electronics

MDPI

Article

Meta-YOLOv8: Meta-Learning-Enhanced YOLOv8 for Precise Traffic Light Color Detection in ADAS

Vasu Tammisetti [1,2,*], Georg Stettinger [1], Manuel Pegalajar Cuellar [2] and Miguel Molina-Solana [2]

[1] Infineon Technologies AG, 85579 Munich, Germany; georg.stettinger@infineon.com
[2] Department of Computer Science and Artificial Intelligence, University of Granada, 18071 Granada, Spain; manupc@decsai.ugr.es (M.P.C.); miguelmolina@ugr.es (M.M.-S.)
* Correspondence: vasu.tammisetti@infineon.com

Abstract: The ability to accurately detect traffic light color is critical for the functioning of Advanced Driver Assistance Systems (ADAS), as it directly impacts a vehicle's safety and operational efficiency. This paper introduces Meta-YOLOv8, an improvement over YOLOv8 based on meta-learning, designed explicitly for traffic light color detection focusing on color recognition. In contrast to conventional models, Meta-YOLOv8 focuses on the illuminated portion of traffic signals, enhancing accuracy and extending the detection range in challenging conditions. Furthermore, this approach reduces the computational load by filtering out irrelevant data. An innovative labeling technique has been implemented to address real-time weather-related detection issues, although other bright objects may occasionally confound it. Our model employs meta-learning principles to mitigate confusion and boost confidence in detections. Leveraging task similarity and prior knowledge enhances detection performance across diverse lighting and weather conditions. Meta-learning also reduces the necessity for extensive datasets while maintaining consistent performance and adaptability to novel categories. The optimized feature weighting for precise color differentiation, coupled with reduced latency and computational demands, enables a faster response from the driver and reduces the risk of accidents. This represents a significant advancement for resource-constrained ADAS. A comparative assessment of Meta-YOLOv8 with traditional models, including SSD, Faster R-CNN, and Detection Transformers (DETR), reveals that it outperforms these models, achieving an F1 score, accuracy of 93% and a precision rate of 97%.

Keywords: meta-learning; meta-YOLO; YOLO; labeling; convolutional neural networks (CNN); optimization; advanced driver assistance system (ADAS); autonomous vehicle (AV); task similarity

Academic Editor: Manohar Das

Received: 8 November 2024
Revised: 8 January 2025
Accepted: 12 January 2025
Published: 24 January 2025

Citation: Tammisetti, V.; Stettinger, G.; Cuellar, M.P.; Molina-Solana, M. Meta-YOLOv8: Meta-Learning-Enhanced YOLOv8 for Precise Traffic Light Color Detection in ADAS. *Electronics* **2025**, *14*, 468. https://doi.org/10.3390/electronics14030468

1. Introduction

Traffic lights are essential for road safety and traffic management, as they regulate the movement of vehicles and pedestrians at intersections. Consequently, accurate detection of traffic lights and their colors is critical for any Advanced Driver Assistance System (ADAS), as it directly impacts the system's decision-making processes, as well as vehicle safety and efficiency. To only name a few, this capability helps prevent accidents, enhances passenger safety, improves traffic flow, and reduces congestion. By accurately recognizing and interpreting traffic light signals, ADAS become more reliable and effective, increasing public trust and accelerating the adoption of these technologies in vehicles.

In summary, the following are key considerations when discussing the importance of traffic light detection [1]:

- Safety: Accurate traffic light detection for ADAS is critical to ensuring the safety of passengers, pedestrians, and other vehicles. For an autonomous vehicle (AV), the detection of stop and proceed lights is essential to obey traffic laws and avoid accidents. This task is particularly challenging due to the varying weather and lighting conditions that require precise identification of traffic light colors.
- ADAS real-time decision making and navigation: The ability of ADAS to recognize and interpret traffic lights in real time is of paramount importance for the prompt decision-making processes that are necessary for adjusting speed and navigating around detours, thus ensuring the smooth and predictable operation of the vehicle.
- Traffic flow optimization: Data obtained from the detection of traffic lights can be seamlessly integrated into smart city infrastructure to optimize traffic flow. This data plays a crucial role in developing adaptive traffic light control systems, which are of essential for reducing road congestion and enhancing traffic efficiency.
- Human–machine interface (HMI) improvement: The implementation of traffic light detection technology is key for enhancing the human–machine interface (HMI) in semi-autonomous vehicles. By providing drivers with accurate, timely information about road conditions, it promotes safer and more efficient driving. This decision-making support not only improves safety but also reduces cognitive load on drivers, ensuring a more comfortable and efficient driving experience.
- Autonomy under diverse conditions: The capacity to detect and interpret traffic lights in a variety of environmental contexts represents a key indicator of the level of autonomy that an ADAS can achieve [2].

Color detection is a fundamental aspect of traffic communication and driver decision-making [3]. Consequently, the recognition of Traffic Light Color (TLC) is a key component of ADAS, serving as a critical input for navigation and ensuring the safety of both vehicle occupants and other road users. However, factors such as varying lighting conditions, adverse weather, occlusion, and traffic light deterioration can compromise color perception, affecting vehicle systems. In regions where color is the primary traffic rule indicator, inaccurate color detection may lead to traffic law violations or accidents [4].

The need for precise color detection is intensified by the accelerated advancement of ADAS technologies. Unlike human drivers, who can frequently ascertain the appropriate light from incomplete information or context, ADAS systems rely on accurate data inputs to ensure safe and efficient operation. The absence of precise color detection could lead to erroneous conclusions by on-board AI systems, posing potential safety risks [2]. However, technologies like radar and LIDAR are ineffective for traffic light color recognition as they can only detect distance and shapes, not color [5].

Existing studies on traffic light detection for ADAS have made progress but face challenges such as large data dependency, varied weather and lighting conditions, real-time processing needs, adaptability, and integration with ADAS. These systems often lack robustness in diverse environments and can increase driver cognitive load. This work introduces advanced meta-learning and innovative labeling to enhance detection accuracy and resilience, improve real-time decision making, and seamlessly integrate with traffic management systems. These advancements aim to significantly boost safety, efficiency, and public trust in ADAS technologies, providing a substantial contribution to the field.

Enhanced Adaptability with Meta-Learning in YOLOv8

YOLOv8 [6] is a state-of-the-art, real-time object detection system that predicts bounding boxes and class probabilities from full images in a single evaluation. However, it often struggles with the variability and complexity of objects like traffic lights.

The application of meta-learning [7], which involves learning how to learn, is a crucial enhancement for traffic light color (TLC) detection using YOLOv8, as it adaptability and detection accuracy in real-world scenarios can be significantly improved [8]. Meta-learning leverages accumulated "meta-data" from various tasks to accelerate skill acquisition by utilizing prior knowledge and successful strategies [9]. In TLC detection, meta-learning's adaptability is particularly beneficial as the model can quickly adjust to new tasks with minimal input data, addressing the challenges posed by varying lighting conditions, partial occlusions, and different weather effects.

Another major challenge in the development of object detection models is the gathering of substantial data, particularly for rare categories [10]. Obtaining training images for traffic lights and rare vehicles can be difficult, and labeling large datasets is resource intensive.

To address these issues, we used task similarity through meta-learning to train object detection models on scarce categories. Meta-learning methodologies in YOLOv8 enhance its adaptability, enabling efficient parameter calibration with limited data. The meta-learned YOLOv8 recognizes TLCs with fewer training examples and maintains robust performance in shifting environments, reducing the dependence on extensive labeled data [11]. This research describes Meta-YOLOv8's application to robust and accurate traffic light detection. Leveraging meta-learning principles, this model outperforms traditional models like SSD, YOLOv8, Faster R-CNN, and Detection Transformers, achieving enhanced color fidelity and improved inference tasks under varying conditions. This advancement represents a significant step forward in TLC recognition systems, offering potential for more reliable and secure autonomous transportation.

The rest of this work is organized as follows: Section 2 reviews and summarizes related work. Section 3 introduces Meta-YOLOv8 and Section 4 describes the methodology used in the experimentation. In Section 5, we present and discuss the results. The manuscript ends with a summary of the main conclusions, existing limitations, and future work.

2. Related Work

In order to contextualize our contribution, we first review the most prominent object detection models that have paved the way for the development of advanced traffic light detection systems. These models include the Single Shot Multibox Detector (SSD), different versions of You Only Look Once (YOLO), Faster R-CNN, recent advances in Detection Transformers (DETR), and Tiny YOLO. In particular, models such as YOLOv3 and SSD have been successfully applied to traffic light detection in ADAS [12–14], while Detection Transformers has demonstrated potential for object detection in complex scenes. Each method employs a distinctive strategy to address the complexities inherent to object detection, yet they also face common challenges, including the need for extensive data and the capacity to adapt to evolving environments. A brief description of each model is provided below.

- Single Shot Multibox Detector (SSD): The SSD [14] method is distinguished by its high processing speed, which is achieved through a single-shot approach that obviates the necessity for a separate region proposal network. By employing a set of default bounding boxes and aspect ratios, SSD is able to predict the presence of objects at multiple scales, thereby facilitating the detection of objects of varying sizes within an image. However, it should be noted that SSD may encounter difficulties in detecting very small objects, and extensive data augmentation may be necessary to achieve the desired level of robustness. With regard to the substantial data dependencies inherent

to its operation, the performance of SSD is contingent upon the extent and diversity of the training data employed, which is necessary for the effective discernment of diverse object scales and aspect ratios [15].

- You Only Look Once (YOLOv8): The YOLO family, particularly the developments observed in YOLOv5 and YOLOv8 [16,17], has the capacity for real-time object detection with a high degree of accuracy. These models adopt a comprehensive approach to image processing, simultaneously predicting bounding boxes and class probabilities in a single evaluation. This approach markedly diminishes the requisite inference time, rendering it well suited to applications that necessitate real-time analysis. One of the principal advantages of the more recent iterations, such as YOLOv8, is the enhancement in the ability to recognize small objects and the improvement in generalization across different datasets. These advancements have been made possible by architectural innovations and rigorous training regimes. Nevertheless, it should be noted that YOLO models may still be susceptible to challenges posed by occluded or overlapping objects. Moreover, while these models have reduced their data requirements through enhanced architectures, they continue to benefit considerably from the availability of extensive annotated datasets to optimize their detection capabilities [18].
- Faster R-CNN: [19] is a pioneering model in the region-based convolutional neural network (CNN) family, and offers a distinctive combination of accuracy and comprehensiveness. A region proposal network (RPN) is employed to hypothesize object locations, with these predictions then refined by a Fast R-CNN detector. Although this two-stage process is more computationally intensive, it offers high precision and recall rates, which are particularly useful in scenarios where accuracy is critical. One limitation of Faster R-CNN is its relatively slow processing speed, which makes it less suitable for real-time detection tasks. Furthermore, the model requires substantial data inputs to effectively train both the RPN and the detector, making it a data-intensive model [20].
- Detection Transformers (DETR): introduced an end-to-end object detection framework that employs Transformers [13], an architectural approach that has demonstrated considerable success in the field of natural language processing. DETR circumvents the need for numerous manually designed components by learning to perform object detection as a direct set prediction problem. While it benefits from Transformers' capacity to attend to global contexts within an image, DETR typically necessitates longer training periods and larger datasets to achieve optimal performance levels. Furthermore, DETR encounters difficulties in the detection of small objects due to the global nature of attention mechanisms. Nevertheless, it provides a promising avenue for adaptability due to its flexible architecture that is not constrained by preset anchor boxes or proposals [21].
- Tiny YOLOv4: [22] is a streamlined version of the YOLO object detection model. It has been designed to be faster and more efficient, particularly on edge devices with limited computational resources. The model maintains an optimal balance between speed and accuracy by employing a reduced number of layers and parameters in comparison to the full YOLOv4 model. Tiny YOLOv4 is particularly effective for applications requiring real-time processing, such as traffic light color detection, where it can quickly identify and classify objects with relatively low latency. However, it may not consistently attain the same degree of accuracy than more sophisticated models can achieve by employing advanced architectures and learning strategies to enhance detection performance, particularly for smaller and densely packed objects.

Each one of the models above represents a discrete approach to the shared objective of object detection. The efficacy of these models is contingent upon the availability of extensive and heterogeneous datasets, as well as their capacity to adapt to novel and unforeseen conditions. As the field of object detection continues to evolve, achieving the optimal balance between data requirements and adaptability will remain a central theme. This balance is decisive to the development of more sophisticated and efficient detection algorithms.

3. Our Proposal

Considering the factors above, this manuscript introduces the Meta-YOLOv8 model with the aim of addressing some of the enduring limitations of the aforementioned models. The application of meta-learning principles (learn to learn [23]) enables the Meta-YOLOv8 model to learn color features from a relatively limited amount of data, whereas conventional models and methods are unable to demonstrate greater efficacy in the learning process and suffer from catastrophic forgetting when exposed to new classes. This also helps significantly reducing the data dependency, which is a major limitation of conventional models [24]. Our model is particularly advantageous in the context of traffic light detection, where the specific characteristics of traffic lights may vary across different countries and new traffic lights may be introduced. The Meta-YOLOv8 model exhibits exceptional adaptability to new detection scenarios and environmental variations without the need for extensive retraining while handling the catastrophic forgetting problem [25]. It can identify new traffic light categories while maintaining real-time inference capabilities and offers competitive training and inference times, making it ideal for practical, resource-constrained deployments where rapid response times are critical [26].

Furthermore, systems such as Meta-YOLOv8 enhance their detection capabilities over time through continuous acquisition of new data, effectively addressing the dynamic nature of traffic environments [26].

3.1. Meta-YOLOV8 Architecture

Our proposed Meta-YOLOv8 architecture leverages the latest advancements in YOLOv8, which excels in object detection, image classification, and instance segmentation tasks while significantly improving real-time inference speed. The new approch in our method lies in employing both the base model and its clone within an MAML-based meta-learning framework [7]. This dual-network approach, its with knowledge sharing mechanism, not only enhances the model's adaptability to unseen data but also reduces its reliance on large datasets during training and validation, distinguishing it from conventional single-network training methods.

YOLOv8 comprises three primary components (as depicted in Figure 1): the backbone, head, and neck. This architecture includes several improved elements, such as the C2F block, the SPPF block, and an additional bottleneck. These components will be elaborated upon in the following sections.

The core functionality of YOLOv8 is the extraction of salient features from an image, followed by a reduction in spatial dimensions. The neck component combines these extracted features across varying spatial ratios and dimensions. The head component then classifies and localizes the target object within the image. The backbone of the model is made up of three distinct blocks: a convolution block, a C2F (Cross-Stage Partial Bottleneck with 2 Convolutions) block, and a spatial pyramid pooling feature block [27].

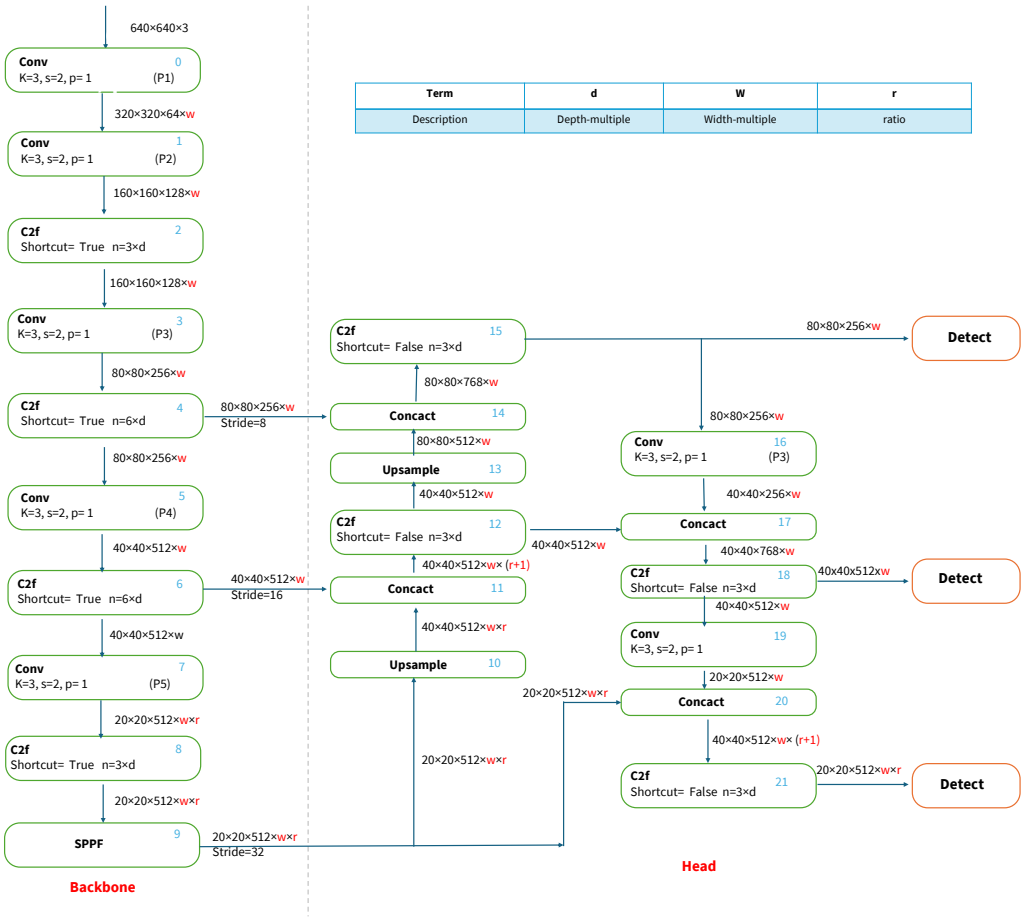

Figure 1. Meta-YOLOV8 architecture; inspired from Roboflow image (accessed on 28 October 2024): https://blog.roboflow.com/what-is-yolov8/.

3.1.1. CBS (Convolutions, Batch Normalization, and Pooling)

In the context of machine learning, the term "CBS" is used to refer to a specific set of techniques, namely, those involving ''convolutions", "batch normalization", and "SiLU", which stands for "Sigmoid-weighted linear units". We employed a $3 * 3$ convolution, followed by batch normalization and SiLU.

3.1.2. Batch Normalization

Batch normalization is the process of normalizing the activation values resulting from convolution. This entails the calculation of averages and standard deviations across the batch, with the objective of stabilizing the distribution of activations.

3.1.3. SiLU (Sigmoid Linear Unit)

After convolution and, optionally, batch normalization, the SiLU activation function is applied to the output. The C2F block (Figure 2) is a convolutional block with a bottleneck that commences with a $1 * 1$ convolution with a single stride and no padding. Subsequently, the number of channels is reduced by half and conveyed through the bottleneck. The bottleneck is present in both the double-convolution block with and without a shortcut. In the event that the shortcut is set to "true", a skip connection is incorporated into the output. Subsequently, the output is concatenated and passed through another convolutional layer.

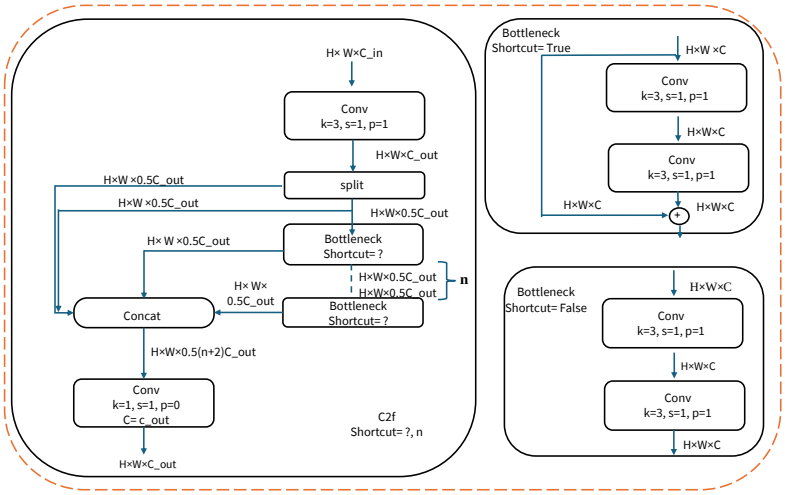

Figure 2. Cross-Stage Partial Bottleneck with 2 Convolutions (C2F).

3.1.4. Spatial Pyramid Pooling Fast (SPPF)

The extension of spatial pyramid pooling (SPP) is referred to as "spatial pyramid pooling fast". The SPPF architecture comprises a convolutional layer followed by three max-pooling layers, as illustrated in Figure 3. The noteworthy aspect of this process is that the output of each layer is concatenated and subsequently conveyed to the final convolution layer ((3 June 2023) https://github.com/ultralytics/ultralytics/issues/189 [27]).

The fundamental concept underlying spatial pyramid pooling (SPP) is the partitioning of the input image into a grid, with the objective of independently pooling features from each grid cell. This approach allows the network to effectively process images of varying sizes. Essentially, SPP enables neural networks to handle images of different resolutions by capturing multi-scale information through pooling operations at various levels of granularity. This capability is particularly advantageous in tasks such as object recognition, where objects may appear at different scales within an image.

Although spatial pyramid pooling offers numerous advantages, it is also relatively computationally expensive. To address this issue, SPP-Fast employs a simplified pooling methodology. Instead of using multiple pooling levels with varying kernel sizes, SPP-Fast utilizes a single fixed-size kernel for pooling, thus reducing computational requirements. This approach offers a compromise between accuracy and speed.

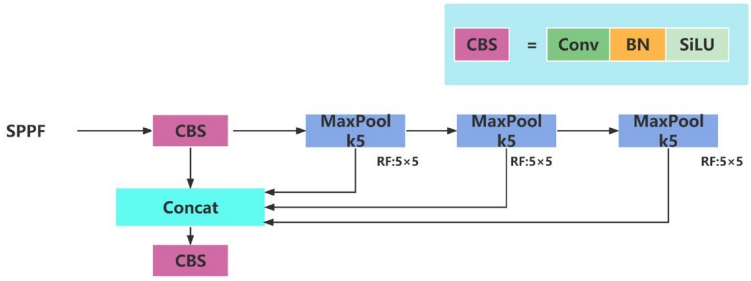

Figure 3. Spatial pyramid pooling fast block diagram.

3.1.5. Detection Block

The detection block in YOLOv8 is responsible for identifying objects within images. In contrast to preceding versions (during the period of our project), YOLOv8 is an anchor-free model, whereby the center of an object is predicted directly, as opposed to utilizing an offset from a known anchor box. This approach facilitates a more expeditious and efficacious prediction process. The detection block encompasses two tracks: one for bounding

box predictions and the other for class predictions. Each track comprises two convolutional blocks followed by a single Conv2d layer, as illustrated in Figure 4; these generate the bounding box loss and class loss, respectively [27].

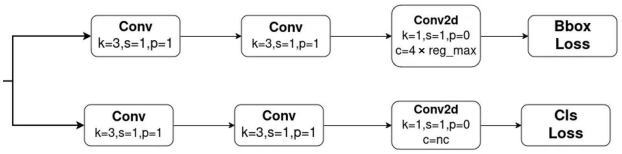

Figure 4. YOLOv8 detection block: Anchor-free model predicting object centers with separate tracks for bounding box and class.

3.2. Meta-Learner

In the initial stage of the learning process, the outer loop weights are iteratively updated, guided by the loss function, which measures the model's performance against a known output [28]. This progression can often be visualized, as depicted in Figure 5, and shows the optimization path of the loss function in relation to the model's weights.

Next, a meta-learner (inner loop) with adaptability is introduced. This component takes the previously generalized weights—now optimized and more closely aligned with the requirements of our specific task—and refines them within a more narrowly tailored loss and weight landscape, as shown in Figure 5b.

Finally, when data specific to the particular task are applied, the learner updates the model with a new set of weights. These latest updates are precisely targeted to detect our object of interest, reflecting the culmination of the learning process, where the model has acquired the necessary specificity and accuracy for successful object detection. The mathematical representation of weight updates is explained below.

The meta-learner employs second-order computations (see Equation (1)) to learn across tasks taken from the same distribution. The system utilizes a blend of two-stage optimization: the first stage focuses on learning task similarity (outer loop), and the second stage corresponds to task-specific learning. These stages are intended to improve overall proficiency [7,29].

$$\theta^{'} = argmin \frac{1}{M} \sum_{i=1}^{M} L(in(\theta, D_i^{tr}) D_i^{test}) \tag{1}$$

The individual terms in Equation (1) are defined as follows: M represents the number of tasks in the group, while D_i^{tr} and D_i^{test} denote the i^{th} task in the training and test sets, respectively. The function L represents the task loss, and the data in D_i^{tr} is used for inner loop training. For each task in a batch, the neural network is initialized with θ. This initial value is then optimized in the head of Meta-YOLOv8 through one or a few gradient descent steps on the training set D_i^{tr} to obtain fine-tuned task parameters Θ_i. Considering only one phase of training in the detector, the assignment parameters are equated to [7]

$$\Theta_i \simeq in(\theta^{'}, D_i^{tr} = \theta^{'} - \alpha \nabla_{\theta'} L(\theta^{'}, D_i^{tr})) \tag{2}$$

This process involves updating the metaparameters θ from Equation (1) based on the average loss of the fine-tuned parameters for each task, Θ_i, as shown in Equation (2), using the test set D_i^{test}. Consequently, after fine-tuning, Meta-YOLOv8 optimizes the loss more effectively, outperforming simple pre-training, as described earlier. Various adaptations contribute to increased learning speed and efficiency, as well as improved handling of new tasks and task distributions. A more detailed explanation and interactive analysis of some variations can be found in [7,28].

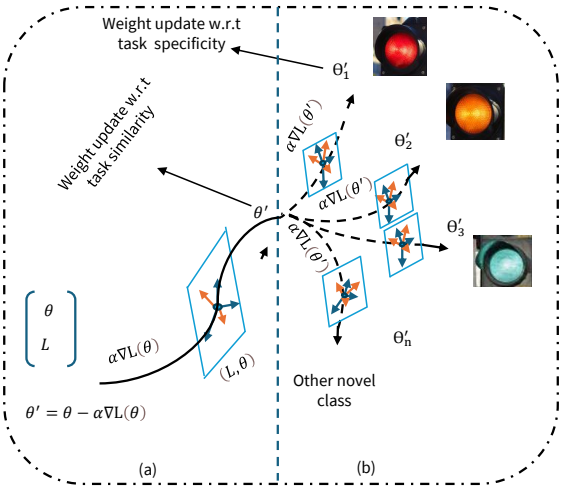

Figure 5. The base model (**a**) for TLC is initialized with random weights θ and trained on similar tasks to prime it for final task performance, with its learning trajectory guided by a predefined loss function and iterative weight updates to $\theta^{'}$. A meta-learner (**b**) further refines these weights to a set of values $\Theta^{'}$, aligning them with the specific task's requirements, until the model is fine-tuned with task-specific data, resulting in a tailored set of weights ($\Theta^{'}_{i}$) optimized for each class detection.

4. Methods

In this section, we describe the data we used for the experiments and how we preprocessed and annotated them. We also describe the particular architecture being used and the metrics we register in the experiments.

4.1. Data

In the absence of specific public datasets tailored to the advanced requirements of our object detection model, which include high-quality labeled images covering various lighting conditions, angles, and weather scenarios, we constructed a bespoke fusion traffic dataset using different public datasets like Kitty: https://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=2d (accessed on 10 July 2023); Kaggle: https://www.kaggle.com/datasets/wjybuqi/traffic-light-detection-dataset?resource=download (accessed on 10 July 2023); CARLA: https://www.kaggle.com/datasets/sachsene/carla-traffic-lights-images (accessed on 10 July 2023); LISA: https://www.kaggle.com/datasets/mbornoe/lisa-traffic-light-dataset/code (accessed on 10 July 2023); CityScapes: https://www.kaggle.com/datasets/shuvoalok/cityscapes (accessed on 1 December 2023); and Eurocity: https://eurocity-dataset.tudelft.nl/eval/user/login?_next=/eval/downloads/detection (accessed on 15 January 2024). Table 1 provides a summary of the used datasets showcasing the varying degrees of feature significance and data integrity. Our criteria for image selection were multifaceted, ensuring a comprehensive representation of real-world driving scenarios.

Firstly, we prioritized image quality, focusing on pixel density, object clarity, and aspect ratio, and also considered the other complex traffic scenarios in the case of faint light or glare light. Consistency in these parameters was vital for maintaining the integrity of the input data. We selected images with varied aspect ratios to mimic the diverse visual inputs encountered by drivers and autonomous driving systems. Additionally, color clarity was a decisive factor, as it directly affects the model's ability to discern and accurately classify traffic lights under various lighting and weather conditions.

Table 1. Diverse datasets showing varying degrees of feature significance and data integrity.

No.	Dataset	Important Features	Quality/Uncertainty in Data
1	Kitty	Long distance and edges of traffic lights	90%/10%
2	Kaggle	Long distance and edges of traffic signal lights	75%/30%
3	Carla Traffic Light Images	Colors of traffic signals in different weather conditions	85%/20%
4	LISA Traffic Light Dataset	Long-distance view and edges of traffic signals	80%/20%
5	Cityscapes	Traffic signals in different weather condition	85%/15%
6	Eurocity	Color and contrast of traffic signals	90%/15%

We also emphasized selecting images taken from distances similar to a driver's viewpoint. This ensures that the model is trained on data representative of a driver's perspective, enhancing its practical applicability. Moreover, we included images with pronounced edge features to aid object detection algorithms in recognizing the contours and boundaries of traffic lights amid cluttered backgrounds.

Our dataset includes images from complex environments, such as road junctions, where detection patterns are intricate due to multiple traffic lights, signals, and varying vehicular movements. This challenges the model with the high levels of complexity and ambiguity found in dense traffic scenarios, thereby bolstering its robustness and adaptability.

Through the creation of this fusion traffic dataset, we have curated a collection of images that not only captures a wide array of traffic light characteristics but also encapsulates the complexity of real-world driving conditions. The diversity and quality of the dataset are expected to significantly enhance the generalization capability of the object detection model, enabling it to perform with high reliability and accuracy across diverse operational contexts.

4.2. Data Preprocessing

The process of preparing data for a traffic light detection model is important to ensure its efficiency and precision. We followed the usual preprocessing steps [30,31]:

- Data cleaning: Corrupted and irrelevant images (such as those that were blurred, improperly exposed, or did not contain any traffic lights) were removed.
- Image resizing: To maintain consistency with the training model and to reduce computational load, images were resized to a standard dimension while preserving their aspect ratio. This uniformity is necessary for batch processing during model training.
- Normalization: Pixel values in the images were normalized to have a mean of zero and a standard deviation of one. This step is critical for helping the model's convergence during training and improving its generalization abilities.
- Augmentation: Techniques such as random rotations, flipping, scaling, and cropping were applied to artificially expand the dataset (for some images only). This not only helps in preventing overfitting but also ensures the model is invariant to common variations in the real world.

- Color space conversion: Considering the importance of color in traffic light detection, images were converted into different color spaces such as HSV (hue, saturation, value) or LAB, which might be more effective in highlighting traffic lights under various lighting conditions.
- Contrast adjustment: Histogram equalization was used on the images to enhance contrast, ensuring that traffic lights were distinguishable even under sub-optimal lighting conditions.
- Noise reduction: To improve image quality, noise reduction techniques such as Gaussian blurring or median filtering were utilized to smooth out the images, reducing the impact of sensor noise or compression artifacts.
- Edge enhancement: Edge detection filters (e.g., Sobel, Canny) were applied to some images to accentuate the borders of traffic lights, which can aid the model in identifying these objects against complex backgrounds.
- Region of interest (ROI) extraction: In some cases, ROIs were defined to focus the model's attention on specific areas where traffic lights are likely to be found, thereby reducing the computational complexity, and improving detection performance.
- Data splitting: The dataset was randomly split into training, validation, and testing sets. This ensures that there is no data leakage, and the model's performance can be accurately evaluated.
- Balance classes: To prevent model bias towards over-represented classes, techniques such as over-sampling the minority class or under-sampling the majority class were applied to balance the dataset.

These preprocessing steps were designed to address specific challenges inherent in traffic light detection, such as varying lighting conditions, diverse environmental settings, and the need for high-fidelity light recognition [31].

4.3. Labeling Methods

Bounding boxes (ground truth) are integral to the functionality of deep neural networks in object detection, particularly in YOLO, as they provide decisive spatial localization information. This enhances the network's ability to accurately identify and outline object boundaries within an image. Bounding boxes play a pivotal role in tasks such as pinpointing object locations for autonomous driving or isolating objects in complex scenes due to their localization capabilities. As annotations in training data, they furnish the network with labeled instances of object placement, facilitating the association of visual features with spatial coordinates and object identification.

During the training phase, bounding boxes are instrumental in defining the loss function. The network refines its predictions to match ground truth annotations, which is necessary for honing localization precision. Additionally, bounding boxes come with an "objectness score" that indicates the probability of an object's presence within the box, helping the network to differentiate between relevant objects and background noise. This distinction is particularly beneficial for accurately detecting objects.

In scenarios with multiple objects, bounding boxes enable the simultaneous detection and localization of various items by allocating unique spatial regions to each one. Techniques such as non-maximum suppression further refine this process by removing redundant and overlapping boxes, thereby improving detection accuracy. Furthermore, bounding boxes create an interpretable output that visually demonstrates the detected objects' locations and extents, which is vital for downstream tasks requiring precise localization. They are also employed in region-of-interest (ROI) pooling, which extracts uniform feature maps from designated regions to ensure the network focuses on pertinent object areas during feature extraction and classification.

Lastly, in meta-learning or learn-to-learn scenarios, bounding box annotations in labeled datasets allow pre-trained object detection models to be fine-tuned with new data, enhancing performance on specialized detection tasks. The multifaceted role of bounding boxes is foundational not only for training and optimization but also for the practical application and interpretability of deep learning in object detection.

In the field of object recognition within computer vision, accurate identification and localization of objects are critical for developing reliable models. One illustrative case is TLC recognition, which challenges the effectiveness of traditional labeling and annotation techniques. Typically, as shown in Figure 6, the entire housing of a traffic light is labeled as a single entity. Although this method effectively distinguishes traffic lights from other luminous objects, such as street lamps or road surface reflections, it has its limitations [32].

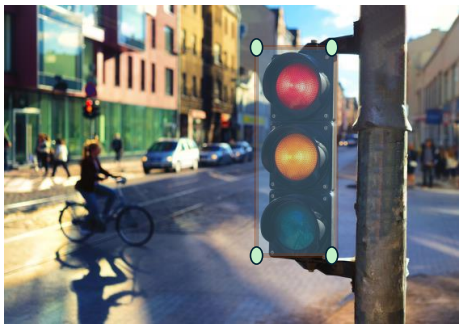

Figure 6. Conventional labeling, where bounding box covers entire traffic light, in which 1/3 of object area does not impact the learning process.

The primary challenge of this approach is its computational complexity, which arises during both the training and inference phases due to the processing of a large amount of irrelevant data. Conventional labeling methods require models to recognize both the color components and the contextual environment of traffic lights. However, in adverse weather and night conditions, surrounding features are often obscured, leaving the model to rely solely on the color components, as illustrated in Figure 7. Consequently, traditional models frequently fail to accurately detect TLCs in low-visibility conditions.

(a) (b)

Figure 7. Traffic lights in adverse weather conditions: (**a**) foggy and (**b**) rainy. Illustrating how the lights appear without surrounding features.

To address the problem of accurately detecting TLCs, we developed an enhanced labeling method compatible with meta-learning, which primarily focuses on the color components of traffic lights. This targeted labeling approach ensures that the model relies on the colored, illuminated regions rather than extraneous features surrounding them. By minimizing the model's dependence on these surrounding features, as seen in Figure 8, we improve its robustness and effectiveness in challenging weather conditions. The core task is to detect the color of the illuminated traffic light, which typically involves one

or two lights within the light box under normal conditions. This method prevents the model from unnecessarily processing the entire light box when only a fraction of that space contains the relevant information.

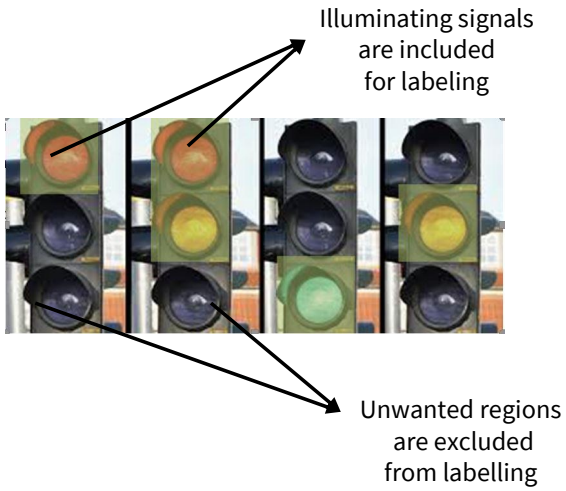

Figure 8. Targeted labeling, which mainly focuses on the illuminating regions which can have the highest impact on learning.

The impact of inefficiency in the annotation process is significant. By refining the annotation to exclude irrelevant parts of the traffic light box, a substantial number of computations can be avoided, potentially speeding up the inference process. For instance, ignoring even one-third of the unnecessary traffic light box during inference could save thousands of computations, thereby increasing recognition speed [33]. To optimize this process, precise annotation of individual colors within the traffic light box using tight bounding boxes is essential. Automated labeling algorithms often fall short of our project targets, necessitating manual labeling. A total of 315 images were manually labeled using LabelImg and divided into an 80:20 ratio for training and validation. Additionally, 20 images were allocated for few-shot training and testing (2-way 8-shot) to evaluate the model's adaptability. We carefully considered various scenarios within the 315 training images, deliberately excluding two specific conditions: rain and fog. This exclusion was to prevent data leakage and to rigorously assess the model's adaptability under diverse circumstances.

Typically, dataset sizes in standard model development are 8 to 10 times larger than our dataset. Meticulous data annotation ensures the model focuses on salient features necessary for accurate color detection. While traditional methods can distinguish lights from other bright objects, further improvement is possible. By concentrating on the illuminated portions of the traffic light box and excluding superfluous data, computational resources are conserved and model performance is significantly enhanced.

This refined annotation methodology results in more efficient and effective model operation, of the utmost importance for computer vision applications in traffic management and autonomous vehicle navigation. Hence, only the portion of the image displaying the active light is labeled, as depicted in Figure 8, to facilitate better learning by the model [32,34].

Finally, the main advantages of the above technique are the following:

- By considering a reduced set of features, the meta-model's learning process becomes more efficient. This targeted approach helps the model better differentiate between objects and their unique attributes.
- Eliminating unnecessary features simplifies the meta-model, making it easier to interpret and maintain. Additionally, this simplification can lead to faster inference times and reduced computational resource usage, resulting in lower latency.

- The simplified and focused model can operate effectively in harsh weather conditions, which presents a significant challenge for traditional models trained with conventional labeling data.

4.4. Evaluation Metrics

A comparative assessment of Meta-YOLOv8 was conducted against SSD, FRCNN, DETR, and the standard YOLOv8 models for traffic light detection. Key metrics such as precision, recall, F1 score, IoU, and mAP were utilized to evaluate the performance of these models. These indicators were chosen to measure the accuracy of detection, the congruence of predicted bounding boxes with actual data, and overall performance across different object classes. Additionally, the models' frame rates (FPS) were assessed to determine their capacity for real-time processing. Furthermore, the robustness of the models was tested under various lighting and weather conditions, along with their detection range and ability to manage occlusions, to evaluate their operational effectiveness and reliability [19].

4.5. Experiment Setup

Our configuration employs a combination of hardware and software elements to facilitate the training and deployment of our traffic light detection model. For training, we utilized Tesla T4 and A100 GPUs available through Colab's platform. Deployment on edge devices was achieved using the NVIDIA Jetson Nano. To prepare the dataset, we annotated our images using `LabelImg`, `MakesenseAI`, an open-source graphical image annotation tool.

The model was constructed in Python (3.10) using `TensorFlow` (2.8.0) and `PyTorch` (2.2.1). `OpenCV` (4.8.1) was utilized for image transformation and feature extraction, while `Matplotlib` (3.8.0) was employed for data visualization.

In our experiments, we replicated the training process of MAML (model-agnostic meta-learning) using a CNN backbone [7,29]. Instead of cloning, as in MAML, we employed two YOLOv8 models with identical configurations. We established a pipeline to facilitate the sharing of weights between these models (see Figures 5 and 9).

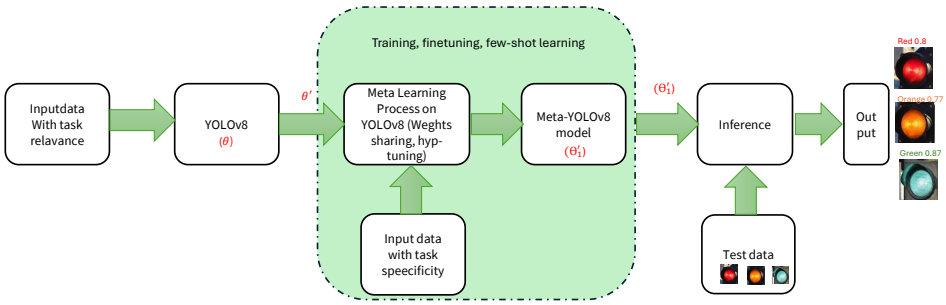

Figure 9. Data flow diagram of Meta-YOLOv8.

Initially, we used a pre-trained YOLOv8 model, which was then trained on a dataset relevant to the target classes, serving as the outer loop. The weights from this training were subsequently transferred to a second YOLOv8 model, which had fewer layers than the base model. This second model functioned as the inner loop during training for task-specific data, effectively leveraging the knowledge from the outer loop.

Details on the weight and loss function updates in the meta-learner (inner loop) are explained in the section below. The experimental files are available at the paper's code repository (last edit on 10 November 2024): https://github.com/VasuTammisetti/Meta-Learning-Enhanced-YOLOv8-for-Precision-Traffic-Light-Color-Detection-in-ADAS.

4.6. Training Process

The model's streamlined design renders it suitable for a multitude of applications and adaptable to various hardware platforms, ranging from edge devices to cloud APIs. Given the size of our dataset and the dimensions of the images, we selected a medium model from the YOLOv8 series with 25.9 parameters and 78.9 floating-point operations per second (FLOPs) as the base model [35].

To train a foundational model, we employed meta-learning strategies that leverage task similarity. Task similarity describes the extent to which different tasks share common characteristics or patterns. For instance, when preparing a model for traffic light detection, we pre-trained it using images of car turn signals and brake lights. These images share common color features with traffic lights and are more readily available.

We initially selected a high learning rate of 0.1 and substantial momentum to extract high-level features from input images [36,37]. In the subsequent phase, we conducted a thorough selection process to identify optimal values for the learning rate and momentum. Through experimentation and fine-tuning using the AutoKeras tool [38], we determined that a learning rate of 0.0089 and a momentum of 0.937 were most suitable for our training dataset. These choices, based on the concept of task similarity, have proven instrumental in effectively classifying traffic lights during the model's pre-training phase.

5. Results and Discussion

This section provides a comparative analysis of Meta-YOLOv8's training versus the standard YOLOv8 model. We begin by outlining the modifications and enhancements introduced by Meta-YOLOv8, emphasizing its advantages in scenarios with limited or specialized datasets. The discussion covers the theoretical foundations, training methodology, and dataset management strategies of Meta-YOLOv8. We then present associated results, focusing on key performance metrics such as F1 score, precision rate, box loss, and class loss. Graphs and images illustrate the recognition capabilities and adaptability of Meta-YOLOv8 under various conditions.

5.1. Meta-YOLOv8 Comparison with Base Model (YOLOV8)

The Meta-YOLOv8 training approach differentiates itself from the standard YOLOv8 framework by modifying the model's weights using a dataset that shares task similarities with the target domain. For instance, when developing a model to detect military trucks, which have unique characteristics and are rarely found on public roads, collecting a substantial number of images becomes challenging. Object detection models usually require a large corpus of data for effective training. Meta-learning techniques are particularly beneficial here because military and civilian trucks share many similarities, such as body parts, tires, and colors. This approach helps overcome the challenge of needing large amounts of data, addressing the rare-data issues common in traditional deep learning models.

When training with Meta-YOLOv8, the model's weights are strategically adjusted from the start (both inner and outer loops), as demonstrated in Figure 5. This careful tuning allows the model to learn more effectively, making it well suited for tasks such as recognizing different types of vehicles. During training, the model undergoes multiple iterations and epochs (an epoch is a complete pass through the entire training dataset, which consists of multiple iterations where each iteration updates the model using a batch of data), with each iteration refining its weights [39]. This process results in a model that is highly adept at detecting the desired objects. The dataset is divided into smaller, manageable units (usually called tasks in meta-learning), and the model's weights are fine-tuned on a task-by-task basis, progressively converging towards an optimal parameter set for the final object detection task.

Using this methodology, we trained our model with a limited set of examples to detect traffic light colors, aiming to develop a rapid and adaptable system capable of addressing new color detection challenges [40]. This approach mitigates the limitations of training data and facilitates the effective transfer of models to new, related detection tasks. It underscores the versatility and efficiency of Meta-YOLOv8 in specialized object detection scenarios.

A comparison of performance metrics between the two models reveals significant differences in efficacy. The Meta-YOLOv8 framework model demonstrates superior performance, with an F1 score of 93% compared to the base model's 54% (see Figure 10). The F1 score, a harmonic mean of precision and recall, provides a balanced measure of the model's accuracy, indicating a substantial improvement in identifying and classifying relevant instances.

The precision rate (PR) of our model is also notable at 97%, indicating highly accurate and reliable identification of relevant instances. In contrast, the base model achieves a precision rate of 52.5% (see Figure 10), highlighting its lower prediction capability. Precision is particularly valuable in contexts where the cost of false positives is high, and our model's elevated precision demonstrates its efficacy in such scenarios.

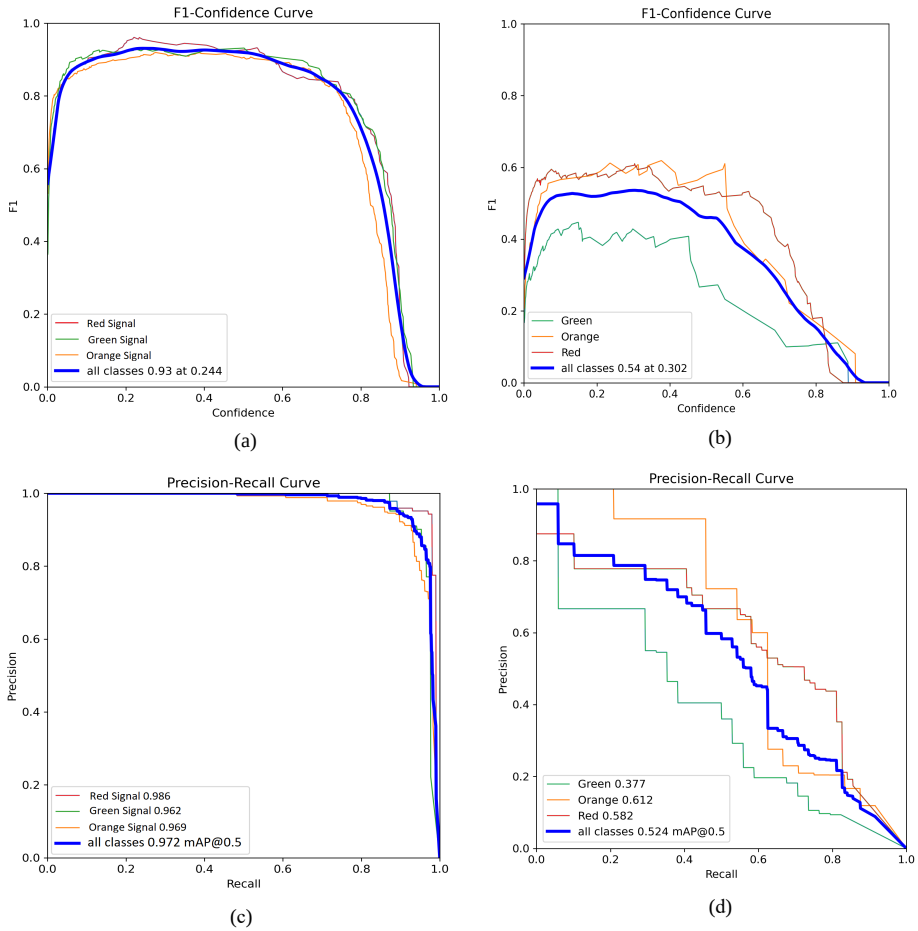

Figure 10. F1 score and precision comparison between Meta-YOLOv8 (**a**,**c**) and YOLOv8 (**b**,**d**).

Furthermore, the box loss—that quantifies the error in bounding box predictions—is significantly lower in our model (12%) compared to the base model (25%), as depicted in Figure 11. Lower box loss and variance in results indicate more accurate and stable predictions of object locations and classes, reducing misclassifications due to poor localization [19,41].

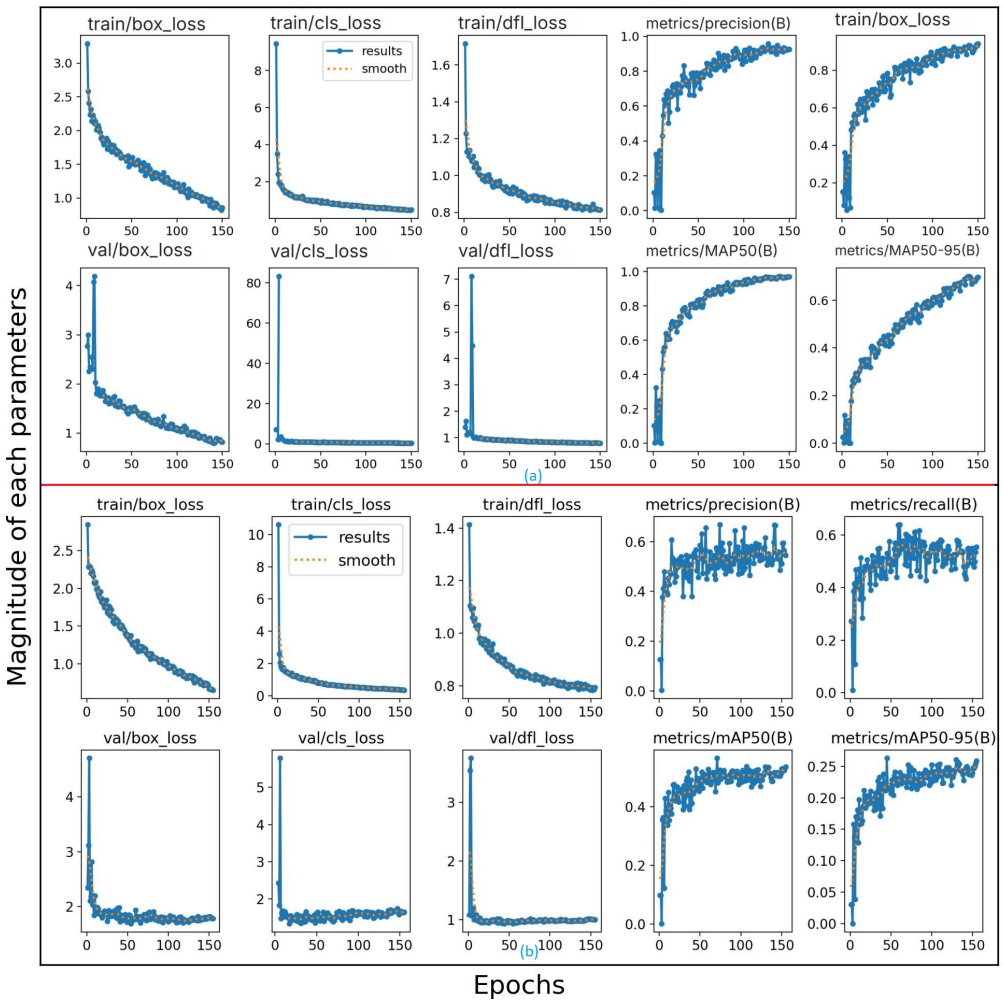

Figure 11. Comparison of different performance metrics between Meta-YOLOv8 (**a**) and YOLOv8 (**b**).

Class loss, which measures the discrepancy in assigning class labels to detected objects, is another relevant metric. Our model exhibits a class loss of 8%, compared to the base model's 18% (see Figure 11). This significant reduction in class loss and variance in test results highlights a good improvement in distinguishing between different classes within the dataset. This advancement is particularly advantageous for complex multiclass traffic light detection tasks [42].

The Meta-YOLOv8 model's effective performance metrics are primarily due to its meta-learning capabilities, which enhance its ability to understand patterns in input data. This improved pattern recognition allows the model to learn more efficiently, leading to better results shown in Figures 12 and 13.

Additionally, the model's improved validation metrics not only demonstrate its current accuracy and reliability but also highlight its capability to adapt to changes. This adaptability is an important thing for practical applications, where maintaining performance despite variations in data distributions or operational conditions is essential.

Figure 12. Meta-YOLOv8 (**b**) with new labeling exhibits superior detection capabilities relative to YOLOv8 (**a**) with conventional labeling, effectively minimizing ambiguity and precisely differentiating among various colors within a single frame devoid of overlap. YOLOv8 (**a**) uses overlapping bounding boxes, while Meta-YOLOv8 (**b**) displays a distinct delineation of colors, ensuring clarity for both observers and automated systems, thereby eliminating potential confusion.

Figure 13. The performance of Meta-YOLOv8 is demonstrated under various conditions, including different ranges and lighting scenarios: (**a**) effectiveness in bright daylight from approximately 200 m away, and (**b**) functionality in low light compared to a clear day (cloudy evening) from around 150 m from the driver's perspective. In both cases, traffic lights are accurately detected from a long range. Additionally, images (**c**,**e**) show testing in a morning light environment from approximately 200 m and 170 m, respectively. Images (**d**,**f**) depict a complete night-time scene, demonstrating the model's ability to detect and differentiate between street lights and traffic signals under challenging conditions with faint and glaring lights from an approximate distance of 50–65 m.

Moreover, the Meta-YOLOv8 model achieves these enhanced metrics with fewer training data compared to the base YOLOv8 model. This efficiency in learning from a smaller dataset underscores its potential for reduced computational resources and time, offering significant advantages during both the development and deployment phases of machine learning projects [19,41]. Notably, the model demonstrates robust performance even with minimal data, distinguishing it from conventional models. This superiority has been quantitatively validated using various metrics such as the F1 score and precision–recall curves. These metrics collectively showcase the model's ability to accurately identify relevant features and maintain accuracy across diverse test scenarios [43].

5.2. Model Adaptability

To evaluate the adaptability of our model—an essential aspect of meta-learning—we tested it under two distinct weather conditions: rain and heavy fog. These conditions were not part of the initial training phase, allowing us to assess the model's capacity for adaptation.

A modest dataset, comprising 20 images per scenario, was assembled, with 8 images designated for training and 2 for validation (2-way 8-shot). After training, the model was tested on previously unseen images, and the results are presented in Figures 14 and 15. Figures 14a,c and 15 illustrate the model's performance in heavy fog conditions, with visibility reduced to 25% and 40% compared to daytime conditions and a detection distance of approximately 55 and 75 m. The model also demonstrated adaptability in rainy conditions (Figures 14b,d and 15), achieving approximately 30% and 20% visibility and detection distances of around 60 and 70 m. These results suggest that the Meta-YOLOv8 model can adapt to new environments with minimal data, showcasing its proficiency in continuous learning.

Figure 14. Adaptability performance evaluation of Meta-YOLOv8 in adverse weather: (**a**,**c**) Detection capability during dense fog at approximately 55 and 75 m, respectively, and (**b**,**d**) operational efficiency in intense rain at approximately 60 and 70 m, respectively, from the driver's viewpoint.

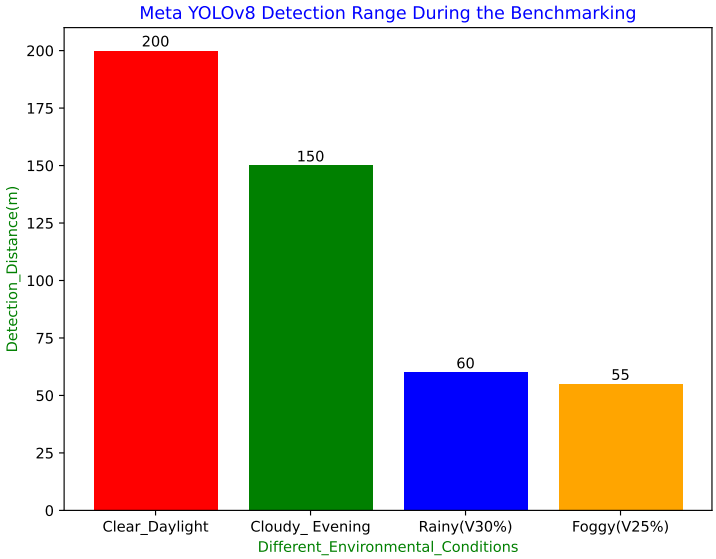

Figure 15. Performance of Meta-YOLOv8's detection range under various weather conditions, with 'V' denoting visibility.

5.3. Meta-YOLOv8 vs. Other Existing Methods

5.3.1. FPS Comparison

In the assessment of TLC detection models on the A100 GPU, based on frames per second (FPS), notable discrepancies emerge, underscoring the computational efficiency of each model [44]. The Single Shot Multibox Detector (SSD) achieves a processing speed of 42 FPS, indicating a strong preference for speed. In contrast, Meta-YOLOv8 exhibits a balanced trade-off between speed and accuracy, with a processing speed of 53 FPS, as shown in Figure 16. Despite operating at a relatively low frame rate of 28 FPS, Detection Transformers (DETR) is recognized for its exceptional accuracy and ability to handle intricate detection tasks. At the lower end of the spectrum is the Faster R-CNN model, operating at 7 FPS. Despite its relative slowness, it is favored for its high precision in various object detection scenarios. Notably, among the detection architectures considered, Meta-YOLOv8 demonstrates proficiency during inference, offering an advantageous combination of speed and accuracy.

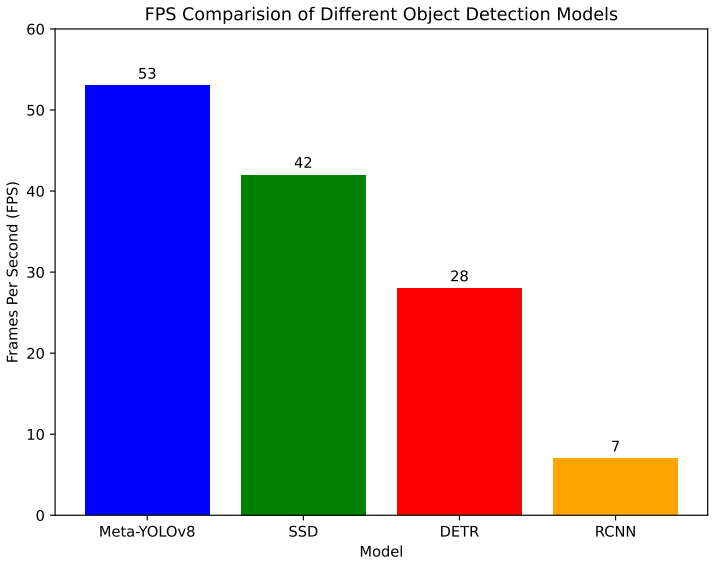

Figure 16. FPS (frames per second) comparison of different TLC detection models during inference.

5.3.2. Mean Average Precision (mAP)

The mean average precision (mAP) is typically assessed at an intersection over union (IoU) threshold of 50%, denoted as mAP@0.5, and across a range from 50% to 95%, denoted as mAP@0.5:0.95. A comparison of various TLC detection models reveals considerable variation in detection accuracy across these two mAP metrics (see Figure 17). Our results show that Meta-YOLOv8 attains the highest detection precision, with an impressive mAP@0.5 score of 97% and an mAP@0.5:0.95 score of 67%, thereby exhibiting robust performance across a wide range of IoU thresholds. This illustrates Meta-YOLOv8's capacity for precise and reliable TLC detection. The SSD model also demonstrates noteworthy performance, achieving an mAP@0.5 of 41% and an mAP@0.5:0.95 of 13%. These results indicate that SSD effectively maintains a balance between detection speed and accuracy, making it a practical choice for real-time applications.

Meanwhile, Faster R-CNN, which prioritizes precision, records an mAP@0.5 of 23.4% and an mAP@0.5:0.95 of 11.3%. These are enough for TLC detection tasks. At the lower end of the spectrum, Detection Transformers exhibits the least favorable scores, with an mAP@0.5 of 13.8% and an mAP@0.5:0.95 of 3%. Despite their sophisticated methodology for addressing intricate detection challenges, these outcomes suggest that their current iteration exhibits comparatively low precision. This diverse range of performance highlights the need to select an appropriate object detection model based on the specific requirements of accuracy and computational efficiency for the intended application.

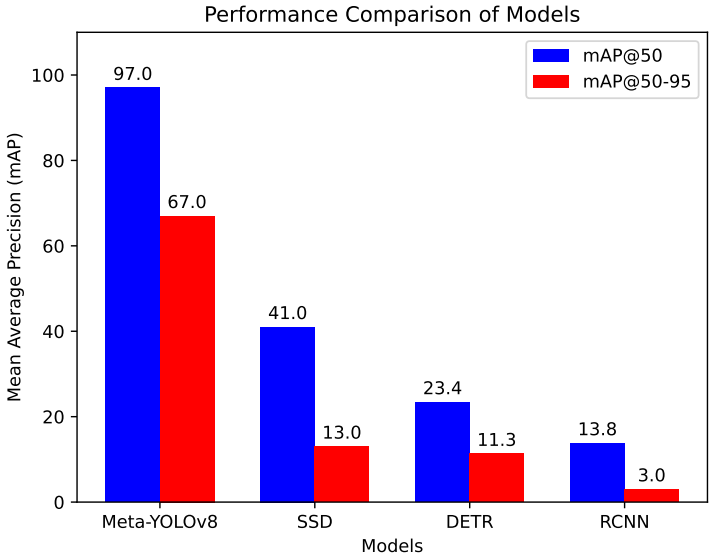

Figure 17. Comparison of mAP of different TLC detection models.

5.3.3. Test Accuracy

A comparison of the test accuracy of different TLC detection models reveals significant discrepancies in performance. Meta-YOLOv8 stands out, with a notable test accuracy of 93%, indicating its exceptional capability to accurately identify objects in test scenarios. In contrast, SSD-300 demonstrates a test accuracy of 44.83%, showcasing a robust performance that balances speed and accuracy. The Faster R-CNN achieves a test accuracy of 27%, as shown in Figure 18, reflecting a moderate level of precision in its detections. Despite its advanced architecture designed for complex detection tasks, DETR has the lowest accuracy among the models considered, with a test accuracy of 23.40% [19,41].

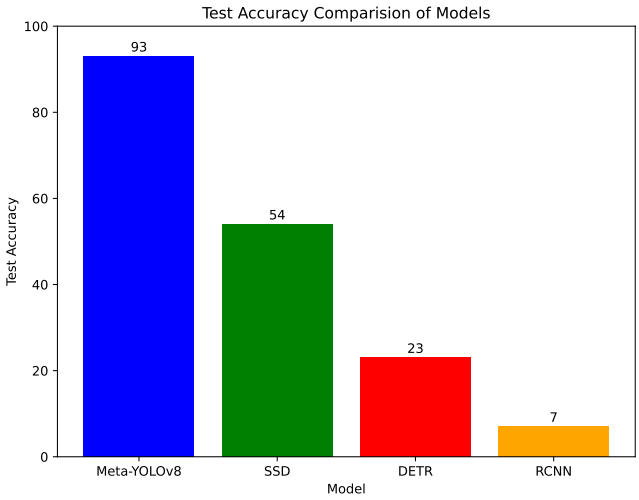

Figure 18. Test accuracy comparison of different TLC detection models.

5.3.4. FLOPS and Parameters

In assessing the computational efficiency of various TLC detection models, notable differences emerge when comparing the number of floating-point operations (FLOPs) and parameter counts. The Meta-YOLOv8 and YOLOv8 models stand out for their efficiency, requiring only 79 billion FLOPs and comprising 25 million parameters (see Figure 19). These attributes contribute to their lightweight and rapid performance. In contrast, the SSD model requires 175 billion FLOPs and has a parameter count of 25.013 million, striking a balance between computational complexity and efficiency. The Faster R-CNN model, demanding 278 billion FLOPs and containing 278 million parameters, is the most computationally intensive, reflective of its capability for detailed and accurate detection. Similarly, the DETRs model, which uses a Transformer-based architecture, requires 60.53 billion FLOPs and has 43.555 million parameters (see Figure 19). This represents a moderate trade-off between efficiency and the complexity of the model's design. This comparative analysis underscores the inherent trade-offs between the computational demands of TLC detection paradigms and the sophistication of their model architectures [45].

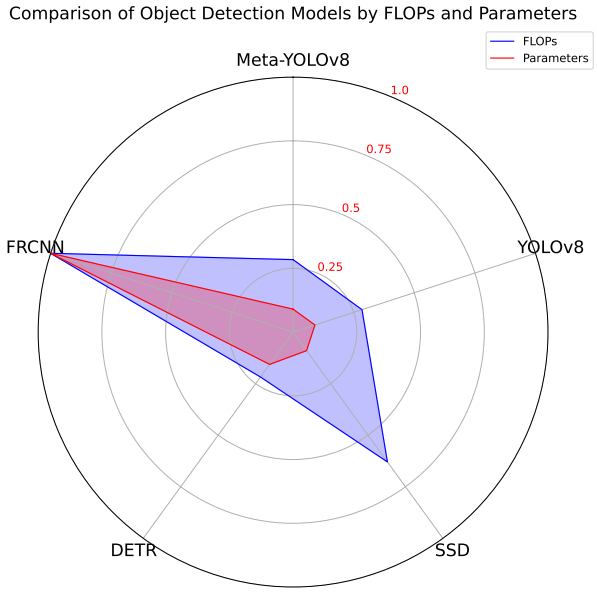

Figure 19. FLOPS and parameter comparison of different TLC detection models.

Empirical results show that the Meta-YOLOv8 model outperforms the base YOLOv8 model across all examined metrics. This demonstrates that integrating meta-learning principles into our model's structure not only enhances the learning process but also significantly improves predictive performance. Given these findings, Meta-YOLOv8 represents a major advancement in the field of object detection, providing a valuable tool for applications requiring high precision and reliability.

6. Conclusions

Our experiments have demonstrated the potential of the Meta-YOLOv8 model for traffic light detection in ADAS. By targeting the light segments of the traffic lights, our model achieves improved accuracy and consistent performance over long distances and varying lighting and weather conditions. Compared to SSD, Faster R-CNN, DETR, and conventional YOLOv8, Meta-YOLOv8 reduces computational load while maintaining high reliability, as evidenced by an exceptional F1 score, accuracy of 93% and precision rate of 97%. The model demonstrates precise localization and classification of traffic lights, along with improved differentiation between bright objects and TLCs. This capability is crucial for real-time ADAS applications. It also shows efficient learning from smaller datasets and significant reductions in box and class loss. However, challenges remain, such as the need for extensive fine-tuning to distinguish similar hues and the inherent computational complexity of meta-learning.

Additional refinement is necessary to enhance the model's adaptability when handling a larger number of target classes (exceeding 10). This improvement should also focus on mitigating catastrophic forgetting, especially across different geographic regions with diverse traffic light configurations. Continuous development and optimization will be critical to realizing the full potential of Meta-YOLOv8, especially in resource-constrained environments and under varying conditions. Future efforts will focus on improving algorithmic structure, refining data representation, and increasing model versatility. The use of diverse and synthetic datasets will enhance generalization and detection capabilities, especially for rare traffic lights. Additionally, advancements in color differentiation under varying lighting conditions and further reduction in computational requirements will address deployment challenges. Developing flexible algorithms to accommodate regional variations in traffic light design will further enhance the global applicability of the model.

Limitations

The proposed Meta-YOLOv8 model differs from traditional object detection approaches, which typically rely on incremental training and fine-tuning. Meta-YOLOv8's performance is highly data-driven, emphasizing the necessity for carefully curated input data to maintain task similarity for effective traffic light detection. One significant challenge is the standardization of traffic lights across different regions, requiring the inclusion of diverse data to ensure model generalization. Consequently, the model may struggle to recognize unusual traffic lights not represented in the training data and to discriminate closely related colors at a distance, such as orange and red, or due to biases from common backgrounds.

Furthermore, the model's ability to generalize to traffic light detection is further constrained by international variations in traffic light design, underscoring the need for continuous adaptation to achieve robust global performance.

Author Contributions: Conceptualization: V.T., M.P.C., and M.M.-S.; methodology: V.T. and M.M.-S.; software: V.T. and M.M.-S.; formal analysis: V.T. and M.M.-S.; resources: G.S.; writing (review and editing): V.T., G.S., M.P.C., and M.M.-S.; supervision: G.S., M.P.C., and M.M.-S.; project administration: V.T., G.S., and M.M.-S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Infineon Technologies AG (Munich, Germany) and the University of Granada (Spain). It received funding from the European Union's Horizon Europe Research and Innovation Program through Grant Agreement No. 101076754 (AIthena project). This work was also partially funded by the Spanish Ministry of Economic Affairs and Digital Transformation (NextGenerationEU funds) through the project IA4TES MIA.2021.M04.0008.

Data Availability Statement: Data is contained within the article.

Conflicts of Interest: The authors Vasu Tammisetti and Georg Stettinger were employed by Infineon Technologies AG. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationship that could be construed as potential conflicts of interest.

References

1. Mogelmose, A.; Trivedi, M.M.; Moeslund, T.B. Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey. *IEEE Trans. Intell. Transp. Syst.* **2012**, *13*, 1484–1497. [CrossRef]
2. Zhai, C.; Li, K.; Zhang, R.; Peng, T.; Zong, C. Phase diagram in multi-phase heterogeneous traffic flow model integrating the perceptual range difference under human-driven and connected vehicles environment. *Chaos Solitons Fractals* **2024**, *182*, 114791. [CrossRef]
3. Navarro Lafuente, A. Business Modelling of 5G-Based Drone-as-a-Service Solution. Master's Thesis, Universitat Politècnica de Catalunya, Barcelona, Spain, 2024.
4. Jain, A.; Mishra, A.; Shukla, A.; Tiwari, R. A novel genetically optimized convolutional neural network for traffic sign recognition: A new benchmark on Belgium and Chinese traffic sign datasets. *Neural Process. Lett.* **2019**, *50*, 3019–3043. [CrossRef]
5. Gautam, S.; Kumar, A. Image-based automatic traffic lights detection system for autonomous cars: A review. *Multimed. Tools Appl.* **2023**, *82*, 26135–26182. [CrossRef]
6. Varghese, R.; Sambath, M. YOLOv8: A Novel Object Detection Algorithm with Enhanced Performance and Robustness. In Proceedings of the 2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS), Chennai, India, 18–19 April 2024; IEEE: Piscataway, NJ, USA, 2024; pp. 1–6.
7. Finn, C.; Abbeel, P.; Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; PMLR: Proceedings of Machine Learning Research: New York, NY, USA, 2017; pp. 1126–1135.
8. Beyaz, A.; Gerdan, D. Meta-learning-based prediction of different corn cultivars from color feature extraction. *J. Agric. Sci.* **2021**, *27*, 32–41.
9. Binangkit, J.L.; Widyantoro, D.H. Increasing accuracy of traffic light color detection and recognition using machine learning. In Proceedings of the 2016 10th International Conference on Telecommunication Systems Services and Applications (TSSA), Denpasar, Indonesia, 6–7 October 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 1–5.
10. Pandharkar, M.; Raoundale, P. A Systematic Study of Approaches used to Address the Long Tail Problem. In Proceedings of the 2023 10th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 15–17 March 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 430–437.
11. Chen, Q.; Dai, Z.; Xu, Y.; Gao, Y. CTM-YOLOv8n: A Lightweight Pedestrian Traffic-Sign Detection and Recognition Model with Advanced Optimization. *World Electr. Veh. J.* **2024**, *15*, 285. [CrossRef]
12. Müller, J.; Dietmayer, K. Detecting traffic lights by single shot detection. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 266–273.
13. Almagambetov, A.; Velipasalar, S.; Baitassova, A. Mobile standards-based traffic light detection in assistive devices for individuals with color-vision deficiency. *IEEE Trans. Intell. Transp. Syst.* **2014**, *16*, 1305–1320. [CrossRef]
14. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Proceedings, Part I 14; Springer: Berlin/Heidelberg, Germany, 2016; pp. 21–37.
15. You, S.; Bi, Q.; Ji, Y.; Liu, S.; Feng, Y.; Wu, F. Traffic sign detection method based on improved SSD. *Information* **2020**, *11*, 475. [CrossRef]
16. Tian, Y.; Yang, G.; Wang, Z.; Wang, H.; Li, E.; Liang, Z. Apple detection during different growth stages in orchards using the improved YOLO-V3 model. *Comput. Electron. Agric.* **2019**, *157*, 417–426. [CrossRef]
17. Safaldin, M.; Zaghden, N.; Mejdoub, M. An Improved YOLOv8 to Detect Moving Objects. *IEEE Access* **2024**, *12*, 59782–59806. [CrossRef]

18. Zaatouri, K.; Ezzedine, T. A self-adaptive traffic light control system based on YOLO. In Proceedings of the 2018 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC), Hamammet, Tunisia, 20–21 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 16–19.
19. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In Proceedings of the 29th Advances in Neural Information Processing Systems (NeurIPS), Montreal, QC, Canada, 7–12 December 2015; Volume 2.
20. Gavrilescu, R.; Zet, C.; Foșalău, C.; Skoczylas, M.; Cotovanu, D. Faster R-CNN: An approach to real-time object detection. In Proceedings of the 2018 International Conference and Exposition on Electrical And Power Engineering (EPE), Iasi, Romania, 18–19 October 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 165–168.
21. Chuang, C.H.; Lee, C.C.; Lo, J.H.; Fan, K.C. Traffic Light Detection by Integrating Feature Fusion and Attention Mechanism. *Electronics* **2023**, *12*, 3727. [CrossRef]
22. Jiang, Z.; Zhao, L.; Li, S.; Jia, Y. Real-time object detection method based on improved YOLOv4-tiny. *arXiv* **2020**, arXiv:2011.04244.
23. Arnold, S.M.; Mahajan, P.; Datta, D.; Bunner, I.; Zarkias, K.S. learn2learn: A library for meta-learning research. *arXiv* **2020**, arXiv:2008.12284.
24. Ren, X.; Zhang, W.; Wu, M.; Li, C.; Wang, X. Meta-yolo: Meta-learning for few-shot traffic sign detection via decoupling dependencies. *Appl. Sci.* **2022**, *12*, 5543. [CrossRef]
25. Shmelkov, K.; Schmid, C.; Alahari, K. Incremental learning of object detectors without catastrophic forgetting. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 3400–3409.
26. Flores-Calero, M.; Astudillo, C.A.; Guevara, D.; Maza, J.; Lita, B.S.; Defaz, B.; Ante, J.S.; Zabala-Blanco, D.; Armingol Moreno, J.M. Traffic sign detection and recognition using YOLO object detection algorithm: A systematic review. *Mathematics* **2024**, *12*, 297. [CrossRef]
27. Karim, M.J.; Nahiduzzaman, M.; Ahsan, M.; Haider, J. Development of an Early Detection and Automatic Targeting System for Cotton Weeds using an Improved Lightweight YOLOv8 Architecture on an Edge Device. *Knowl.-Based Syst.* **2024**, *300*, 112204. [CrossRef]
28. Finn, C.B. *Learning to Learn with Gradients*; University of California: Berkeley, CA, USA, 2018.
29. Tammisetti, V.; Bierzynski, K.; Stettinger, G.; Morales-Santos, D.P.; Cuellar, M.P.; Molina-Solana, M. LaANIL: ANIL with Look-Ahead Meta-Optimization and Data Parallelism. *Electronics* **2024**, *13*, 1585. [CrossRef]
30. Starck, J.L.; Murtagh, F.; Bijaoui, A. *Image Processing and Data Analysis: The Multiscale Approach*; Cambridge University Press: Cambridge, UK, 1998.
31. Vandaele, R.; Nervo, G.A.; Gevaert, O. Topological image modification for object detection and topological image processing of skin lesions. *Sci. Rep.* **2020**, *10*, 21061. [CrossRef]
32. Rädsch, T.; Reinke, A.; Weru, V.; Tizabi, M.D.; Schreck, N.; Kavur, A.E.; Pekdemir, B.; Roß, T.; Kopp-Schneider, A.; Maier-Hein, L. Labelling instructions matter in biomedical image analysis. *Nat. Mach. Intell.* **2023**, *5*, 273–283. [CrossRef]
33. Li, R.; Cao, W.; Wu, S.; Wong, H.S. Generating target image-label pairs for unsupervised domain adaptation. *IEEE Trans. Image Process.* **2020**, *29*, 7997–8011. [CrossRef]
34. Badrinarayanan, V.; Handa, A.; Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling. *arXiv* **2015**, arXiv:1505.07293.
35. Lee, Y.; Hwang, J.w.; Lee, S.; Bae, Y.; Park, J. An energy and GPU-computation efficient backbone network for real-time object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Long Beach, CA, USA, 16–20 June 2019; IEEE: Piscataway, NJ, USA, 2019.
36. Sutskever, I.; Martens, J.; Dahl, G.; Hinton, G. On the importance of initialization and momentum in deep learning. In Proceedings of the International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; pp. 1139–1147.
37. Smith, L.N.; Topin, N. Super-convergence: Very fast training of neural networks using large learning rates. In Proceedings of the Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications, Baltimore, MD, USA, 10 May 2019; SPIE: Bellingham, WA, USA, 2019; Volume 11006, pp. 369–386.
38. Sobrecueva, L. *Automated Machine Learning with AutoKeras: Deep Learning Made Accessible for Everyone with Just Few Lines of Coding*; Packt Publishing Ltd.: Birmingham, UK, 2021.
39. Fu, K.; Zhang, T.; Zhang, Y.; Yan, M.; Chang, Z.; Zhang, Z.; Sun, X. Meta-SSD: Towards fast adaptation for few-shot object detection with meta-learning. *IEEE Access* **2019**, *7*, 77597–77606. [CrossRef]
40. Wang, Y.X.; Ramanan, D.; Hebert, M. Meta-learning to detect rare objects. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 9925–9934.
41. Yan, J.; Wang, H.; Yan, M.; Diao, W.; Sun, X.; Li, H. IoU-adaptive deformable R-CNN: Make full use of IoU for multi-class object detection in remote sensing imagery. *Remote Sens.* **2019**, *11*, 286. [CrossRef]

42. Wang, S.; Zhang, Z.; Chao, Q.; Yu, T. AFE-YOLOv8: A Novel Object Detection Model for Unmanned Aerial Vehicle Scenes with Adaptive Feature Enhancement. *Algorithms* **2024**, *17*, 276. [CrossRef]
43. Chabi Adjobo, E.; Sanda Mahama, A.T.; Gouton, P.; Tossa, J. Automatic localization of five relevant Dermoscopic structures based on YOLOv8 for diagnosis improvement. *J. Imaging* **2023**, *9*, 148. [CrossRef]
44. Wang, G.; Luo, C.; Sun, X.; Xiong, Z.; Zeng, W. Tracking by instance detection: A meta-learning approach. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 6288–6297.
45. Wang, N.; Gao, Y.; Chen, H.; Wang, P.; Tian, Z.; Shen, C.; Zhang, Y. NAS-FCOS: Fast neural architecture search for object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 11943–11951.