The confidence scores indicate both the model's confidence in object presence and the accuracy of its prediction for the box.

Multiplying the conditional class probabilities from the grid cell containing the individual box's center by the individual box confidence predictions yields class-specific confidence scores. These scores encapsulate both the probability

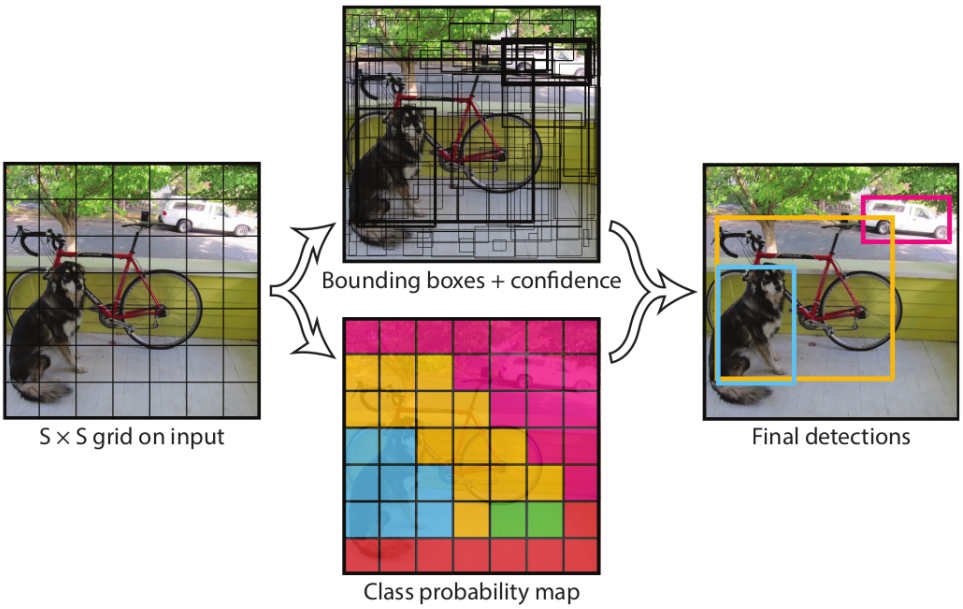


Figure 2.8: Visualized steps of the YOLO algorithm. The left image shows the initial partitioning of the input image into $S \times S$ squares ($S = 7$). The center-top image visualizes the bounding boxes predicted for each of the cells. The widths of the box lines represent the confidence scores for each box. In the center-bottom image, each cell's color represents the maximum probability class from the C conditional class probabilities. Finally, the right image shows the resulting detections, where the bounding box colors carry the same meaning as in the center-bottom image. [6]

of the class appearing in the box and the accuracy of the predicted box's fit to the object.

Among all the bounding boxes, only the relevant ones are chosen by applying a threshold on the class confidence score and using Non-Maximum Suppression. This technique discards a region if its Intersection-over-Union (IoU) overlap with a higher-scoring unsuppressed region exceeds a predefined threshold.

YOLO offers a highly efficient implementation of the object detection algorithm, often employed in real-time applications with speeds of up to 40 frames per second. However, the algorithm also encounters challenges in accurately predicting bounding boxes due to spatial constraints and reliance on coarse features, particularly for small objects or those in intricate configurations. These limitations contribute to localization inaccuracies, serving as the primary source of error.

Subsequent versions of YOLO have addressed many of the initial drawbacks and have significantly enhanced the method in terms of both accuracy and speed.

2.2 Explainable AI

Explainable artificial intelligence (XAI) commonly refers to AI systems that enable humans to maintain cognitive oversight, or the techniques utilized to facilitate this oversight.

While XAI is broad in scope, this thesis solely targets one of its aspects: **pixel attribution methods** [16]. Therefore, this section will delve deeper into this specific topic, although it is important to note that other approaches exist.

2.2.1 Pixel Attribution Methods

The pixel attribution methods aim to uncover the decision-making process of deep learning models, particularly neural networks that operate on image data, and identify their potential biases. They achieve that by identifying and highlighting the pixels within the input image that influence a specific decision made by a neural network.

The output of these methods is a pixel attribution map, typically a grayscale image of the same size as the input image. In this map, each pixel is assigned a numerical value representing the importance of the corresponding pixel in the input image for the specific decision made by a neural network, as illustrated for example in Figure 2.9.

These maps, along with the methods themselves, are referred to by various names such as attention maps, saliency maps, sensitivity maps, attribution maps, feature attribution maps, and others.

Various types of pixel attribution methods exist. Two basic classifications of these methods are relevant for this thesis:

The first classification is based on the way the method interacts with the model.

- **White box methods:** These algorithms require direct access to the analyzed model. They utilize its internal information, such as gradients and activations in specific layers of the neural network, for their explanations. While they tend to be fast, they can be challenging to design due to their model-specific nature. They are also referred to as model-specific methods.

- **Black box methods:** These algorithms treat the model as a black box, meaning they do not require or possess any information about the analyzed model. Instead, they manipulate the input and analyze the resulting output. While they are typically slower than white box methods due to their need of iterating over many input variations, they are easier to use and design as they are not dependent on the specific model. They are also known as model-agnostic methods.

The second classification highlights two primary groups of methods based on how they derive the values for the pixels in attribution maps:

- **Gradient-based methods:** These techniques generate pixel attribution maps by analyzing the gradient (derivative) of the output with respect to specific elements, such as the input image or intermediate feature maps of the neural network. Essentially, the gradient indicates which parts of the specified elements are crucial for the output, as it demonstrates the effect of a slight change in these elements on the output. Since the gradient has the same size as the elements, which generally differs from the input image size, it must be appropriately projected onto the input image to obtain the pixel attribution map. Typically, these methods belong to the white box group because they involve manipulation of the internal parts of the model, such as incorporating hooks to capture the gradients.
- **Occlusion-based methods:** These methods rely on the premise that changing a part of the input image essential for the prediction will significantly alter the output, while altering an unimportant part will have minimal effect. By systematically altering specific parts of the input image, these methods highlight the pixels that are essential for the model's decision. They are often categorized as black box methods and are also referred to as perturbation- or ablation-based methods.

The majority of these methods can be applied to analyze neural networks designed for both image classification and object detection, as they share the same fundamental principle. However, some methods encounter difficulty in providing separate explanations for individual object classes (e.g., dogs and cats) or instances of objects (e.g., two different dogs) if there are multiple objects in the image. This limitation arises from techniques that, for example, utilize activations from the forward pass, making them not suitable for such general cases.

Numerous pixel attribution methods exist, with new ones emerging each year. Gradient-based methods vary in their approach to utilizing gradients, some employing second-grade gradients, others weighting gradients by corresponding activations, and some normalizing the activations beforehand. Similarly, occlusion-based methods differ in their techniques for altering specific parts of the input image, with some using blurring, others binary masks, and still others grayscale masks.

In this work, four specific methods will be further described.

2.2.1.1 Vanilla Gradient

The *Vanilla Gradient* [16] method, originally known as *Image-Specific Class Saliency Visualization* [17], is a straightforward gradient-based approach and one of the earliest pixel attribution methods.

This method basically involves three steps:

1. Input the image of interest into the analyzed model and obtain the results through a forward pass.

2. Calculate the gradient of the desired output (e.g., specific class probabilities) with respect to the input image to obtain the desired pixel attribution map.

3. Visualize the pixel attribution map.

However, this approach faces two main challenges, both of which stem from a common component of neural networks called *ReLU* (Rectified Linear Unit). One of them is a well-known problem called dying ReLU, where neurons in the network may become inactive, leading to zero gradients. The second challenge lies in how the backward pass of the gradients through the ReLU is implemented, which directly affects the resulting attribution maps.

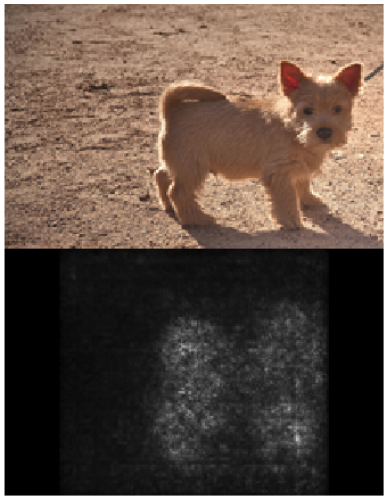


Figure 2.9: Example of using the vanilla gradient algorithm to explain the classification of an image by a neural network. The top image shows the original input, while the bottom image displays the resulting pixel attribution map, where the importance of each pixel is indicated by its intensity (whiteness). [17]

Also, subsequent improvements were introduced to this method in later iterations. For example, it was observed that in the results, a few pixels with much higher than average gradients often appeared, which could throw off the color scales used for visualization. To address this issue, it proved convenient to eliminate these outliers, for instance, by capping the gradients at the 99th percentile.

It would be reasonable to consider this method a lightweight white box method. While it does not directly utilize internal model information, it requires a direct call of the backward pass and propagation of the gradient through the neural network, which may rely on the specific model architecture.

2.2.1.2 SmoothGrad

SmoothGrad [18] serves more as a suggested improvement to existing gradient-based methods rather than a standalone technique.

Its premise revolves around reducing noise by introducing random perturbations to the input image. The rationale behind this approach lies in the observation that gradients often exhibit high variability at small scales. By adding noise to the image and repeating this process multiple times, the resulting gradients can be averaged to mitigate fluctuations.

The method comprises four primary steps:

1. Generate multiple versions of the input image with added noise.
2. Generate a pixel attribution map for each of these noisy images.
3. Average the attribution maps.
4. Visualize the averaged attribution map.

Mathematically, the calculation of the averaged attribution map can be expressed as:

$$\hat{M}(x) = \frac{1}{n}\sum_{1}^{n} M(x + \mathcal{N}(0, \sigma^2)), \tag{2.1}$$

where x represents the input image, $\hat{M}$ is the SmoothGrad attribution map, M is the attribution map generated with the original method, n denotes the number of perturbed versions of the input image, and $\mathcal{N}(0, \sigma^2)$ represents Gaussian noise with standard deviation σ.

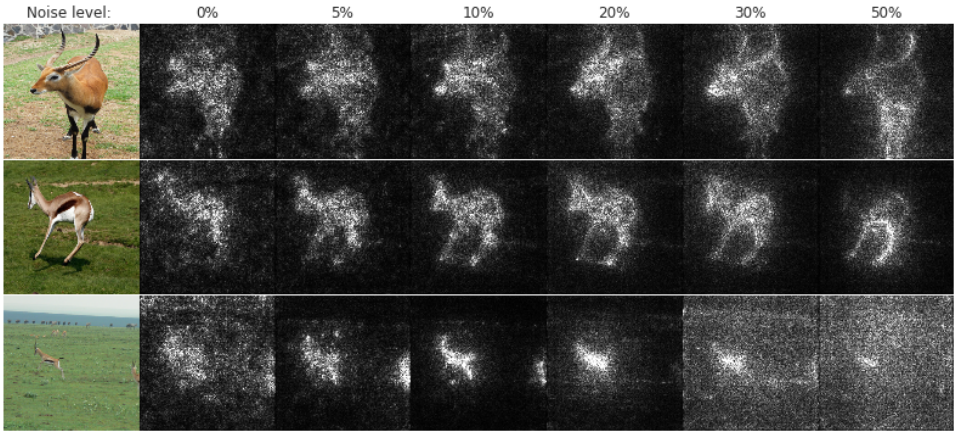


Figure 2.10: Effect of the parameter σ on the resulting pixel attribution map for three images ($n = 50$). The noise levels correspond to $\sigma/(x_{max} - x_{min})$, where x_{max}, x_{min} represent the maximum and minimum values of the image, respectively. [18]

Setting the parameters n and σ requires careful consideration and will vary depending on the specific use case. For instance, selecting an appropriate

value for σ is crucial, as different neural network models exhibit varying sensitivities to noise. Setting it too high may result in the model failing to recognize the target object in the image.

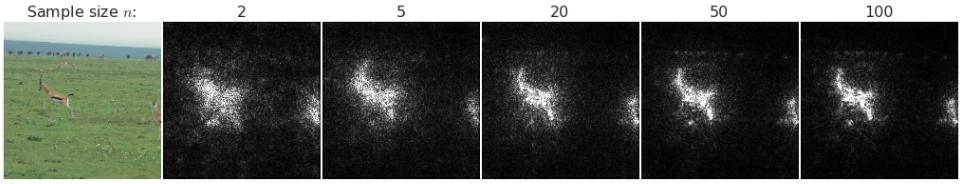


Figure 2.11: Effect of the parameter n on the resulting pixel attribution map. The noise level is set to 10%. [18]

The SmoothGrad method inherently operates without relying on any internal information from the analyzed neural network, it solely interacts with the input data. As it does not constitute a fully-fledged method, it cannot be classified regarding its interaction with the model. Instead, it can be stated that it maintains the original classification in this regard of the method to which it is attached.

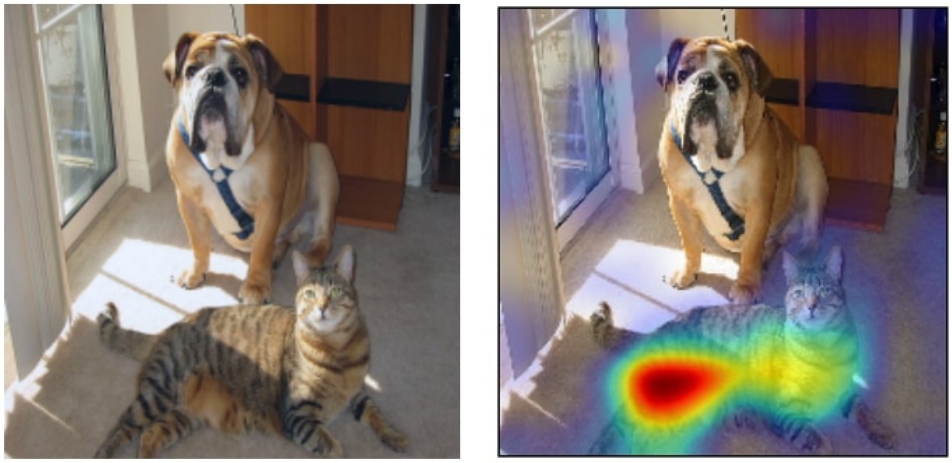
2.2.1.3 Grad-CAM

The Grad-CAM (*Gradient-weighted Class Activation Mapping*) [19] method, a gradient-based approach, is commonly used to explain decisions made by convolutional neural networks (CNNs).

The process of generating the pixel attribution map using this method can be divided into seven steps:

1. Input the image of interest into the analyzed model and obtain its predictions through a forward pass.
2. Calculate the gradient of the desired specific output (e.g., specific class probabilities) with respect to the outputs of the last convolutional layer, also known as feature (activation) map, of the analyzed CNN.
3. Weight the multi-channel feature map by the corresponding gradients using element-wise multiplication.
4. Average the weighted feature map along the channel dimension to obtain 2D feature map.
5. Set all values less than zero to zero.
6. Resize the result to the size of the input image to obtain the attribution map.
7. Visualize the pixel attribution map.

The fundamental premise of this method lies in the assumption that the feature maps contain information about all features present in the input image. These encoded features range from basic (e.g., edges, corners) in the lower levels of the CNN to more abstract in the higher levels. However, a challenge arises as these feature maps contain features of all recognized objects in the image, not solely the ones of interest. Hence, gradient weighting is employed to highlight the importance of specific regions in the feature maps, as indicated by the gradients, for the selected output.



(a) : Original image **(b) :** Grad-CAM's explanation

Figure 2.12: Example demonstrating the Grad-CAM algorithm explaining the classification of an image by a neural network. Here, the model's prediction that there is a cat in the image is explained. The most important parts for the classification are colored red. [19]

It is also important to note that the gradient can be calculated with respect to any of the intermediate feature maps of the CNN. This method will then highlight the different, aforementioned types of features.

Indeed, the Grad-CAM method is a typical exemplar of the white box approach, as it relies on internal information from the neural network, specifically the activations and gradients, to generate its results.

2.2.1.4 D-RISE

The D-RISE (*Detector Randomized Input Sampling for Explanation*) [20] method is an occlusion-based technique specifically designed for explaining deep learning object detection models. It builds upon the RISE method, which served a similar purpose for image classifiers.

Figure 2.13: Example illustrating multiple D-RISE explanations of object detections made by a neural network. The significance of each pixel for the detection is represented by a color gradient, ranging from blue to red, where red indicates higher importance. [20]

The process of generating the pixel attribution map using D-RISE involves six steps:

1. Generate a specified number of RISE masks.
2. Apply each generated mask to the image of interest.
3. For each mask, input the masked image into the analyzed object detector and obtain its detections through a forward pass.
4. Determine a score for each mask by comparing the detections of the target object in both the masked and unmasked images.
5. Weight all masks by their scores to obtain the pixel attribution map.
6. Visualize the pixel attribution map.

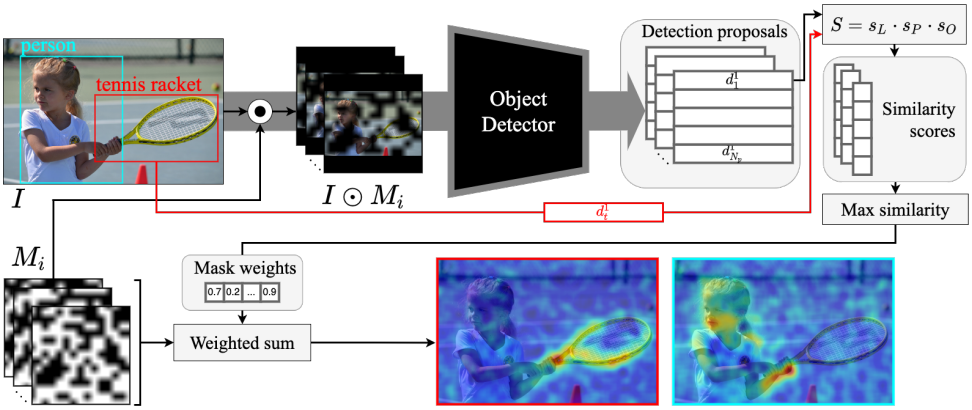


Figure 2.14: Pipeline of the D-RISE algorithm. [20]

As these steps provide only a rough outline and this method is referred to in the later chapters of this work, it is essential to delve into the details of each step.

Generating RISE mask:
The first step of the algorithm involves generating RISE masks for the input image. The resulting masks are grayscale images. This task comprises three sub-tasks:

1. Sample N binary grids of size $h \times w$ (smaller than the image size $H \times W$) by independently setting each element to 1 with probability p and to 0 with probability $(1 - p)$.
2. Upsample all grids to size $(h + 1)C_H \times (w + 1)C_W$ using bilinear interpolation, where $C_H \times C_W = [C_H/h] \times [C_W/w]$ represents the size of the cell in the upsampled grid.
3. Crop areas $H \times W$ with uniformly random offsets ranging from $(0, 0)$ up to (C_H, C_W) to obtain the masks.

Also, three important parameters need to be set in this step:

- Parameter N: The number of masks used in this algorithm, influencing the convergence of the method.
- Parameter $(h \times w)$: The grid size, which represents the resolution of the method.
- Parameter p: The probability of activating (setting to 1) a cell in the binary grid, indicating the percentage of the image being unmasked.

Masking the input image:
The second step is straightforward. It involves taking the masks generated in the first step and pixel-wise multiplying them with the input image. This operation is possible because the masks are of the same size as the input image.

Each pixel of the image contains three values - one for each color channel. These three values will be multiplied by the value of the corresponding pixel in the mask. As the values in the mask approach zero (ranging from one to zero), the corresponding pixels in the image are increasingly dimmed.

Processing the masked image
In the third step, the masked image is fed into the analyzed detector for each generated mask.

The output of the detector may be in a custom format, as different models may implement it differently. In most cases, however, it can be converted to the format of detection vector.

Assuming that the detector has identified K objects in the image, it is useful to describe each of these objects using a detection vector $d_i = [L_i, P_i, O_i]$, where i ranges from 1 to K.

- $L_i = (x_1^i, y_1^i, x_2^i, y_2^i)$ represents the bounding box of the i-th object, defined by its top-left (x_1, y_1) and bottom-right (x_2, y_2) corners.
- $P_i = (p_1^i, ..., p_C^i)$ is a vector of probabilities indicating the presence of each of the C classes within the region L_i.
- O_i represents the objectness score, reflecting the probability of the region L_i containing an object of any class.

For masked image m, a set of detection vectors is acquired, denoted as $D_m = [d_1^m, ..., d_{K_m}^m]$, where K_m represents the number of detected objects in the masked image and m ranges from 1 to N - number of generated masks.

Calculating mask scores:

The fourth step of the algorithm aims to score the masks. This is achieved by evaluating the detections (obtained in the third step) of the target object in the masked images.

It is essential to first select the target object. Typically, it is chosen from the objects detected in the input image. This approach is utilized when the method aims to expose the parts of the image important for the object's detection. However, it can also be provided externally, such as by a human, as a ground truth. This allows for explanations, for example, of why the detector fails to detect the object.

The target object is also defined by a detection vector, denoted as $d_T = [L_T, P_T, O_T]$, where each component carries the same meaning as previously described.

For a mask m, there is a detection vector set D_m (describing all objects detected in the image masked by the mask m) from the previous step. Each detection vector d_i^m from this set can be compared with the target detection vector d_T by assigning a similarity score to it.

The similarity score $s(d_i^m, d_T)$ is calculated as the product of three scalar factors:

$$s(d_i^m, d_T) = s_L(L_i^m, L_T) \cdot s_P(P_i^m, P_T) \cdot s_O(O_i^m, O_T), \tag{2.2}$$

where

$$s_L(L_i^m, L_T) = \text{IoU}(L_i^m, L_T), \tag{2.3}$$

$$s_P(P_i^m, P_T) = \frac{P_i^m \cdot P_T}{||P_i^m||\ ||P_T||}, \tag{2.4}$$

$$s_O(O_i^m, O_T) = O_i^m. \tag{2.5}$$

The s_L, bounding box similarity score, is calculated as Intersection over Union to measure the spatial proximity between two bounding boxes.

The s_P, class probability vector similarity score, is determined by cosine similarity.

And the s_O, objectness similarity score, is computed simply by making it equal to O_i.

Multiplying these three partial similarity scores ensures that if one of them is low, the resulting similarity score is also low. Additionally, it is possible to exclude any of the partial similarity scores if they are not important for the specific use case. Specifically, the objectness similarity score is usually omitted because the analyzed detector does not produce it. It is also possible to change the way some of the similarities score are calculated to fit the method for a specific use case.

For the mask m the similarity score $s(d_i^m, d_T)$ is calculated between each of its detection vectors d_i^m and the target detection vector d_T. The desired mask score s_m is then the maximum of these similarity scores:

$$s_m = \max_i s(d_i^m, d_T), \ 1 \leq i \leq K_m. \tag{2.6}$$

Deriving the pixel attribution mask:

In the fifth step, each of the masks is weighted (multiplied) by the corresponding mask score.

Optionally, the weighted masks can be averaged by dividing them by their total number (N) and subsequently normalized.

The primary drawback of the D-RISE method is its slow speed, as generating explanations typically involves a high number of masks. Each mask requires a forward pass through the model, resulting in considerable computational overhead. Typically, generating a reasonable explanation with bearable level of noise requires processing lower thousands of masks.

2.2.2 Pixel Attribution Map Visualization

There are various methods for visualizing pixel attribution maps, but two approaches are particularly common. Both techniques leverage the fact that the map has the same dimensions as the input image.

The first method involves normalizing the map so that its values fall within the range [0, 1]. Once normalized, these values can be used as pixel intensities by performing pixel-wise multiplication with the input image. Consequently, pixels with higher attribution values remain largely unchanged in the original image, while those with lower values are dimmed.

The second approach utilizes the attribution map as a heat map, a visualization method commonly used for grayscale images where pixel values are represented by different colors. Various color palettes can be employed, such as the cool-to-warm theme, where cooler colors (e.g., blue) represent lower values and warmer colors (e.g., red) represent higher values, transitioning gradually from red to blue. Typically, this method involves overlaying the grayscale image (the attention map) onto the input image with partial transparency.

Chapter 3

Data and Experiments

3.1 Dataset

The dataset plays a pivotal role in ensuring clear validation of the algorithm's outcomes and the consequent analysis in this work. This section directly addresses the second objective outlined in section 1.2.

3.1.1 Dataset Selection

There are two main requirements for the eventually selected dataset to meet:

1. The dataset must contain RGB images related to autonomous driving, for example, images obtained from a car driving around a city. The need for RGB images arises from the fact that object detectors, which are the ultimate focus of this work, take them as input.

2. The dataset must include sequential data, meaning there must be sequences of images taken one after another at a reasonable frame rate, so that individual objects appear in multiple images. Additionally, the images must be in the order they were taken to enable tracking of individual objects across the sequences.

It would also be advantageous if the dataset included labels for objects relevant to autonomous driving, such as cars, pedestrians, traffic lights, etc., as these annotations could prove useful in the subsequent analysis.

After careful consideration of these requirements, **KITTI Multi-Object Tracking (KMOT)** [21] dataset has been selected.

Various acceptable alternatives exist, such as *PandaSet* [22], *nuScenes* [23], and *A2D2* [24]. Therefore, it is important to note that no particular reason exists for choosing the KMOT dataset over the others, as no in-depth comparison has been conducted. The primary reason for its selection is that it meets the requirements, which is satisfactory for this work.

3.1.2 Dataset Overview

The KMOT dataset comprises **21 training sequences** and 29 test sequences. Each sequence contains consecutive images captured from the roof of a car driving through urban areas at a fixed frame rate of 10 frames per second.

Figure 3.1: Example image from KMOT dataset. (sequence: 15, image: 100) [21]

The training sequences are accompanied by human-provided labels. For all conducted experiments, only the training sequences, comprising a total of 8008 images, will be utilized, as the provided labels allow for leveraging them effectively.

The labels contain information that will be utilized in later sections of this work. The relevant components include:

- **Target ID**: Each labeled object is assigned a unique number within the sequence.
- **Object class**: The dataset encompasses 8 different classes of objects, with only the 'Pedestrian' class being relevant for this work.
- **Bounding box**: It is a rectangle that encloses the object and defines its position. It is represented by four decimal numbers: $[x_1, y_1, x_2, y_2]$, where $[x_1, y_1]$ denotes the coordinates of the top right corner and $[x_2, y_2]$ denotes the coordinates of the bottom left corner.
- **Truncation**: This attribute indicates whether an object is truncated, meaning part of the object extends beyond the boundary of the image (0 for non-truncated, 1 for truncated).
- **Occlusion**: This attribute describes the extent of occlusion of an object by other objects (0 for fully visible, 1 for partially occluded, and 2 for largely occluded - the exact distinction between 1 and 2 is not relevant for this work).
- **Location**: Provides the 3D location of the object in camera coordinates, denoted as $[x, y, z]$. The orientation of the axes is irrelevant for this thesis.

It is noteworthy that variations exist in the image sizes among the training sequences, although these inconsistencies do not impact the results of the conducted experiments. Namely, the images come in four different dimensions (height, width): 375x1242, 370x1224, 374x1238, and 376x1241.

3.2 Model

The goal of this section is to leverage the insights gained about modern object detectors to select an appropriate model for this thesis. This section also directly addresses the third objective outlined in section 1.2.

One essential requirement is that the model must be deep-learning-based. Another critical specification is that the model ought to be pre-trained, enabling immediate deployment for object detection. This work focuses on analyzing an existing model rather than training a new one. Additionally, the model should be designed to detect objects relevant to autonomous driving, such as vehicles, traffic signs, and pedestrians.

Furthermore, it would be convenient if the model could process images of arbitrary sizes, as the chosen dataset contains images of very specific dimensions.

According to these criteria, a deep-learning-based two-stage **Faster R-CNN model with a ResNet-50-FPN backbone**, pre-trained on the **COCO dataset** [25], has been chosen.

The model follows the general structure of Faster R-CNNs described in section 2.1.4.3. The second part of the designation specifies the feature-extracting component of the architecture, also known as the backbone. For clarification, ResNet stands for *Residual Network* [26], 50 indicates the number of layers, and FPN refers to the *Feature Pyramid Network* [11].

This model detects a wide variety of classes relevant to autonomous driving, such as 'car', 'person', 'bicycle', 'motorcycle', 'stop sign', 'traffic light', 'truck', and more.

Furthermore, the model can handle images of arbitrary sizes, although it typically resizes them. This resizing process involves setting minimum and maximum size constraints, in this case, 320 and 640, respectively. If either the width or height of the image exceeds these boundaries, bilinear interpolation is used to resize the image to the specified limits while maintaining its aspect ratio.

Another advantage of this model is its accessibility, facilitated by the fact that the pre-trained model is available within the open-source deep-learning framework for Python known as PyTorch. Detailed documentation regarding its utilization and implementation specifics is publicly accessible. [27]

The model output contains bounding boxes for each detected object in the input image. For each bounding box, a label is provided representing the most probable class of the detected object, along with a score, a number within the range of 0 to 1, that represents the detector's confidence in the prediction of the class.

3.3 Explainable AI Method

The XAI methods, along with the object detectors, are the main focus of this work. Therefore, it is important to choose the method for analysis carefully, adjust it if necessary, and set its parameters accordingly. This section directly addresses the fourth objective outlined in section 1.2.

3.3.1 Method Selection

Numerous XAI methods exist, even when narrowed down to those dedicated to explaining the decisions of deep-learning-based object detectors. One particular approach, called pixel attribution methods, has been selected for its straightforward concept of highlighting the pixels of the input image that are important for a specific decision of the analyzed detector. Section 2.2.1 has presented two classifications of such methods to choose from.

The first classification to consider is between gradient-based and occlusion-based methods. While both have their advantages, the occlusion approach has been chosen. This method is preferred because its results are easier to interpret. By ablating parts of the input image and analyzing the changes in the detector's outputs, importance is assigned to the ablated pixels. This is in contrast to the gradient approach, which relies on somewhat abstract interpretations of the backpropagated gradients.

The second decision to make is whether to use black box or white box method. Again, both have their merits, but the black box approach has been chosen. This type is preferred because its implementation is largely independent of the model, and the interpretation of the results remains model-agnostic.

A suitable example of a black box occlusion-based method is the **D-RISE algorithm**, as described in section 2.2.1.4. The subsequent parts of this work will utilize this method.

3.3.2 Method Settings and Adjustments

The D-RISE method requires several parameters to be set before deployment. Additionally, some adjustments can be introduced to the algorithm to facilitate easier analysis in the subsequent stages of this work. This section will refer directly to the section 2.2.1.4.

3.3.2.1 Initial Settings

Note that the normalized version of the algorithm will be used, implying the output pixel attribution map will contain values ranging from 0 to 1.

It is necessary to define the components of the similarity score for the mask score calculation, as specified in equation 2.2. All notation, such as subscripts, refers to the original section.

The *bounding box similarity score* s_L will be kept the same, representing the IoU (Intersection over Union) between boxes.

On the other hand, the *class probability vector similarity score* s_P will be changed. The detector selected in the section 3.2 provides only the probability score of the most probable class for each bounding box. So, it is convenient to represent the similarity score s_P as:

$$s_P(P_i^m, P_T) = p^i, \tag{3.1}$$

where p^i is the aforementioned probability score.

Finally, the *objectness probability score* s_O will not be used, as the selected detector does not provide objectness scores.

3.3.2.2 Method Stability

The primary metric for evaluating the parameter settings and the effectiveness of the improvements introduced in this work will be **stability** of the method.

Since the D-RISE algorithm employs randomness as an essential part of its workflow, the results may vary for the same input and settings. This variability is undesirable for two reasons: firstly, an algorithm intended to explain the decisions of an AI model must provide consistent explanations to be useful, and secondly, for analysis purposes, such instability introduces noise into the data, complicating the interpretation of results. Therefore, the goal is to suppress this variability, thereby enforcing stability.

To measure stability of the algorithm's output, various indicators are proposed:

- **Mean Standard Deviation (MSD)**: The MSD measures the average variability of the pixel attribution values across multiple runs of the D-RISE algorithm. Mathematically, it is defined as:

$$\text{MSD} = \frac{1}{N}\sum_{i=1}^{N}\sigma_i, \tag{3.2}$$

  where N is the total number of pixel locations in the image, and σ_i is the standard deviation of the pixel attribution values at the i-th pixel location across multiple runs of the algorithm. The standard deviation σ_i for the i-th pixel is calculated as:

$$\sigma_i = \sqrt{\frac{1}{M}\sum_{j=1}^{M}(x_{ij} - \mu_i)^2}, \tag{3.3}$$

  where M is the number of runs, x_{ij} is the pixel attribution value at the i-th pixel location in the j-th run, and μ_i is the mean pixel attribution value at the i-th pixel location across all runs, given by:

$$\mu_i = \frac{1}{M}\sum_{j=1}^{M}x_{ij}. \tag{3.4}$$

  A lower MSD indicates higher stability and consistency in the pixel attribution maps produced by the algorithm.

3.3.2.3 Parameter Settings and Method Adjustments

Three parameters need to be configured:

- Parameter $N = 1000$: The stability of the results corresponds to the number of masks used, with more masks leading to increased stability. However, this comes with the drawback of longer explanation generation times as more masks are utilized.
- Parameter $(h \times w) = (5 \times 5)$: The grid size parameter determines the resolution of the method, affecting its ability to capture important details for detection, such as eyes. A smaller value increases the likelihood of detecting finer details, while a larger value generalizes the highlighted areas. However, setting the value too small can lead to instability, while setting it too large may result in overly generalized explanations that are difficult to analyze.
- Parameter $p = 0.5$: The probability of activating (setting to 1) a cell in the binary grid, indicating the percentage of the image being unmasked.

The parameters were selected following a series of experiments aimed at minimizing the MSD (Mean Square Displacement) metric to ensure stability. As demonstrated for example in Figure 3.2, the graph illustrates a clear trend indicating that a larger number of masks results in a smaller MSD, indicative of greater stability.

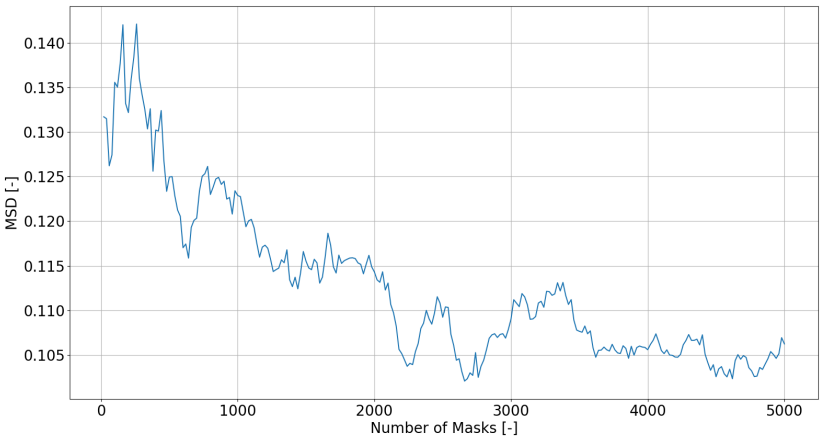


Figure 3.2: Graph depicting the relationship between the MSD (Mean Standard Deviation) instability metric and the parameter N (number of masks) for the D-RISE method. The MSD values were computed based on 10 runs on a single image, ranging from 20 to 5000 masks with increments of 20 masks.

Several adjustments were implemented in the original method to mitigate instability. Notably, instead of masking the entire image, only the region surrounding the bounding box of the detected object is masked. Additionally,

the original black masks were replaced with masks of various colors. Furthermore, the parameter $(h \times w)$ for each mask is subject to random variations within a certain range, introducing variability of analyzed details.

3.4 Experiment Input Data

This section directly addresses the fifth objective outlined in section 1.2, covering the selection of the target class, tracking unique instances of this class across the dataset, and filtering them. Additionally, it both implicitly and explicitly refers to the dataset (3.1) and model (3.2) sections.

3.4.1 Target Class Selection

While selecting the target class, it's crucial to consider the dataset as well as the model.

Although there are multiple class options resulting from the intersection of classes provided by the model and dataset, the class referred to as **'Pedestrian'** throughout this thesis has been chosen. In the dataset context, this class is labeled as 'Pedestrian', whereas in the context of the classes detected by the model, it is 'person'. Instances of the 'Pedestrian' class found in specific images are referred to as *pedestrians*, whereas sets of pedestrians across multiple images that represent the same person in the real world are referred to as *unique pedestrian objects*.

The primary reason for this choice is the significant number of pedestrians in the dataset, coupled with its critical relevance for autonomous driving. The provided labels indicate 167 unique pedestrian objects, comprising 11470 pedestrians within 2529 images.

3.4.2 Object Tracking

Tracking objects across image sequences is a challenging task, even though various modern solutions exist. In this work, the labels provided by the dataset will be leveraged to make tracking effective and straightforward.

Each unique pedestrian object labeled in the dataset is accompanied by a unique target ID and reference bounding box for each frame it appears in.

The tracking comprises the following steps:

1. Run each image of the dataset containing labeled pedestrians through the model and obtain bounding boxes and confidence scores for each detected pedestrian.

2. For each reference box, evaluate all detected boxes in the corresponding image and select the one with the highest Intersection over Union (IoU) with the reference box. Additionally, set a threshold for the minimum IoU to eliminate cases when the detector does not detect the pedestrian instance. In this work, the threshold was set to 0.2.

3. By assigning the target IDs of the reference boxes to the corresponding bounding boxes, valid pedestrian objects across the dataset are effectively tracked.

By the end of this process, each detected pedestrian that is also labeled in the dataset is assigned the ID of the unique pedestrian object it belongs to. The number of detected pedestrians that are also in the dataset is 11055, corresponding to 165 unique pedestrian objects.

3.4.3 Instance Filtering

In object detection, it is common practice to set boundaries for detected objects based on specific use cases. Defining parameters for the analyzed data is also beneficial for ensuring precise and relevant results. Therefore, several constraints for pedestrians are incorporated in this work to optimize the analysis. After each filtering step, number of pedestrians i and number of unique pedestrian objects p is provided.

First, it is essential to set a threshold, 0.5 is used in this work, for the confidence score provided by the detector. If the detector is uncertain about a prediction, the bounding box might be inaccurate or it might misclassify the object, introducing noise into the data. The goal of this thesis is not to analyze incorrect or poor predictions. $[i = 9827, p = 165]$

Another opportunity arises to utilize the labels from the dataset. These labels contain information about pedestrians' occlusion, truncation, and location.

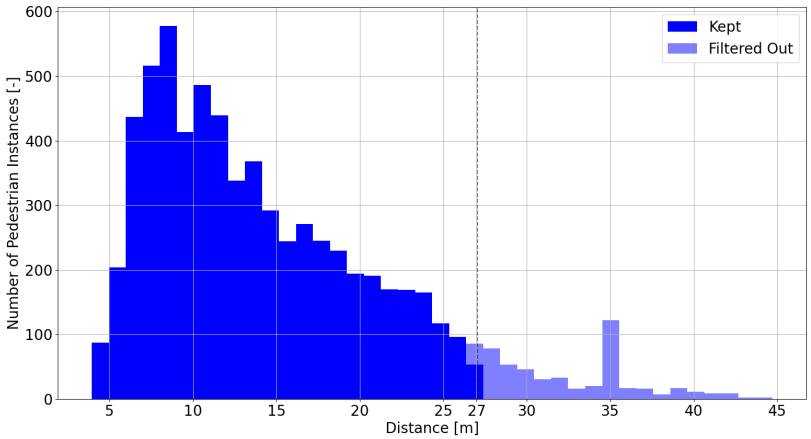


Figure 3.3: Histogram illustrating the distribution of distances from the camera for the previously filtered 6837 pedestrians. Instances beyond the distance threshold of 27 meters are highlighted and labeled as 'Filtered Out', while those within the threshold are labeled as 'Kept'. Note that 12 pedestrians with distances above 45 meters are not shown for convenient visualization.

Using the information about occlusion and truncation to filter out occluded and truncated pedestrians is advantageous. Such cases can, for example, complicate accurate tracking when they are covered by others or result in inaccurate metrics of the XAI outcomes when they are partially out of frame, potentially introducing additional noise into the data. Only pedestrians labeled with 0 for occlusion and truncation are retained. [$i = 6837$, $p = 156$]

The pedestrians' real-world location in camera coordinates can be used to determine their distance from the camera. Since this distance is used in the subsequent analysis, filtering out instances that are too far away is reasonable. In this work, the distance threshold is set to 27 meters, corresponding to the rounded 90th percentile of the filtered pedestrians' distances. This threshold is chosen because distant objects are prone to detection errors and are typically too small for detailed analysis. This step is visualized in Figure 3.3. [$i = 6303$, $p = 150$]

After the filtering steps, 150 unique pedestrian objects, composed of 6303 pedestrians, will be the focus of the subsequent analysis.

3.5 Metrics

The main goal of this thesis is to analyze how explanations provided by an XAI method for autonomous-driving-related object detections change over time for a specific object. Once all the necessary data for this analysis is obtained, defining reasonable metrics is the final step before conducting the actual analysis. This section directly addresses the seventh objective outlined in section 1.2.

When designing the metrics, it's essential to consider their purpose. The metrics are intended to evaluate the output of the experiment pipeline, which consists of pixel attribution maps generated for pedestrians tracked across multiple RGB images each captured from a different viewpoint. Additionally, external information about the pedestrians can be utilized, such as their real-world location in the camera coordinates, location in the image, and dimensions of their bounding boxes.

3.5.1 Frame Number Metric

Each unique pedestrian may be detected in multiple dataset images. The frames of a pedestrian are understood as the dataset images in which the pedestrian appears.

It is important to note that the frames are numbered by their position relative to the dataset image that corresponds to the first appearance of the pedestrian. This means that each frame is indexed based on its position in the sequence of the dataset images, starting from the frame where the pedestrian first appears.

For example, assume a pedestrian appears in dataset image 231, is present in each image until 240 inclusive, then disappears, only to reappear in image 245 and finally disappears after image 250. In this case, image 231 corresponds

to the frame number 1, image 240 to the frame number 10, and so forth. When the pedestrian reappears in image 245, this image corresponds to the frame number 15, and image 250 to the frame number 20. This numbering includes images in which the pedestrian does not appear.

This numbering allows the frames to be interpreted as consistent time representations, providing a clear method for tracking the pedestrian's presence and analyzing changes in the dataset over time.

3.5.2 Distance Metric

The distance of a detected pedestrian from the camera will also be utilized in the subsequent analysis. This metric not only indicates the size of the pedestrian in the image but also reflects changes over time as the distance varies for the specific pedestrian object across multiple images.

The distance can be estimated by considering camera calibration, the pixel height of the pedestrian, and an approximation of their actual height. Leveraging the dataset labels, which contain information about the pedestrian's location in camera coordinates, makes this process much easier.

The location, represented as (x, y, z), can be converted to distance d using the equation:

$$d = \sqrt{x^2 + y^2 + z^2}. \tag{3.5}$$

Here, x, y, and z denote the respective coordinates of the pedestrian in the camera's reference frame.

3.5.3 Body Part Attention Metrics

XAI pixel attribution methods operate by highlighting pixels within an analyzed image that are important for a detector's prediction of an instance of a target object. This output is easily interpretable: For example, if the detector detects a person and the pixel attribution map highlights the person's legs, it is reasonable to deduce that the legs are what the detector focused on when detecting the person in this specific case.

Following this concept, a set of metrics is proposed, labeled as **Body Part Attention (BPA) Metrics**. Here, "attention" refers to the values of the pixel attribution map, indicating the model's focus on specific body parts.

3.5.3.1 Pedestrian Pose Estimation

The initial step in obtaining the BPA metrics is to identify the body parts of the detected pedestrians within the analyzed image.

This thesis defines such pedestrian body parts using **keypoints** that are characterized by its position in the analyzed image, denoted as (x, y).

For this purpose, a widely recognized deep learning system called **OpenPose** [28] is employed. The OpenPose model is used to estimate the pose of each pedestrian, as illustrated in Figure 3.4.

Figure 3.4: Pose estimation for a pedestrian showing 12 estimated keypoints. Dataset specifications: [Sequence: 15, Image: 100, Pedestrian ID: 5]

This work recognizes 18 keypoints, as listed in Table 3.1. In addition to the single keypoint body parts, several group keypoint body parts are defined to be utilized in the subsequent analysis, as shown in Table 3.2.

It is necessary to note that the pose estimations may introduce additional noise to the data as inaccuracies in keypoint detection can occur due to occlusions, varying lighting conditions, and the inherent limitations of the model. These inaccuracies can affect the reliability of the group keypoint body parts and, consequently, the analysis based on them.

Three 2D histograms are presented to illustrate important information about the generated poses. The data is visualized with respect to pedestrian distances ranging from 3.8 m to 27 m, divided into 15 equidistant intervals. This format will be utilized in the subsequent analysis within this work.

The first histogram, depicted in Figure 3.5, illustrates the frequency of detected pedestrian poses containing specific keypoint types within distinct distance interval. A noteworthy observation is that majority of poses occur around 8 meters, aligning with the distribution of pedestrian poses illustrated in Figure 3.3. This trend is consistent across all three histograms. The red dots highlight the keypoint type with the highest occurrence within each distance interval. Notably, the 'neck' keypoint emerges as the most

frequently detected across various distances, although similar occurrences are also observed.

The second histogram, illustrated in Figure 3.6, reveals that the eye, ear, wrist, and nose keypoints are frequently absent in the detected pedestrian poses.

The third histogram, depicted in Figure 3.7, demonstrates that within approximately 20 meters of distance, pedestrian poses predominantly lack 7 or fewer keypoints. Beyond this distance, the incidence of missing keypoints increases, with the most distant pedestrians frequently lacking all keypoints and thus being excluded from subsequent analysis.

3.5.3.2 BPA Metrics Calculation

The second step is to utilize the obtained body parts to calculate the specific metrics:

- **Body Part Attention Sum (BPA-Sum)**: The BPA-Sum metric shows how much attention (pixel attribution) is within the analyzed pixel attribution map around specific body part.

  The calculation process can be divided into five steps:

  1. For each keypoint $k_i = (x_i, y_i)$, where $i \in (1, \ldots, K)$ and K denotes the total number of keypoints for the target body part, generate a blank mask M_i. This mask is essentially an image with a single color channel ranging from 0 to 1 for pixel values. It is the same size as the analyzed pixel attribution map P.
  2. Fill each mask M_i with a Gaussian circle around the keypoint k_i with radius r.

     The Gaussian circle is generated using the Gaussian function, which assigns intensity values (ranging from 1 to 0) to pixels based on their distance from the keypoint. The intensity of each pixel (x, y) in the mask M_i is determined as follows:

     $$\begin{aligned} M_i(x, y) &= e^{-\frac{d_i^2(x,y)}{2r^2}}, \\ d_i(x, y) &= \sqrt{(x - x_i)^2 + (y - y_i)^2}, \end{aligned} \tag{3.6}$$

     where d_i is the Euclidean distance from the pixel (x, y) to the keypoint k_i. This function ensures that pixels closer to the keypoint have higher intensity values, gradually decreasing as distance from the keypoint increases.

     The radius r is determined by the bounding box height of the analyzed pedestrian, based on the assumption that the body parts scale proportionally with this height. In this thesis, the radius for a pedestrian with a bounding box height h s calculated as follows: $r = h/30$.

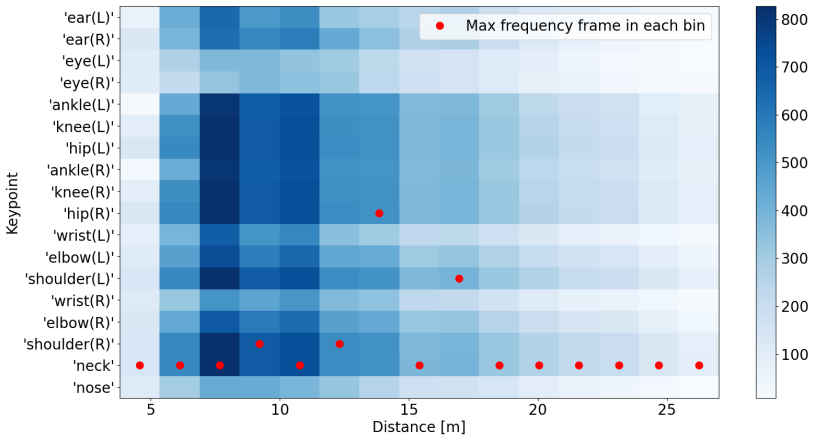


Figure 3.5: Distribution of pedestrian pose keypoint appearences across pedestrian distances from the camera, ranging from 3.8 m to 27 m and divided into 15 equidistant bins. The shade of blue in each frame represents the frequency of the keypoint appearing in the pedestrian poses in the distance interval. Additionally, a red dot marks the dominant keypoint for each distance interval.

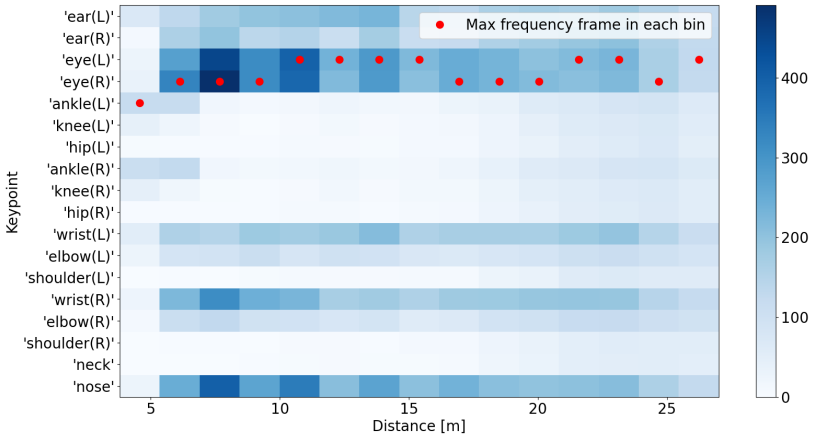


Figure 3.6: Distribution of missing pedestrian pose keypoints across pedestrian distances from the camera, ranging from 3.8 m to 27 m and divided into 15 equidistant bins. The shade of blue in each frame represents the frequency of missing keypoints in the pedestrian poses within the distance interval. Additionally, a red dot marks the dominant keypoint for each distance interval.

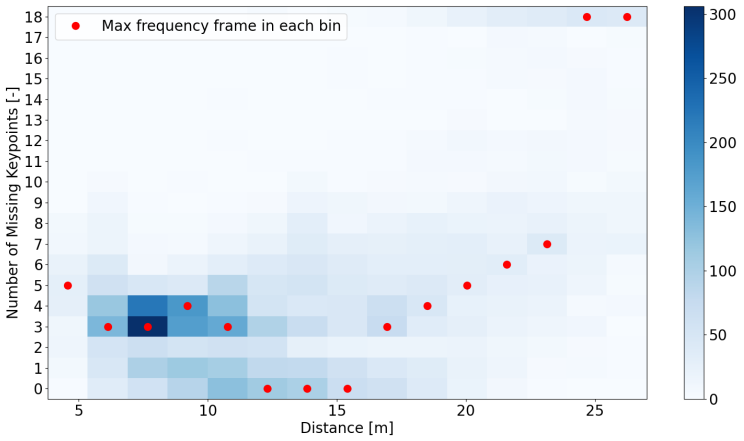


Figure 3.7: Distribution of numbers of missing pedestrian keypoints in pedestrian poses across equidistant pedestrian distances divided into 15 bins. Shades of blue indicate the frequency of poses with the specific number of missing keypoints within each distance interval. Additionally, the predominant number of poses within each distance bin is highlighted.

Keypoints
'nose'
'neck'
'shoulder(R)'
'elbow(R)'
'wrist(R)'
'shoulder(L)'
'elbow(L)'
'wrist(L)'
'hip(R)'
'knee(R)'
'ankle(R)'
'hip(L)'
'knee(L)'
'ankle(L)'
'eye(R)'
'eye(L)'
'ear(R)'
'ear(L)'

Table 3.1: List of Keypoints. (L) and (R) indicate left and right from the perspective of the pedestrian.

3. Create a combined mask M by taking the maximum value for each pixel across all M_i masks:
   The combined mask M is generated by determining, for each pixel (x, y), the maximum value across all individual masks M_i. Mathematically, this operation can be expressed as:

$$M(x, y) = \max_{i=1}^{K} M_i(x, y). \tag{3.7}$$

4. Multiply pixel-wise the combined mask M with the pixel attribution map P to obtain a map MP with attention around the body part's keypoints:

$$P_M = P \odot M. \tag{3.8}$$

5. Sum the pixel values of the P_M map to obtain the desired BPA-Sum:

$$\text{BPA-Sum} = \sum_{x=1}^{W} \sum_{y=1}^{H} P_M(x, y), \tag{3.9}$$

   where W and H denote the width and height of the pixel attribution map, respectively.

- **Body Part Attention Average (BPA-Average)**: The BPA-Average metric indicates the average level of attention (pixel attribution) within the analyzed pixel attribution map around a specific body part.

  The calculation process utilizes the BPA-Sum. Therefore, once it is calculated, two additional steps are required:

  1. Sum the pixel values of the combined mask M:

$$\text{M-Sum} = \sum_{x=1}^{W} \sum_{y=1}^{H} M(x, y), \tag{3.10}$$

     where W and H denote the width and height of the pixel attribution map, respectively.

  2. Divide the BPA-Sum by the M-Sum to obtain the desired BPA-Average:

$$\text{BPA-Average} = \frac{\text{BPA-Sum}}{\text{M-Sum}}. \tag{3.11}$$

- **Relative Body Part Attention (Relative-BPA)**: The Relative-BPA represents the attention a body part draws relative to other parts of the image.

  The calculation process also utilizes the BPA-Sum. Therefore, once it is calculated, two additional steps are required:

  1. Sum the pixel values of the pixel attribution map P:

$$\text{P-Sum} = \sum_{x=1}^{W} \sum_{y=1}^{H} P(x, y), \tag{3.12}$$

where W and H denote the width and height of the pixel attribution map, respectively.

2. Divide the BPA-Sum by the P-Sum to obtain the desired Relative-BPA:

$$\text{Relative-BPA} = \frac{\text{BPA-Sum}}{\text{P-Sum}}. \tag{3.13}$$

Body Part	Components
'shoulders'	'shoulder(L)', 'shoulder(R)'
'elbows'	'elbow(L)', 'elbow(R)'
'wrists'	'wrist(L)', 'wrist(R)'
'hips'	'hip(L)', 'hip(R)'
'knees'	'knee(L)', 'knee(R)'
'ankles'	'ankle(L)', 'ankle(R)'
'eyes'	'eye(L)', 'eye(R)'
'ears'	'ear(L)', 'ear(R)'
'leg(L)'	'hip(L)', 'knee(L)', 'ankle(L)'
'leg(R)'	'hip(R)', 'knee(R)', 'ankle(R)'
'arm(L)'	'shoulder(L)', 'elbow(L)', 'wrist(L)'
'arm(R)'	'shoulder(R)', 'elbow(R)', 'wrist(R)'
'arms'	'arm(L)', 'arm(R)'
'legs'	'leg(L)', 'leg(R)'
'head'	'nose', 'eyes', 'ears'
'lower body'	'legs', 'wrists', 'elbows'
'upper body'	'shoulders', 'neck', 'head'
'full body'	'legs', 'arms', 'neck', 'head'

Table 3.2: Body Parts and Their Components. (L) and (R) indicate left and right from the perspective of the pedestrian.

Chapter 4

Analysis

This chapter aims to analyze the outcomes from Chapter 3 in a manner that directly addresses the eighth objective outlined in section 1.2. This objective directs the analysis towards describing how D-RISE explanations of the selected Faster R-CNN detector's decisions change over time for filtered unique pedestrian objects detected in the KMOT dataset images, utilizing BPA metrics.

The structure of this chapter is influenced by the basis on which the explanation change is observed. Section 4.1 explores the assigned method of analyzing changes over time by using the consecutive images (frames) in which pedestrians appear. In contrast, section 4.2 follows a method of analyzing changes based on the distances of pedestrians from the camera.

4.1 Frame-Based Analysis

The frame-based analysis is founded on the straightforward premise that the dataset images were captured sequentially at a fixed rate. Therefore, it is reasonable to analyze the change in the explanation over the frames, as they represent discrete consecutive time intervals in the observed scene. The method for obtaining frames for each pedestrian is described in Section 3.5.1.

A convenient way to visualize this approach is to plot the BPA metric against the frame numbers for each pedestrian. An example is shown in Figure 4.1.

The graph illustrates the dependence of the BPA-Sum metric for the body part 'full body' on the frame number for all pedestrians. Due to the large number of data points plotted, it would not be possible to distinguish each pedestrian individually. Therefore, only three pedestrians were highlighted, while the others were suppressed. The 'Pedestrian 1' and 'Pedestrian 2' exhibit a typical dependency pattern, while 'Pedestrian 3' demonstrates a more unique behavior.

For illustration, three specific frames were selected for 'Pedestrian 1': A1, B1, and C1, with explanations visualized in subfigures (a), (b), and (c) of Figure 4.5, respectively. For 'Pedestrian 3', also three frames were selected: A3, B3, and C3, with explanations provided in Figure 4.6.

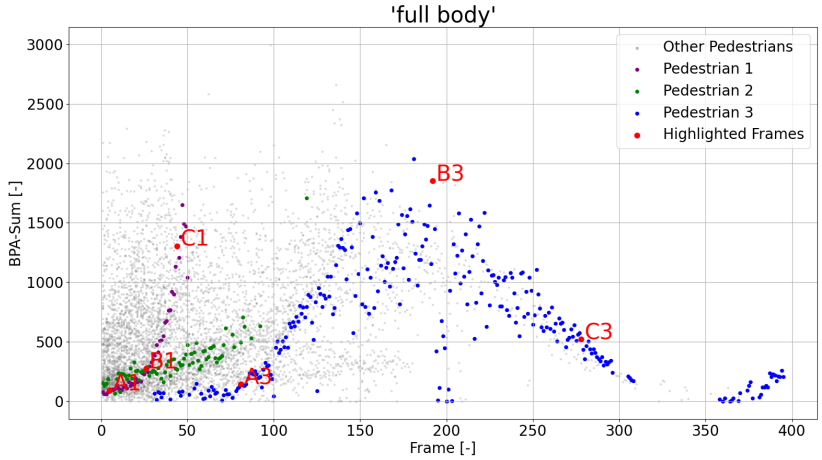


Figure 4.1: Dependence of the BPA-Sum metric for the body part 'full body' on the frame number for all pedestrians. The graph illustrates the variation in the BPA-Sum metric across different frame numbers, highlighting the dependencies for three distinct pedestrians.

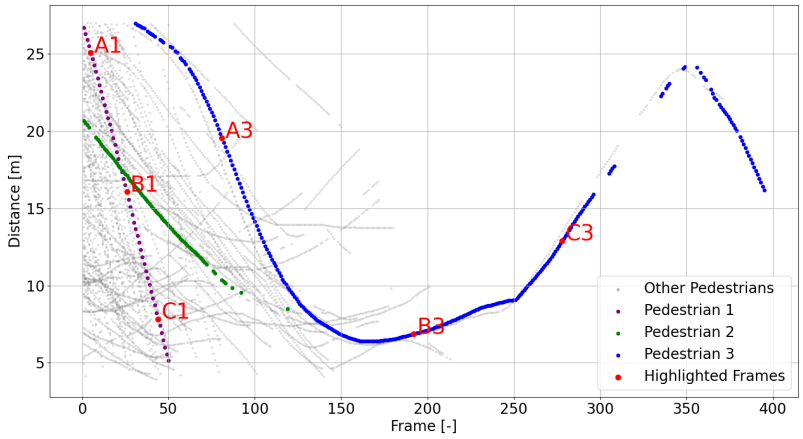


Figure 4.2: Dependence of the distance metric on the frame number for all pedestrians. Highlighted are the dependencies for the same three pedestrians as in Figure 4.1.

Upon closer examination of the results, it appears that the BPA-Sum 'full body' value within a pedestrian sequence depends more on the pedestrian's distance from the camera than on the frame number itself.

To support this theory, the Figure 4.2 depicting the dependence of distance on the frame number can be utilized. By comparing the two graphs, it is evident from the patterns observed among the three highlighted pedestrians that as the distance decreases, the BPA-Sum 'full body' increases for the same frame number, and vice versa. This behavior would be observed for other pedestrians if they were highlighted as well.

To explain this conclusion, it's essential to interpret the nature of the metric. The BPA-Sum 'full body' reflects the concentration of the model's attention around the pedestrian's body, determined by summing the attention values of pixels within the pedestrian's body region. Consequently, the metric's value depends on the number of relevant pixels, therefore it increases as the pedestrian is closer to the camera because it results in a larger representation of the pedestrian in the image.

Among the highlighted pedestrians in Figure 4.2, the distinguishable lines of dots represent dependencies for individual pedestrians. Closer examination of the graph reveals that, generally, as the frame number increases, the distance decreases. This trend makes sense as the car approaches the pedestrian when they are in the image because the camera points in the driving direction, and the distance increases as the car moves away when the pedestrian is out of view.

The unique dependency pattern of 'Pedestrian 3' is caused by the fact that the car first approaches the pedestrian (point A3, Figure 4.1), then stops (point B3), while the pedestrian continues to walk away (point C3).

Another noteworthy observation is the absence of metric data for frame numbers between approximately 320 and 360. This absence could be attributed to three potential explanations. First, pedestrian instances within this range may have been filtered out due to factors such as occlusion, truncation, or low detector confidence. Second, the detector may have failed to detect the pedestrians altogether. Third, regarding the BPA metrics, it is possible that insufficient body parts were detected during this period. This is more likely for pedestrians who are farther away, as indicated by Figure 3.7.

Figures 4.3 and 4.4 serve to illustrate examples of the other two BPA metrics. By highlighting the data points corresponding to the same pedestrians as in the previous analysis, these figures provide a basis for comparison. Although there is potential for deeper analysis of specific cases, such as the values around frame number 200 for the 'Pedestrian 3' data points, this will not be further elaborated in this work.

The secondary reason for not delving deeper is the complexity of interpreting the data as a whole. The analysis thus far has focused on individual pedestrians, but with approximately 150 pedestrians and 3 BPA metrics for several body parts to analyze, continuing this detailed analysis would be impractical.

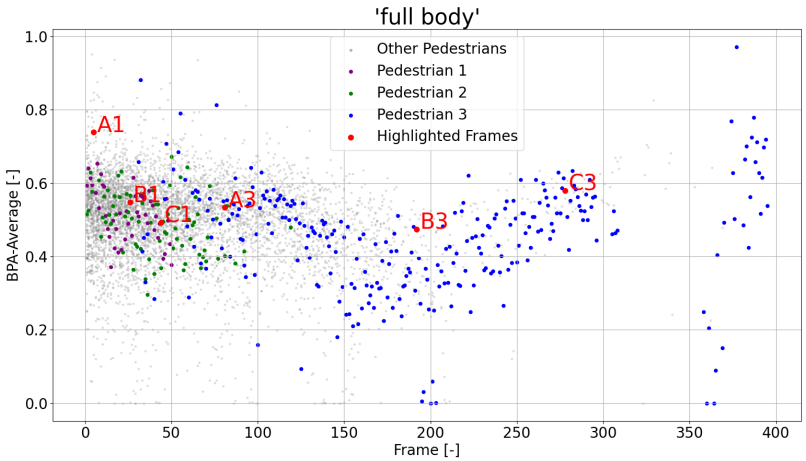


Figure 4.3: Dependence of the BPA-Average metric for the body part 'full body' on the frame number for all pedestrians. Highlighted are the dependencies for the same three pedestrians as in Figure 4.1.

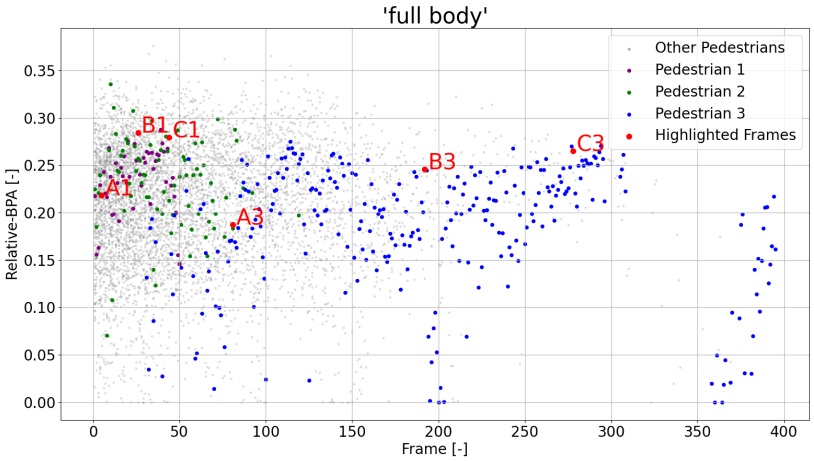
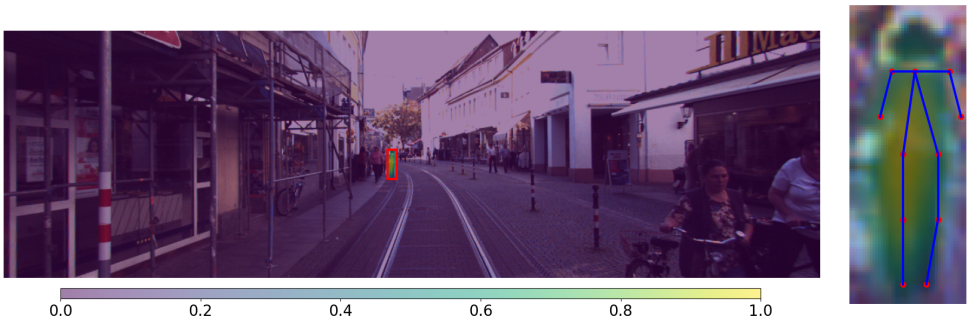
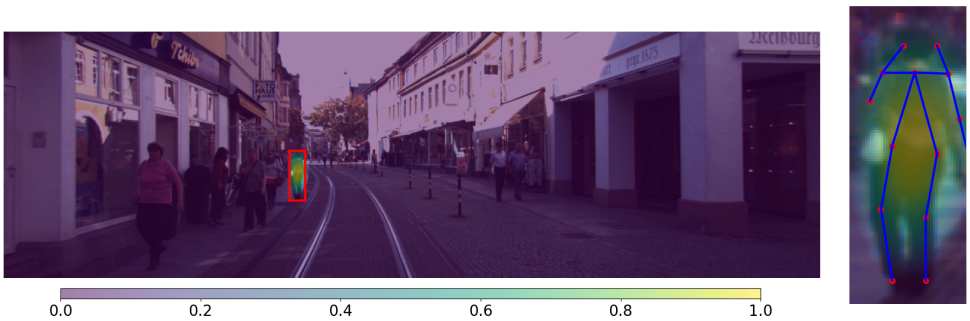


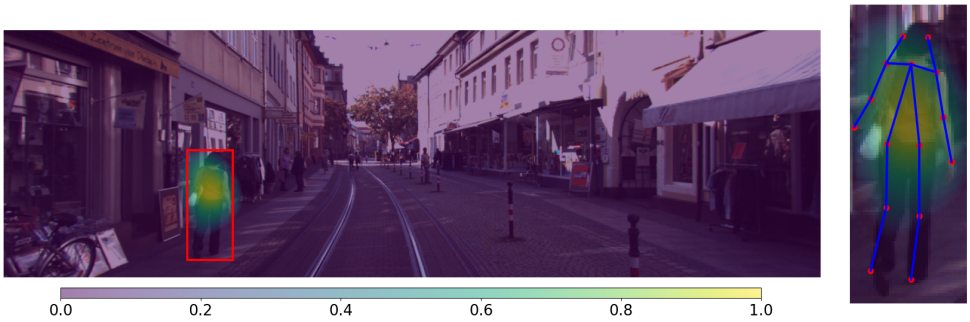
Figure 4.4: Dependence of the Relative-BPA metric for the body part 'full body' on the frame number for all pedestrians. Highlighted are the dependencies for the same three pedestrians as in Figure 4.1.



(a) : Image: 237 (A1)

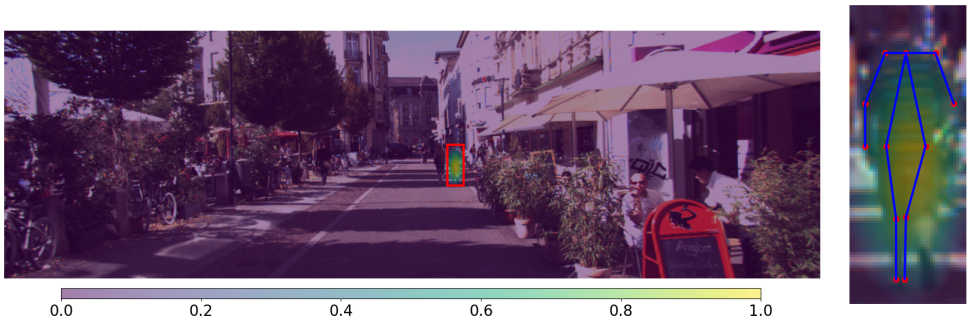


(b) : Image: 258 (B1)

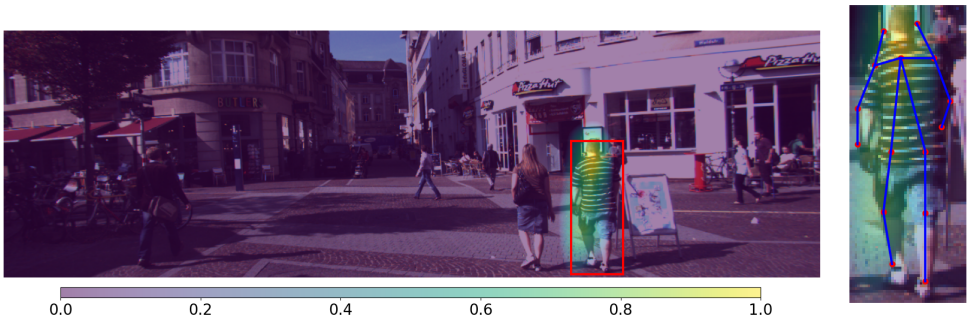


(c) : Image: 276 (C1)

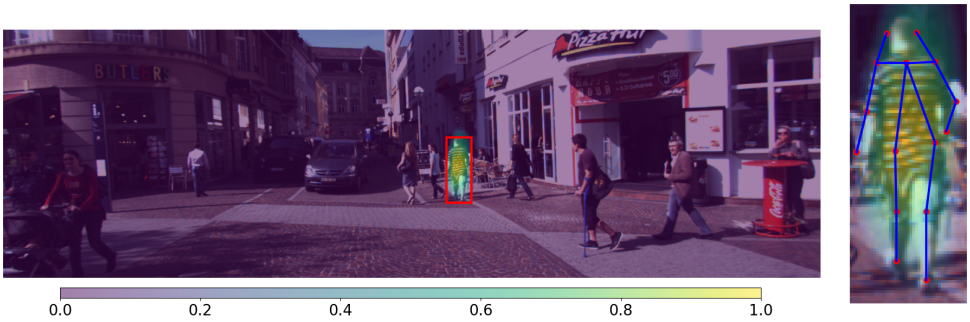
Figure 4.5: Three images displaying a target pedestrian with red bounding boxes predicted by the object detector, overlaid with pixel attribution maps. Cropped images on the right show the pedestrian pose visualized within the bounding boxes. Each image's caption represents its number within the dataset sequence. Dataset specifications: [Sequence: 12, Pedestrian ID: 42].



(a) : Image: 443 (A3)



(b) : Image: 554 (B3)



(c) : Image: 640 (C3)

Figure 4.6: Three images displaying a target pedestrian with red bounding boxes predicted by the object detector, overlaid with pixel attribution maps. Cropped images on the right show the pedestrian pose visualized within the bounding boxes. Each image's caption represents its number within the dataset sequence. Dataset specifications: [Sequence: 19, Pedestrian ID: 48].

The primary reason lies in the limitations of the frame-based analysis approach and the time analysis in general. Since the detector makes decisions based on individual static images, these decisions are influenced by the current state of the object, not by the sequence of states leading up to it or following it. These states may include factors such as whether the pedestrians face the camera, their proximity to the camera, their body posture (e.g., whether their legs are close to each other or whether their arms are close to the body), the colors they wear, and other such attributes.

For the frame-based approach, this means that for the same frame number, each object is typically in a different state. Therefore, this method (and time analysis in general) cannot work at all without incorporating external information about the state of the object, such as the pedestrian's distance from the camera.

4.2 Distance-Based Analysis

The distance-based analysis focuses on observing the dependency of changes in the explanations on one of the pedestrians' states, specifically their distance from the camera.

It is reasonable to start with the same example as in the previous method. In Figure 4.7, a clear dependence of the BPA-Sum 'full body' metric on the distance is recognizable, as described in the previous section 4.1 (paragraph 4 and onwards).

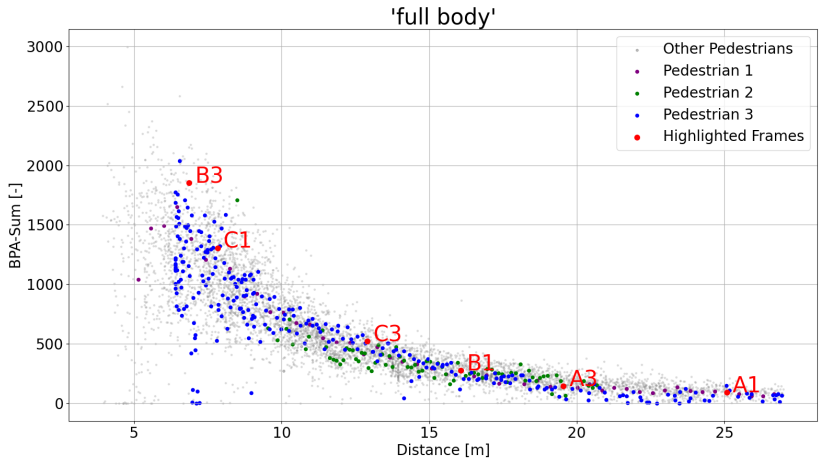


Figure 4.7: Dependence of the BPA-Sum metric for the body part 'full body' on the distance from camera for all pedestrians. Highlighted are the dependencies for the same three pedestrians as in Figure 4.1.

Although a detailed analysis of the metrics considering individual pedestrians could be valuable, it would be impractical due to the high number of pedestrians and data points.

With a stable ground for interpretation established, it is more convenient to divide the x-axis (distances) into equidistant intervals and, for each interval, calculate the average value and standard deviation of the metric. Plotting these average values provides an overview of the metric's dependence on distance for an average pedestrian.

The number of intervals has been experimentally set to 15 to capture general trends in the dependencies within the individual metrics. The minimum visualized distance is 3.8 meters, corresponding to the minimum distance in the analyzed data, which is slightly under 4 meters. The maximum visualized distance is 27 meters, as pedestrians farther than that were filtered out.

It is important to consider the distribution of pedestrian distances, as shown in Figure 3.3, since the number of points in the intervals from which the averages and standard deviations are calculated may vary significantly. This variance can affect the reliability of the calculated metrics for each interval.

Additionally, because the BPA metrics are based on the detected keypoints of the pedestrian poses, the information in Figures 3.5, 3.6, and 3.7 regarding the number of keypoints across different distance intervals might also be important.

4.2.1 BPA Metrics

The first set of graphs is depicted in Figure 4.8. It illustrates the dependency of the BPA-Sum metric on the pedestrian's distance for various body parts defined in Table 3.2. The graph labeled 'full body' corresponds to the detailed graph shown in Figure 4.7.

A notable observation is the significant variation in the y-axis range for individual body parts. This corresponds to the dependency discussed in section 4.1 (paragraph 6), where the BPA-Sum metric relies on the number of pixels of the pedestrian. For instance, the body part 'upper body' can consist of up to 6 keypoints, while the body part 'ankles' only has a maximum of 2 keypoints. Generally, more keypoints correspond to more pixels and thus more attention.

Additionally, the dependencies for all body parts generally adhere to the logical rule that smaller distance results in increased attention. This correlation underscores the principle that as the pedestrian moves closer, they tend to occupy more pixels in the image, thus attracting greater attention. Also, the poses of closer pedestrians typically contain more keypoints, as shown in Figure 3.7. However, despite this general trend, some minor inconsistencies occur.

In this regard, there are two distinct groups of body parts. The first group, comprising 'shoulders', 'neck', 'head', and 'upper body', consistently adheres to this rule across all intervals, with attention increasing as the pedestrian approaches.

However, the second group, consisting of 'ankles', 'knees', 'hips', 'legs', 'elbows', 'wrists', 'lower body', and 'full body', exhibits slight inconsistency

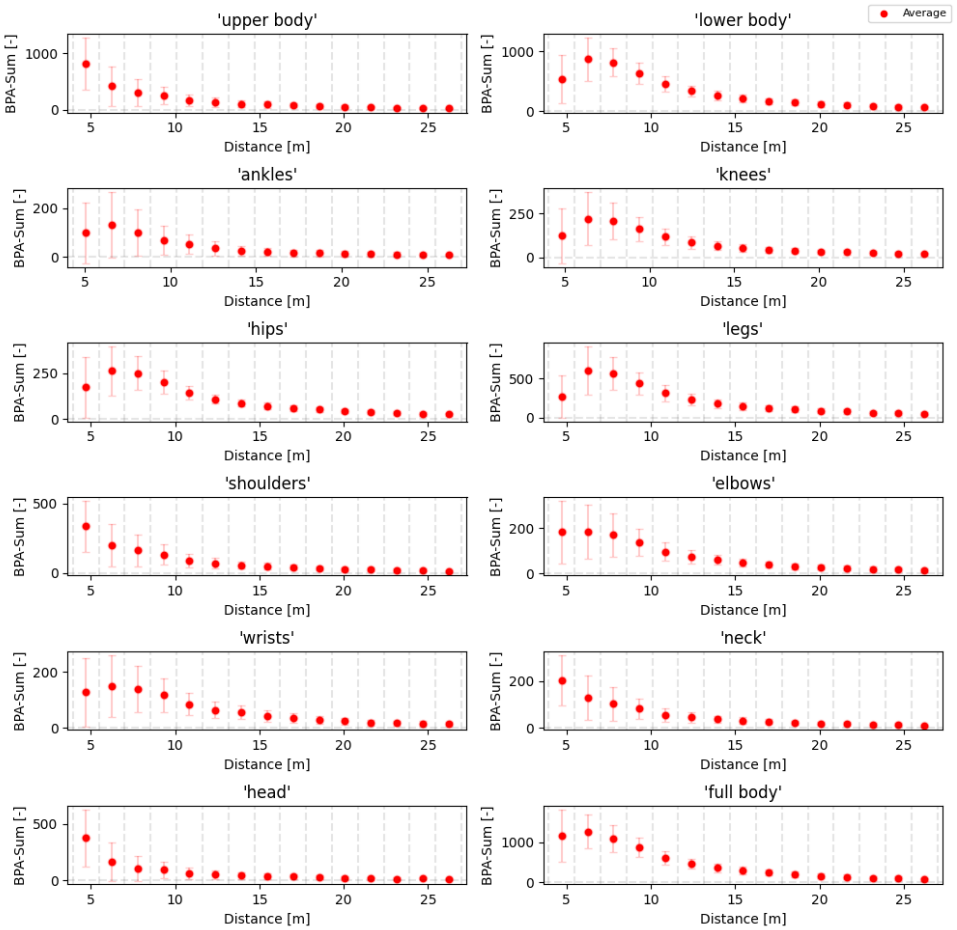


Figure 4.8: Dependence of the BPA-Sum metric for twelve body parts on the distance from the camera for all pedestrians. The x-axis is divided into 15 equidistant intervals from 3.8 m to 27 m. Red points indicate the average metric values within each interval, with error bars representing the standard deviation.

with this rule in the first two intervals, as the first interval shows a smaller value than the second one.

The first group will be represented by the 'upper body,' while the second group, excluding 'full body', will be represented by the 'lower body.' The 'full body' will stand alone as an indicator of the overall dependency. It is convenient to focus solely on analyzing these two body parts because they encompass the other ones, as shown in Table 3.2, and can adequately represent the observed groups.

Note that 'wrists' and 'elbows' are listed under the 'lower body'. This categorization is based on the premise that although anatomically attached to the 'shoulders' from the 'upper body', they typically occupy lower positions in relation to the body.

It is also important to note that while the standard deviation tends to increase as the distance decreases, this observation does not provide new insights into the dependency on distance. This is because the values used

to calculate the average cannot be negative. Therefore, as the average value increases, the standard deviation naturally grows as well. Instead, the information about standard deviation is utilized to visualize the spread of the points within each interval.

To gain more insight into the differences between the two established groups, it is necessary to plot more detailed graphs for the 'upper body' and 'lower body', as shown in Figures 4.9 and 4.10.

The first notable observation when comparing the graphs is that the relative standard deviations for the 'lower body' are generally lower across all distances, indicating greater stability for this metric. The only exception is the first interval.

The 'upper body' graph behaves as expected, with the BPA-Sum decreasing with distance. The 'lower body' graph shows a similar trend, except for the first interval. Therefore, it is reasonable to focus on explanations and pedestrians corresponding to the data points in this interval.

Six instances from this interval were randomly selected to cover the range of BPA-Sum 'lower body' values and are marked in Figures 4.9 and 4.10 by blue points and numbers. These cases are visualized in order in Figures 4.15 and 4.16.

The second set of graphs is depicted in Figure 4.11. It illustrates the dependency of the BPA-Average metric on the pedestrian's distance for various body parts. The metric represents the average attention around the specific body parts.

The established groups persist for this metric as well, which is expected since, although the BPA-Average metric refines the BPA-Sum by removing the direct dependency on the number of pixels, it is still derived from it.

For the group represented by 'upper body', the BPA-Average value remains consistent across distances, hovering around 0.5 relative to the maximum value of 1.0.

Conversely, the group represented by 'lower body' shows varying metric values, starting low for closer pedestrians, rising in the first meters, and then stabilizing. Typically, these values hover around 0.5, with one exception: the 'hips' body part, which converges to a value around 0.8.

By comparing the first intervals of both groups, it is evident that the 'lower body' has a smaller average attention compared to the 'upper body'.

Additionally, the error bars indicate that the 'lower body' group generally exhibits smaller standard deviations across the distance intervals compared to the 'upper body' group, suggesting greater consistency.

The third set of graphs is depicted in Figure 4.8 and illustrates the dependency of the Relative-BPA metric on the pedestrian's distance for various body parts. This metric represents the proportion of the model's total attention that is focused around specific body part.

The graphs also demonstrate consistent dependencies for the two body part groups.

The 'upper body' shows a slight decline in the values of the Relative-BPA metric over the first few meters before stabilizing. The values range in units

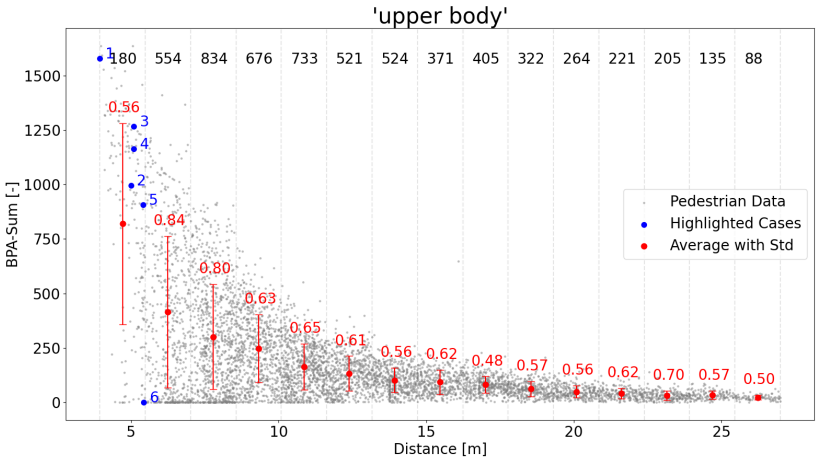


Figure 4.9: Dependence of the BPA-Sum metric for the body part 'upper body' on the distance from the camera for all pedestrians. The x-axis is divided into 15 equidistant intervals ranging from 3.8 m to 27 m. For each interval, the average value (red dot), standard deviation (red error bars), relative standard deviation (red numbers), and number of points (black number) are calculated. Blue points represent cases referred to in section **??**.

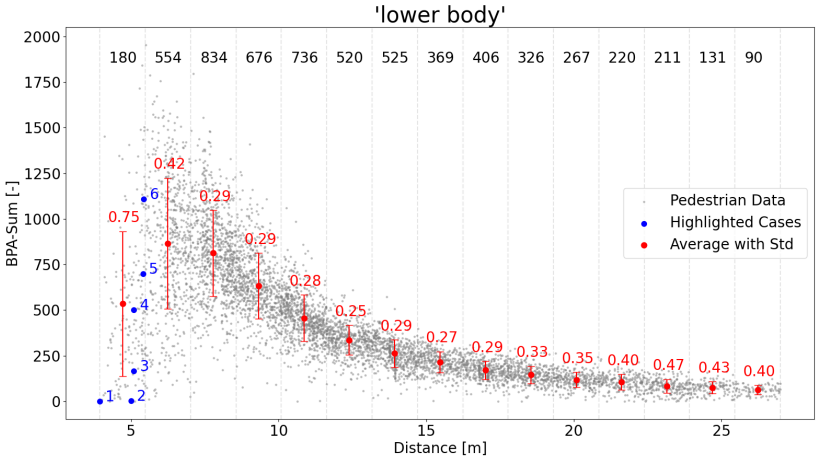


Figure 4.10: Dependence of the BPA-Sum metric for the body part 'lower body' on the distance from the camera for all pedestrians. The x-axis is divided into 15 equidistant intervals ranging from 3.8 m to 27 m. For each interval, the average value (red dot), standard deviation (red error bars), relative standard deviation (red numbers), and number of points (black number) are calculated. Blue points represent cases referred to in section 4.2.1.

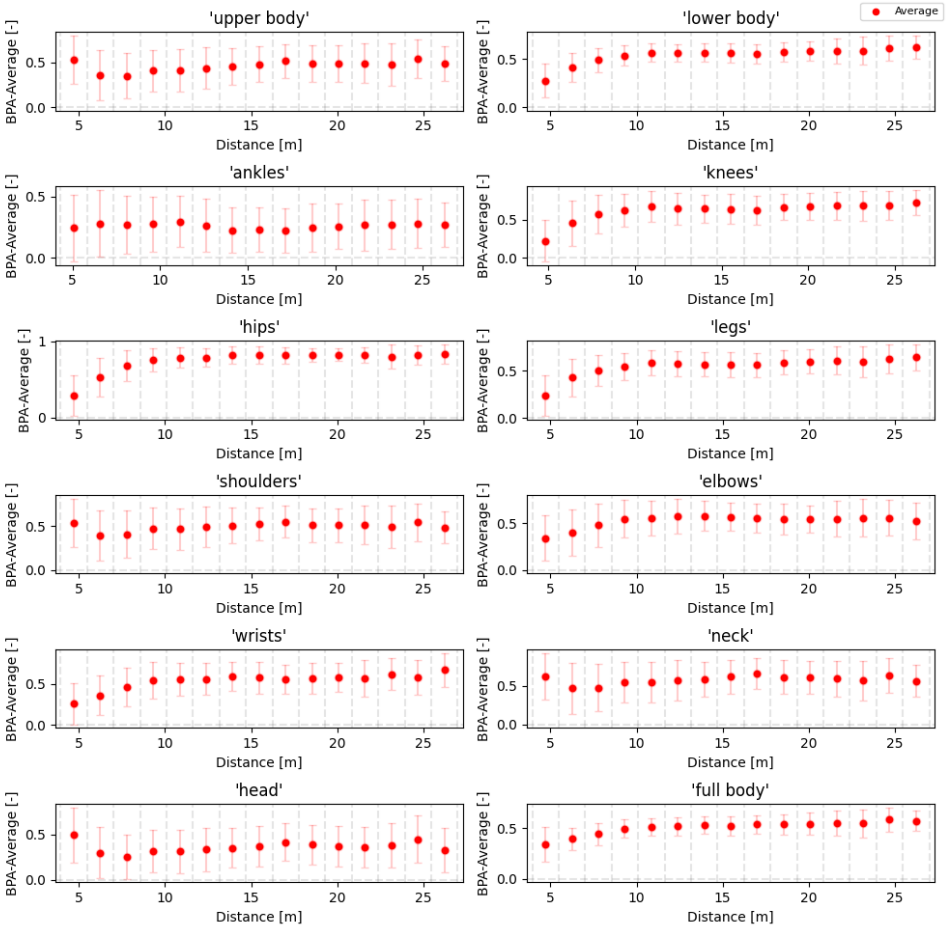


Figure 4.11: Dependence of the BPA-Average metric for twelve body parts on the distance from the camera for all pedestrians. The x-axis is divided into 15 equidistant intervals from 3.8 m to 27 m. Red points indicate the average metric values within each interval, with error bars representing the standard deviation.

of percentages of the total attention for individual body parts. Specifically, the attention for the 'upper body' starts at around 10 percent for closer pedestrians and decreases to around 5 percent for the most distant ones.

For the 'lower body', the metric values initially rise from around 2 percent to approximately 18 percent within the first few meters. After reaching a distance of approximately 17 meters, the values begin to decline, eventually stabilizing around 10 percent.

By comparing both groups, it is evident that the 'lower body' dominates over the 'upper body' across all distance intervals, with the exception of the first interval.

An interesting observation is that the relative attention for 'full body' remains approximately 20 percent across most distances. This suggests that 80 percent of the attention is concentrated elsewhere. Based on visualizing

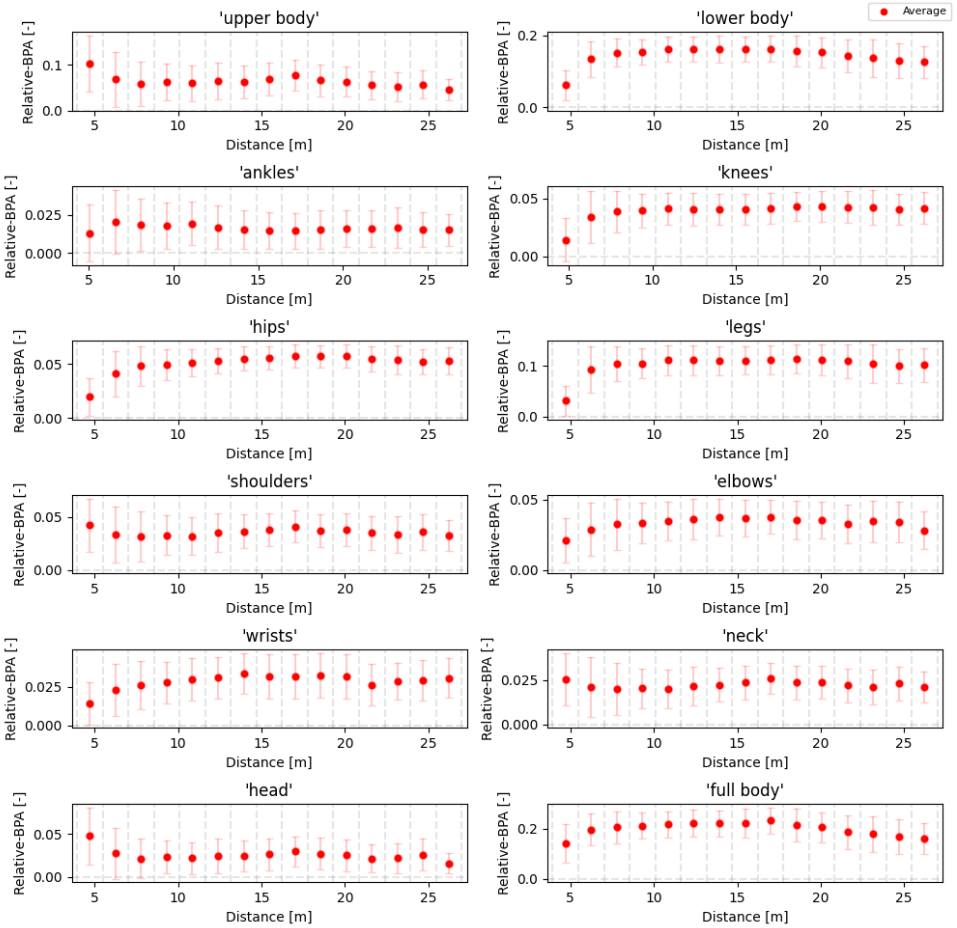


Figure 4.12: Dependence of the Relative-BPA metric for twelve body parts on the distance from the camera for all pedestrians. The x-axis is divided into 15 equidistant intervals from 3.8 m to 27 m. Red points indicate the average metric values within each interval, with error bars representing the standard deviation.

the results, as shown in Figure 4.6, the dispersion in between the detected keypoints appears to offer the best explanation for this distribution.

4.2.2 High and Low Attention Keypoints

The keypoints that consistently attract the highest attention across the distance intervals are generally the hip and knee keypoints, as depicted in Figure 4.13. However, an exception is observed in the first interval, where the 'upper body' keypoints such as 'neck', 'ear(R)', or 'shoulder(L)' dominate.

Conversely, the Figure 4.14 shows that the keypoints that consistently attract the least attention across the distance intervals are the ankle along with the ear keypoints. Once again, the first interval deviates from this trend, with the knee keypoints dominating.

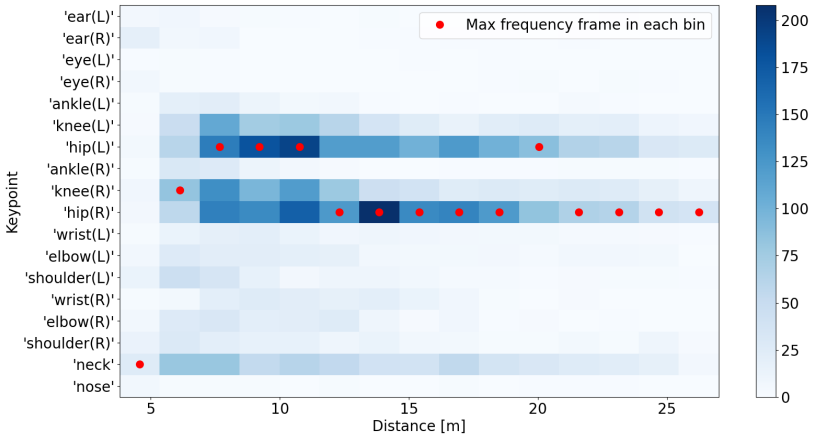


Figure 4.13: Distribution of pedestrian pose keypoints with the highest attention across pedestrian distances from the camera, ranging from 3.8 m to 27 m and divided into 15 equidistant bins. The shade of blue in each frame represents the frequency of the keypoint receiving the most attention relative to others in the interval. Additionally, a red dot marks the dominant keypoint for each distance interval.

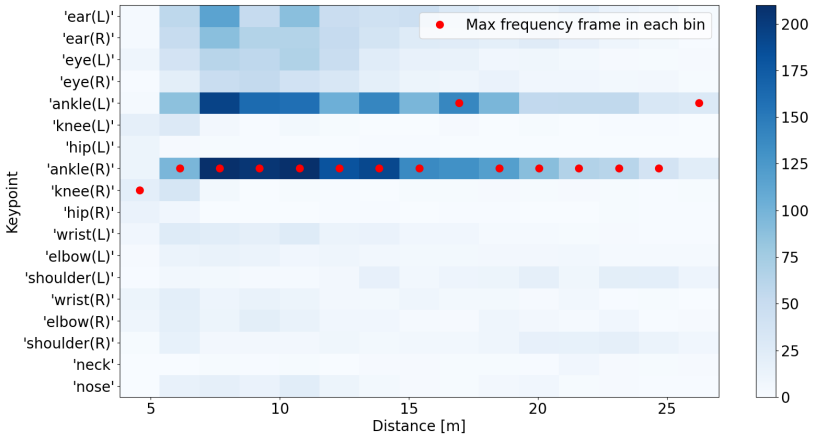


Figure 4.14: Distribution of pedestrian pose keypoints with the lowest attention across pedestrian distances from the camera, ranging from 3.8 m to 27 m and divided into 15 equidistant bins. The shade of blue in each frame represents the frequency of the keypoint receiving the least attention relative to others in the interval. Additionally, a red dot marks the dominant keypoint for each distance interval.

4.2.3 Interpretation of Results

Based on the obtained data and qualitative analysis of the visualized explanations for individual detections, as demonstrated in Figures 4.15 and 4.16, it is reasonable to propose the following theory about the explanations of the detector's decisions and the detections themselves.

The proposed metrics suggest that the explanations for detections across various distances remain largely consistent, indicating that the detector focuses on all of the analyzed body parts, with an overall preference for the 'lower body'. More specifically, there is a notable emphasis on the 'hips' body part, as demonstrated, for example, in Figures 4.11 and 4.13. This observation aligns with the visualizations of general cases, as depicted in Figures 4.6 and 4.5.

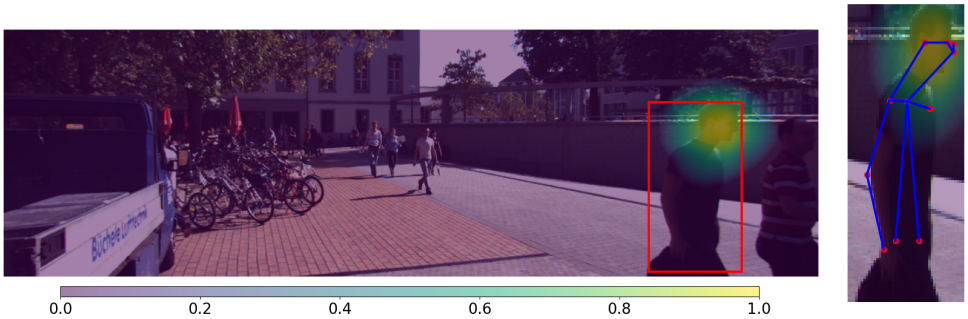
The only uncovered inconsistency arises when the detected pedestrian's distance from the camera is approximately within 6 meters. In such cases, the detector's attention is more likely to shift from the 'lower body' to the 'upper body', specifically focusing on the 'head', 'neck', and 'shoulders'. These dependencies in the metrics are also evident when visualizing such cases, as shown, for example, in Figures 4.15 and 4.16.

Many of the pedestrians corresponding to the aforementioned cases are partially truncated, despite being labeled in the dataset as not truncated, particularly in the section of their legs. This is also evident in Figure 3.6, which illustrates that the first interval's poses exhibit a higher incidence of missing 'ankle(L)' and 'ankle(R)' keypoints compared to other distance intervals, where the ankle keypoints are minimally absent.

It is not surprising that the detector doesn't focus on the ankles of a pedestrian when they are out of the frame. However, what is more interesting is that it appears that the detector, along with the truncated ankles or knees, ceases to focus on the usually targeted hips and transfers attention to the neck and head of the pedestrian.

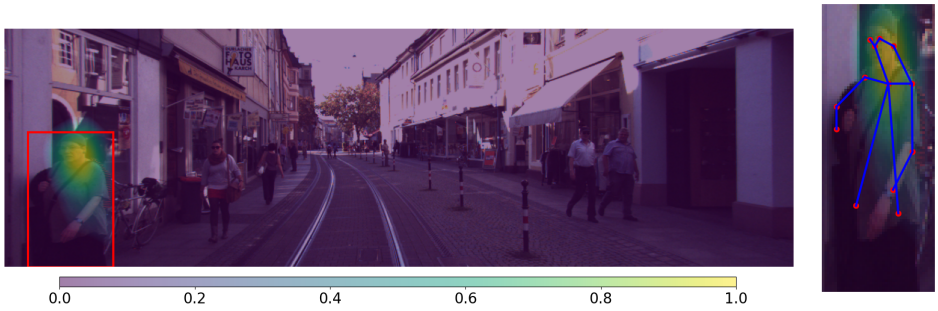
The main indicators supporting this theory are the information about the highest attention keypoint histogram in Figure 4.13, along with the decrease in the first intervals of the graphs visualizing the BPA metrics for the 'lower body' group, and the increase in the 'upper body' graphs. This trend is most evident in Figure 4.9, where the drop in BPA-Sum metric values for the 'lower body' is accompanied by visualizations of several such cases.

When considering these outcomes, it is crucial to keep in mind that correlation does not imply causation. It is very likely that the aforementioned truncation is not the sole factor causing the observed transfer of the model's attention. There are cases where pedestrians have parts of their legs out of the image, yet the attention remains around the typical 'hips' body part, as seen in, for example, Figure 4.16(c). In contrast, there are instances where pedestrians do not have truncated legs, or only to a minor extent, yet the model focuses on the upper part of the body, as seen, for example, in Figure 4.16 (a).



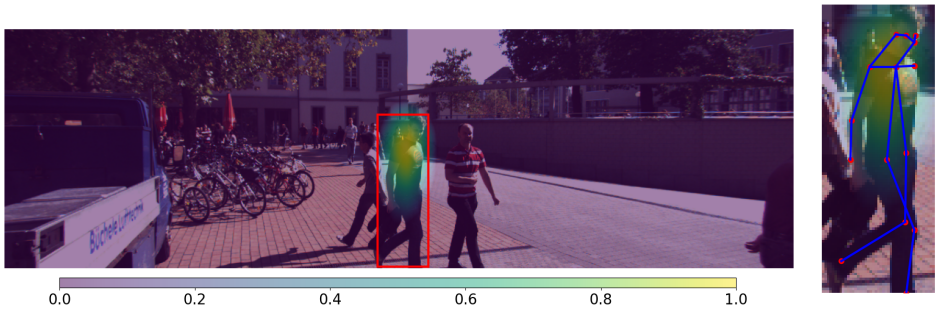
(a) : BPA-Sum: 0.0, [Sequence: 17, Image: 62, Pedestrian ID: 3] (1)

(b) : BPA-Sum: 3.5, [Sequence: 19, Image: 304, Pedestrian ID: 27] (2)

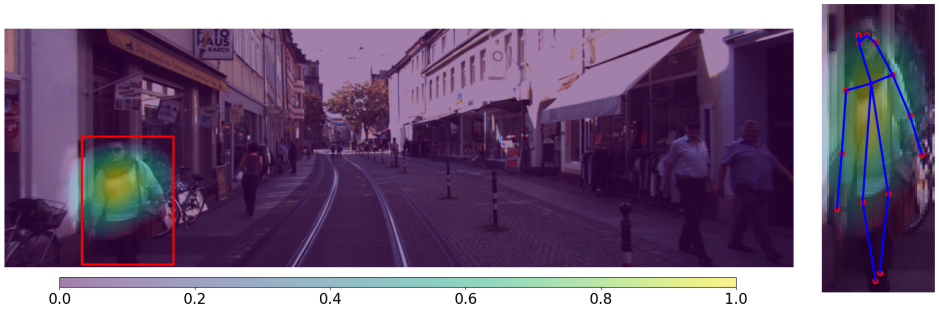


(c) : BPA-Sum: 164.6, [Sequence: 13, Image: 266, Pedestrian ID: 44] (3)

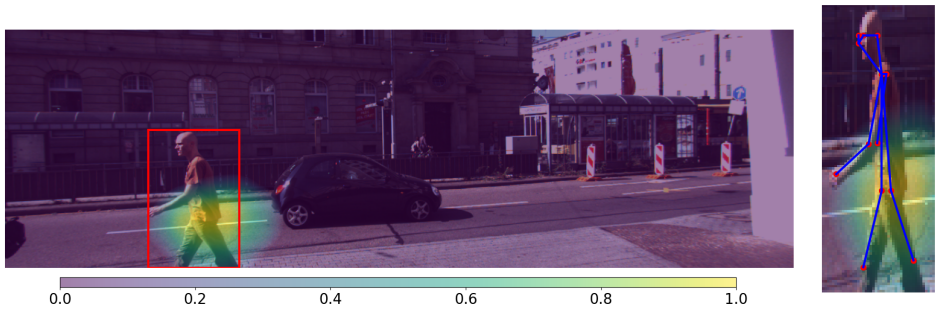
Figure 4.15: Three images displaying a target pedestrian with red bounding boxes predicted by the object detector, overlaid with pixel attribution maps. Cropped images on the right show the pedestrian pose visualized within the bounding boxes. Each image's caption contains the BPA-Sum metric value calculated for the specific case, with dataset specifications provided in square brackets and reference numbers from Figure 4.10 enclosed in parentheses.



(a) : BPA-Sum: 500.7, [Sequence: 17, Image: 43, Pedestrian ID: 3] (4)



(b) : BPA-Sum: 699.4, [Sequence: 13, Image: 270, Pedestrian ID: 65] (5)



(c) : BPA-Sum: 1107.1, [Sequence: 19, Image: 982, Pedestrian ID: 84] (6)

Figure 4.16: Three images displaying a target pedestrian with red bounding boxes predicted by the object detector, overlaid with pixel attribution maps. Cropped images on the right show the pedestrian pose visualized within the bounding boxes. Each image's caption contains the BPA-Sum metric value calculated for the specific case, with dataset specifications provided in square brackets and reference numbers from Figure 4.10 enclosed in parentheses.

Chapter 5

Conclusion

The primary objectives of this thesis outlined in section 1.2 were largely achieved, with the last optional step remaining incomplete.

Initially, the thesis explored historical and modern object detection methodologies, focusing on deep-learning techniques such as two-stage and one-stage detectors.

Subsequently, attention turned to explainable AI methods, specifically examining pixel attribution techniques such as gradient-based Grad-CAM, Vanilla Gradient, and the ablation-based D-RISE method.

From the multitude of available autonomous driving datasets, the KITTI Multi-Object Tracking (KMOT) dataset was chosen for its comprehensive labeling and relevance to the thesis.

Following thorough research on detectors, a two-stage Faster R-CNN model was selected, leveraging its 'person' class to represent pedestrians, the primary objects of interest in the subsequent analysis.

From the proposed XAI methods, the D-RISE method was selected due to relatively straightforward interpretation of its attention maps.

Following the implementation of the main algorithm, which utilized the previously chosen components, input data for analysis was generated.

Before proceeding with the actual analysis, it was crucial to establish appropriate metrics. Thus, the Body Part Attention (BPA) metrics, based on the pose of analyzed pedestrians, were introduced. These poses were estimated using the well-known OpenPose detector and divided into several body parts such as 'head', 'lower body', or 'upper body'.

Three BPA metrics were introduced:

- BPA-Sum, representing the total attention around specific body parts, calculated as the sum of the pixels of the pixel attribution map around the body part's keypoints.
- BPA-Average, representing the average attention value (ranging from 0 to 1) around the body part.
- Relative-BPA, which corresponds to the attention for the body part relative to the overall amount of attention in the pixel attribution map.

The analysis of the D-RISE explanations of the Faster R-CNN model's detections of pedestrians in the images of the KMOT dataset was conducted using two approaches.

The frame-based analysis representing the assigned time-based approach, which, although employed, was deemed insufficient. The concept of time analysis was deemed unreasonable, as detectors base decisions on individual static images. These decisions are influenced by the current state of the object rather than by a sequence of states leading up to or following it.

Conversely, the distance-based analysis focused on one component of the pedestrians' state: their distance from the camera. This analysis revealed that while explanations and consequently detections remained consistent across the entire analyzed range of distances (3.8 meters to 27 meters) – with the model focusing on the whole body of the pedestrian, particularly emphasizing the middle part, specifically the hips – pedestrians within 6 meters, especially those with parts of their legs (e.g., ankles) out of the image, were more likely to be recognized by the upper part of their bodies, such as heads and shoulders.

Bibliography

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, p. 1137–1149, June 2017. doi: 10.1109/tpami.2016.2577031.

[2] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, p. 273–297, Sept. 1995. doi: 10.1007/bf00994018.

[3] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, CVPR-01, IEEE Comput. Soc. doi: 10.1109/cvpr.2001.990517.

[4] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, IEEE. doi: 10.1109/cvpr.2005.177.

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, June 2014. doi: 10.1109/cvpr.2014.81.

[6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, June 2016. doi: 10.1109/cvpr.2016.91.

[7] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *International Journal of Computer Vision*, vol. 104, p. 154–171, Apr. 2013. doi: 10.1007/s11263-013-0620-5.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, p. 1904–1916, Sept. 2015. doi: 10.1109/tpami.2015.2389824.

[9] R. Girshick, "Fast r-cnn," in *2015 IEEE International Conference on Computer Vision (ICCV)*, IEEE, Dec. 2015. doi: 10.1109/iccv.2015.169.

[10] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask r-cnn," in *2017 IEEE International Conference on Computer Vision (ICCV)*, IEEE, Oct. 2017. doi: 10.1109/iccv.2017.322.

[11] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, July 2017. doi: 10.1109/cvpr.2017.106.

[12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, *SSD: Single Shot MultiBox Detector*, p. 21–37. Springer International Publishing, 2016. doi: 10.1007/978-3-319-46448-0_2.

[13] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," 2017. doi: 10.48550/ARXIV.1708.02002.

[14] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," 2018. doi: 10.48550/ARXIV.1804.02767.

[15] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "Yolov9: Learning what you want to learn using programmable gradient information," 2024. doi: 10.48550/ARXIV.2402.13616.

[16] C. Molnar, *Interpretable Machine Learning*. 2 ed., 2022. sec: 10.2. `https://christophm.github.io/interpretable-ml-book` [Accessed 18-05-2024].

[17] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," 2014. doi: 10.48550/ARXIV.1312.6034.

[18] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "Smoothgrad: removing noise by adding noise," 2017. doi: 10.48550/ARXIV.1706.03825.

[19] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," *International Journal of Computer Vision*, vol. 128, p. 336–359, Oct. 2019. doi: 10.1007/s11263-019-01228-7.

[20] V. Petsiuk, R. Jain, V. Manjunatha, V. I. Morariu, A. Mehra, V. Ordonez, and K. Saenko, "Black-box explanation of object detectors via saliency maps," 2021. doi: 10.48550/ARXIV.2006.03204.

[21] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, June 2012. doi: 10.1109/cvpr.2012.6248074.

[22] P. Xiao, Z. Shao, S. Hao, Z. Zhang, X. Chai, J. Jiao, Z. Li, J. Wu, K. Sun, K. Jiang, Y. Wang, and D. Yang, "Pandaset: Advanced sensor suite dataset for autonomous driving," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, IEEE, Sept. 2021. doi: 10.1109/itsc48978.2021.9565009.

[23] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," 2020. doi: 10.48550/ARXIV.1903.11027.

[24] J. Geyer, Y. Kassahun, M. Mahmudi, X. Ricou, R. Durgesh, A. S. Chung, L. Hauswald, V. H. Pham, M. Mühlegg, S. Dorn, T. Fernandez, M. Jänicke, S. Mirashi, C. Savani, M. Sturm, O. Vorobiov, M. Oelker, S. Garreis, and P. Schuberth, "A2d2: Audi autonomous driving dataset," 2020. doi: 10.48550/ARXIV.2004.06320.

[25] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2014. doi: 10.48550/ARXIV.1405.0312.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, June 2016. doi: 10.1109/cvpr.2016.90.

[27] "Torchvision main documentation — pytorch.org - fasterrcnn_resnet50_fpn 2014." `https://pytorch.org/vision/main/models/generated/torchvision.models.detection.fasterrcnn_resnet50_fpn.html`, [Accessed 18-05-2024].

[28] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, p. 172–186, Jan. 2021. doi: 10.1109/tpami.2019.2929257.