International Journal of Informatics and Applied Mathematics
e-ISSN:2667-6990 Vol. 7, No. 1, 33-56

YOLO Network-based URL Detection in Varied Conditions with Small-Sample Insights

Leila Boussaad[1,2] and Aldjia Boucetta[1,2]

[1] Management dept., Batna 1 University, Batna, Algeria
[2] LAMIE Laboratory, Computer science dept., Batna 2 University, Batna, Algeria
boussaad.mous@gmail.com, boucetta_batna@yahoo.fr

Abstract. Object detection is a pivotal aspect of computer vision, essential for diverse recognition tasks. This study centers on exploring deep learning methodologies for object detection, specifically targeting the identification of URLs in images captured by mobile phones. We conduct a comparative analysis of three models from the YOLO family – YOLOv3, YOLOv4, and YOLOv5 – recognized for their efficacy in object detection. Our research addresses the unique challenge of detecting URLs in images, particularly considering the limited availability of URL-labeled dataset. Through rigorous experimentation and evaluation, we demonstrate the generalization capabilities of YOLOv3, YOLOv4, and YOLOv5, as measured by average precision scores. Furthermore, we highlight the resilience of the YOLOv4 model against various image-related challenges. Our findings contribute significantly to the advancement of computer vision, specifically in the domain of object detection for real-world applications. By evaluating the performance of cutting-edge deep learning models, we provide valuable insights into their effectiveness for URL detection, thereby enriching our understanding of their practical utility. This research serves as a foundation for future investigations aimed at leveraging deep learning techniques to enhance object detection accuracy across diverse contexts.

Keywords: Object Detection · Deep Learning · Convolutional Neural Networks (CNN) · YOLO · URL Detection · Small Sample.

Received: 19 December 2023 | Revised: 20 April 2024 | Accepted: 25 April 2024
DOI: 10.53508/ijiam.1406569

1 Introduction

Object detection plays a central role in any recognition system, encompassing the task of identifying an object's class and estimating its spatial coordinates by delineating a bounding frame around the object. Recent advancements in deep learning-based object detection have delivered remarkable outcomes. However, the real-world implementation of object detection faces a host of challenges when confronted with actual images, including factors like noise, occlusion, lighting fluctuations, rotations, and others. These elements have a pronounced impact on the precision of object detection and demand thorough scrutiny during the detection process.

Conversely, the web has consistently served as a medium that allows the transfer of data in a simple and fast way. It counts as a necessary tool in modern life, offering a multitude of prospects for both individuals and large corporations.

World Wide Web, often referred to as the Web or WWW, encompasses all publicly accessible websites and pages that users can access on their local devices via the Internet. These pages and documents are interconnected through hypertext links, which users can click to access information. This information can take various forms, including text, images, audio, and video. To visit a website, a specific page on a site, or more precisely, an "online resource" (such as content or an online service), users can enter its address, known as a Uniform Resource Locator (URL), into the browser's address bar. The URL is indispensable for pinpointing a particular page within the vast sea of billions of web pages. Each web resource possesses a unique URL, which serves as the web address displayed in your browser.

To be more efficient and remove the step of entering the URL, especially with increasing processing capabilities such as the availability of smartphones and visual input devices such as cameras built into smartphones, this process can be divided into three steps: image acquisition, URL localization, and URL recognition. In this paper, we will mainly focus on the second step, which plays a crucial role in the localization of URLs in a captured image containing text. Object detection methods are well suited to accomplish this process. The detection of a URL can be useful in several fields, particularly in the field of tourism. The tourist can take a picture of a URL and view website information without having to type on their keyboard. Businesses, store owners, and their customers can advertise and post information by leaving URLs to the services they offer on their ad slots, and users can retrieve the URL(s) by clicking a button. The model can also be used to capture URL references while listening to a presentation at a conference. Given the capability of smartphone cameras lately, taking a picture of a URL in transit should result in a "good quality" image that can be passed as input to a model, and the URL(s) of interest will be recovered.

The primary objective of this study is to evaluate the efficacy of three established object detection models, namely YOLOv3, YOLOv4, and YOLOv5, in the specific context of detecting URLs within images. This task presents several challenges due to the diverse conditions under which the images are captured, including variations in lighting, orientation, and the presence of noise.

Our motivation to employ YOLO models for URL detection stems from their capability to execute real-time object detection directly from images, eliminating the need for extensive preprocessing or feature engineering. Unlike conventional methods reliant on handcrafted features and post-processing techniques, YOLO models offer a more streamlined and efficient approach. By evaluating the performance of YOLO models in detecting URLs across diverse conditions, our goal is to offer valuable insights into their efficacy and potential real-world applications. Furthermore, we aim to discern any distinctive advantages these models may possess over existing methods for URL detection.

Through this investigation, we endeavor to advance the comprehension of object detection methodologies and their practical relevance in addressing challenges associated with URL detection in images.

The subsequent sections of the manuscript are designed as follows: Section 2 provides a comprehensive review of the related work in object detection using YOLO models, this section aims to establish the context and significance of our research within the broader landscape of computer vision. Following this, Section 3 offers a concise overview of the key concepts underpinning this paper, highlighting the distinctive attributes of the chosen models, which motivated our selection. Section 4 outlines the evaluation process we employed, detailing the dataset, metrics, and experimental setup. Section 5 is dedicated to the presentation of experimental results and ensuing discussions, where we analyze the performance of the YOLO models in URL detection tasks. Finally, in Section 6, the paper draws its conclusions.

2 Literature review

The YOLO network has been widely utilized across various domains, encompassing tasks such as detecting malicious URLs [1], identifying small ships in optical images [2], analyzing smoking behavior in images [3], and recognizing vehicle targets [4]. These applications underscore the versatility and effectiveness of the YOLO algorithm in object detection tasks.

Advancements in object detection have been driven by models like YOLOv3, YOLOv4, and YOLOv5, each tailored to address specific challenges encountered in real-world scenarios. YOLOv5 introduces the DK_YOLOv5 model, optimized for low-light conditions and demonstrating superior accuracy compared to other models [5]. Additionally, YOLOv5 has been pivotal in creating real-time detectors resilient to lighting and rotation variations, as evidenced in applications like vehicle wheel detection [6]. These models have significantly propelled object detection technology, finding applications ranging from facial recognition to autonomous vehicles [6]. Moreover, advanced techniques have been proposed, such as an enhanced YOLOv3 method for small object detection, incorporating modules like DCM, CBAM, and multi-level fusion to enhance feature expression and accuracy [7].

Researchers have also explored practical implications of YOLO-based models. For instance, a website developed using the YOLOv3 algorithm assists visu-

ally impaired individuals in real-time object recognition and auditory guidance [8]. Additionally, studies on the scalability of on-device object detection using YOLOv4, CNNs, and TensorFlow Lite have addressed challenges related to limited computational resources and real-time performance [9]. Techniques such as model compression, quantization, and hardware accelerators have been evaluated to optimize efficiency and effectiveness, laying the foundation for more robust object detection systems. These advancements underscore the versatility and effectiveness of YOLO-based models in overcoming diverse challenges encountered in object detection scenarios.

Leveraging the capabilities of the YOLO network, tailored approaches can be developed to accurately detect URLs within photos, harnessing the algorithm's demonstrated high accuracy and recall rates across various object detection tasks. This objective forms the central focus of the present study.

3 Backgrounds

Prior to delving into the process and the diverse techniques employed in object detection, it is essential to establish a precise comprehension of object detection itself. Frequently, this term is used interchangeably with techniques like image classification, object recognition, segmentation, and more. Nonetheless, it is imperative to acknowledge that many of the techniques mentioned are distinct tasks typically encompassed within the broader realm of object detection. Treating them as synonyms is inaccurate, as each corresponds to a task of equal importance. Thus, we can distinguish these computer vision tasks [10]:

Image classification is about predicting the class of an element in an image, while **object localization** is about locating the presence of objects in an image and indicating their location using a bounding box (see Figure 1a), and **object detection** is about locating the presence of objects with a bounding box and the types or classes of objects located in an image. Figure 1b clearly shows the result of an object detection process in a road scene.

Another extension of this division of computer vision tasks is **semantic image segmentation**, where instances of recognized objects are indicated by highlighting specific pixels of the object. This technique gives a precise location (at the pixel level) of an object and the pixels found. The pixels produced can also be called a mask (see Figure 1c). Combining semantic segmentation with object detection leads to **instance segmentation**, which first detects object instances and then segments each into detected boxes (in this case called regions of interest). In other words, each object in the image gets its own unique mask, even if there are other objects with the same class (see Figure 1d).

In this paper, we will focus on object detection, where it is widely accepted that progress in this field has generally crossed two periods: "the traditional object detection period (before 2014)" and "the period of deep learning-based detection (after 2014)" [11].

During the traditional object detection period (before 2014), object detection predominantly relied on classical machine learning techniques. Among the

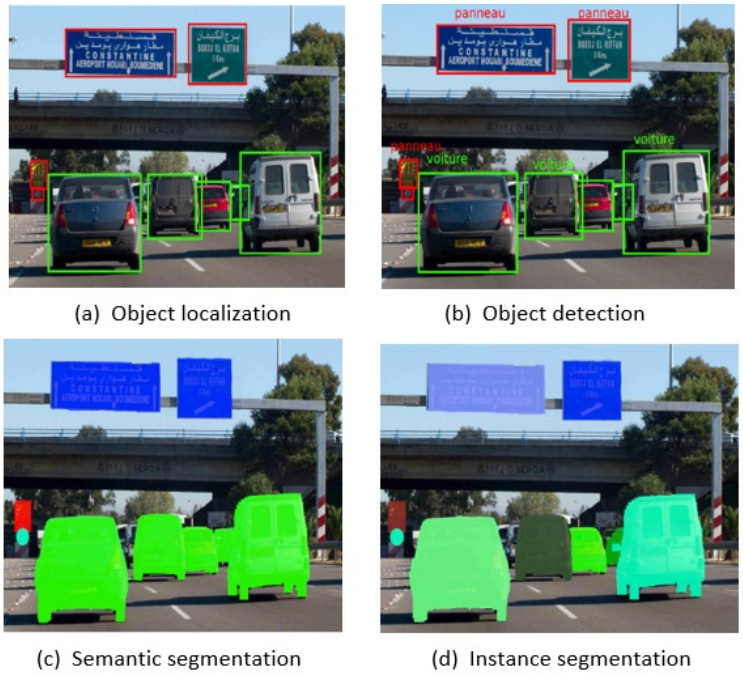


Fig. 1: The different computer vision tasks.

notable methods that emerged during this period, three significant approaches are worthy of mention. These include the Viola-Jones detector, originally developed in 2001 by Paul Viola and Michael Jones [12] for real-time human face detection, and it has demonstrated its utility in diverse applications. The Histogram of Oriented Gradients (HOG), introduced in 2005 by N. Dalal and B. Triggs [13], represents an enhancement over SIFT descriptors, shape contexts, and contour orientation histograms. HOG provides a robust, scale-invariant solution. Additionally, there is the Deformable Partial Model (DPM) [14], initially proposed by P. Felzenszwalb in 2008 as an extension of the HOG detector. DPM introduced a novel strategy involving learning the components and their overall structure.

Despite the success of traditional approaches, the effort required to create effective and efficient detection models remains significant. Therefore, they have been completely replaced by methods based on deep neural networks, resulting in greater accuracy and generalization [11]. In the era of deep learning, object detection is grouped into two classes: "two-step detection" and "one-step detection". Typically, an object detector solves two successive tasks: finding an arbitrary number of objects (perhaps even zero) and classifying each object and estimating its size using a bounding box. Methods that combine both tasks in one step are called single-step detectors.

One-stage detectors skip the region proposal step, which is generally part of two-stage object detection like Faster-RCNN and Mask-Mask-RCNN, and perform detection directly on a dense sampling of locations. They generally consider all positions in the image as potential objects and try to classify each region of interest as a background or target object.

Within the realm of single-step object detection algorithms, a noteworthy mention goes to the YOLO (You Only Look Once) family of algorithms [15], which serves as the central focus of this investigation. This approach employs a single, fully trained neural network that receives an image as input and directly generates predictions for bounding boxes and their associated class labels for an entire image in one pass. This end-to-end optimization, akin to image classification, contributes to its remarkable speed. The base YOLO model achieves a prediction rate of 45 FPS (Frames Per Second), as benchmarked on a Titan X GPU [15].

Another noteworthy discovery presented by Redmon et al. [15] highlighted the broad applicability of YOLO to both artwork and natural images sourced from the internet. Additionally, YOLO demonstrated superior performance compared to detection techniques such as the Deformable Parts Model (DPM) and Region-Based Convolutional Neural Networks (RCNN), surpassing them by a significant margin.

In the subsequent sections, we provide a concise overview of the three models under consideration.

3.1 YOLOv3

YOLOv3 [16] is primarily comprised of two key components: a feature extractor and a detector. The initial step involves passing the image through the feature extractor known as Darknet-53. Darknet-53 is responsible for processing the image and generating feature maps at various scales. These feature maps at each scale are subsequently directed into distinct branches of the detector. The detector's primary function is to process these multiple feature maps at diverse scales, culminating in the creation of output grids that contain objectivity scores and bounding boxes. The complete architecture of YOLOv3 is illustrated in Figure 2.

Darknet-53 integrates both residual blocks and Feature Pyramid Networks (FPNs), as illustrated in Figure 3. Serving as a feature extractor, Darknet-53 accepts single-scale images of arbitrary dimensions as input and yields appropriately scaled multi-level feature maps. This feature design enables exceptional performance across a broad range of input resolutions.

3.2 YOLOv4

Developed in 2020 by Alexei Bochkovsky, the YOLOv4 [18] architecture features CSPDarknet53 as its backbone, which builds upon the foundation of DarkNet-53. It incorporates a CSPNet (Cross Stage Partial Network) strategy, as referenced

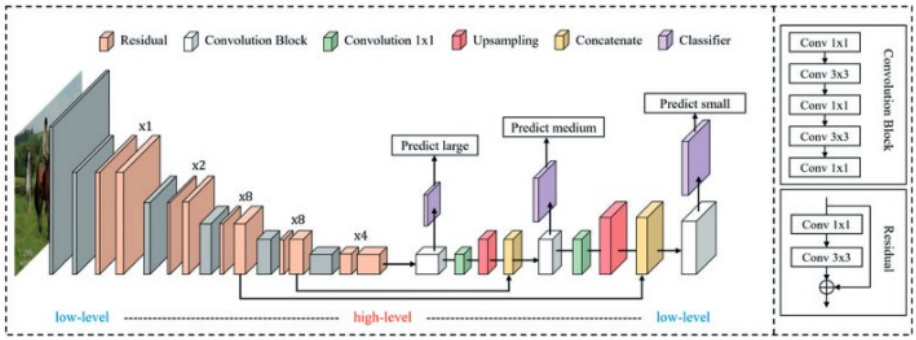


Fig. 2: YOLOv3 Architecture [17].

Fig. 3: Feature Pyramid Network (FPN)[17].

in [19], to partition the base layer's feature map into two segments and subsequently reunite them through a multistep hierarchy. This division and reunification approach facilitates more degraded flow within the network. Following the backbone, YOLOv4 adopts PANet (Path Aggregation Network), as cited in [20], as a parameter aggregation method from various levels of the backbone for distinct detector levels, deviating from the FPN approach employed in YOLOv3.

Furthermore, an SPP block, as referenced in [21], is introduced for the notable expansion of the receptive field. This block effectively isolates crucial contextual features while maintaining minimal impact on network operational speed. Finally, YOLOv3 is employed as the network's head, tasked with extracting pertinent features. Figure 4 provides a clear structure of the YOLOv4 architecture.

3.3 YOLOv5

Developed by Ultralytics in 2020 [22], this development marked a substantial enhancement in facilitating real-time object detection. The shift from Darknet to PyTorch as a framework played a pivotal role in this improvement. Darknet, known for its complexity in configuration and limited production readiness, was surpassed by PyTorch, leading to significant reductions in both training and prediction times.

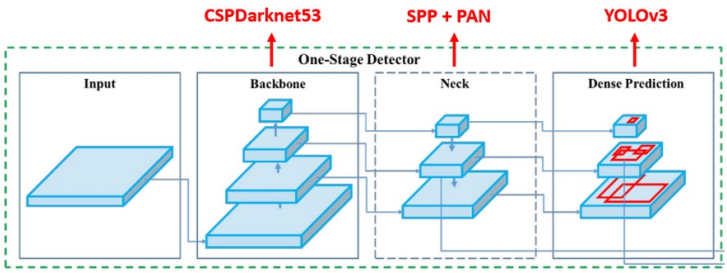


Fig. 4: YOLOv4 Architecture [18].

The model's architecture bears similarities to YOLOv4, incorporating CSP-Darknet53 as the backbone, SPP and PANet for the neck, and employing YOLOv3 as the head, as illustrated in Figure 5.

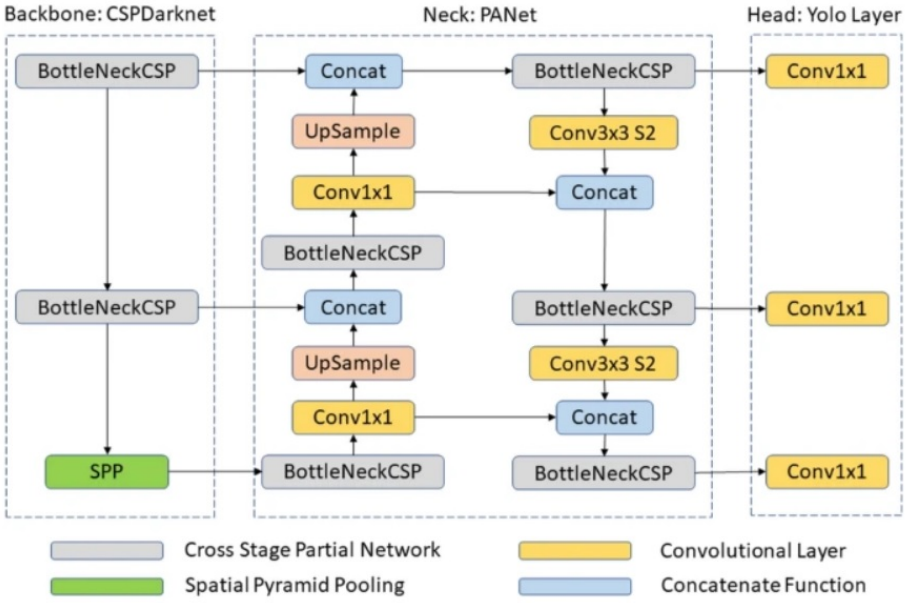


Fig. 5: YOLOv5 Architecture [23].

The reduction in model size is a standout feature of the fifth version of YOLO (YOLO-v5). Notably, the smallest YOLO-v5 model is 27MB, a substantial decrease from the 244MB size of YOLO-v4 on Darknet. YOLO-v5 also asserts superior accuracy and a higher frames-per-second performance compared to its predecessors.

4 Methodology Steps Description

This section offers an in-depth clarification of the evaluation methodology employed in this study. The research involves a comparative analysis of three deep models within the YOLO family: YOLOv3, YOLOv4, and YOLOv5. These models are single-stage detectors recognized for their fast, high-accuracy detection capabilities and are ideally suited for deployment on low-end systems, such as embedded platforms. The overall process of evaluation includes three main stages, as shown in Figure 6.

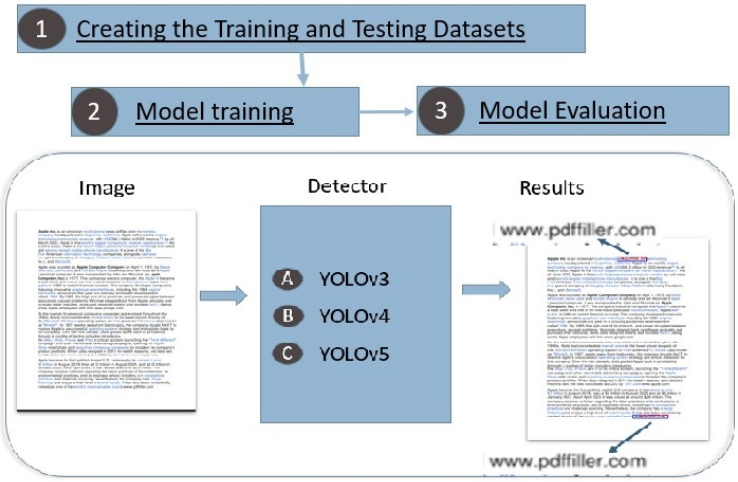


Fig. 6: Flowchart of the evaluation method.

Within this section, we outline the primary stage involving the creation and preparation of the training dataset. Subsequently, we delve into the training process for the three selected models. The section culminates with a series of tests conducted on images presenting various challenges, aimed at assessing the models' generalization aptitude across distinct usage scenarios featuring diverse content.

4.1 Creating and Preparing the Training and Testing Datasets

The chosen object detection algorithms are rooted in deep learning, and their intricate architecture necessitates training on specific datasets to attain the desired objectives. The dataset plays a pivotal role in influencing the models' performance; thus, it is imperative to have a robust dataset in order to achieve optimal performance.

In the context of this research, our focus lies on the application of various models for the detection of URLs. A challenge surfaced during the preliminary

phase of our study, as no standardized database was readily available for conducting a comprehensive evaluation and comparison of these models. In response, we undertook the task of creating our own dataset, comprising a total of 160 images, all of which feature URLs.

The URL starts with three consecutive letters 'w' and a dot, followed by a label. The label is a series of English letters from a to z (not case-sensitive) and can also contain digits from 0 to 9. Hyphens can be added, but not at the beginning or at the end, and adding more than one consecutively is not allowed. The label length is between 3 and 63 characters maximum. In the end, after a point, an extension is added. The most used extensions are "`.com`", "`.net`" and "`.org`". This part can be called the domain name.The URL can start with a protocol such as `http://`, but modern web clients like browsers automatically add the protocol before the URL if it doesn't contain one. The URL can also contain, after the extension name, more data, such as the filename `/index.html` or subdirectories like `/dir1/dir2` (see Figure 7).

Fig. 7: URL structure.

For our image dataset, we created random tag names according to the previously listed conventions with the defined extension added at the end, which is:`.com`, `.net`, `.org`, `.fr`, `.dz`, `.ca`, `.uk`. These extensions are widely spread, especially in our region. We added a few URLs with additional data at the end, but for the majority of images, we focused heavily on the domain name (label and extension).

Object detection techniques exclusively process pixel-level data, which implies that they perceive distinct variations between the letter 'A' in one font style and the same letter 'A' in a different font style. Moreover, there can be substantial disparities between a handwritten letter and its printed equivalent, despite both conveying the same semantic meaning through different visual representations (see Figure 8). So, as a starting point for creating the dataset, the printed URLs are written using the popular Arial font. and for color, black is chosen.

The majority of global printing is performed on A4 paper, and thus, our dataset will exclusively feature sample URLs that have been printed on A4 paper. These URLs will be juxtaposed with randomly generated text written in various languages, encompassing Latin, Arabic, Chinese, Russian, and Indian scripts.

After the generation of numerous simulated images, the next step consists of a manual labeling process. During this phase, each image's linked URL box is defined and manually set. Subsequently, these annotations are stored using

Fig. 8: The letter 'A' written by different fonts.

the YOLO image annotation format, wherein each image corresponds to an individual text annotation file, denoted by the same name as the image itself. Each line within the annotation file serves to define a ground-truth object present within the image, represented like this:

$$\text{“} < objectclass >< x >< y >< width >< height > \text{”}$$

This dataset format is compatible exclusively with Darknet-based versions of YOLO, namely YOLOv3 and YOLOv4, and is not compatible with YOLOv5. To accommodate YOLOv5, we adopted the Roboflow web platform, which serves as a comprehensive solution for hosting, annotating, and converting datasets across diverse formats.

4.2 Model training

The models are trained on 80 % (127 images) of all the data (160 images). The training is carried out without data augmentation in a Google Colab environment. The training parameters are:

- YOLOv3 utilizes input images set to dimensions of 416×416 during its 30-epoch training, spanning a total duration of 7 hours to achieve optimal weights.

- YOLOv4, on the other hand, configures input images at dimensions of 608×608 throughout its 30-epoch training, resulting in a total training duration of 7 hours to obtain the best weights.

- In the case of YOLOv5, the medium-sized model is selected, achieving a more effective balance between precision and speed. The input image dimensions are configured at 640×640 over the course of 30 epochs. Notably, in contrast to its predecessors, this model demonstrated exceptional speed by completing the training process in just 15 minutes, ensuring optimal weights.

Originally, YOLOv3 and YOLOv4 were implemented using the Darknet framework, a choice consistent with the historical development of YOLO versions. Darknet offers a lightweight and efficient implementation, making it well-suited for real-time object detection. This framework has been integral to earlier iterations of YOLO and has demonstrated effectiveness in various applications.

In contrast, YOLOv5 has undergone a shift in its implementation, now utilizing PyTorch, a widely adopted deep learning framework. PyTorch is renowned for its user-friendly nature, providing a seamless environment for research experimentation and development. The decision to transition to PyTorch for YOLOv5 was influenced by its flexibility, extensive community support, and robust capabilities in the realm of deep learning research.

Moreover, to mitigate overfitting concerns, we employed built-in regularization techniques such as L1 weight decay, incorporated transfer learning by utilizing pre-trained models on larger datasets, and implemented cross-validation.

Specifically, we employed k-fold cross-validation, where the dataset is partitioned into k (k=3) subsets, or folds, of equal size. The model is trained k times, each time using a different combination of k-1 folds for training and the remaining fold for validation. This process allows for a more robust evaluation of the model's performance, as it ensures that every data point is used for both training and validation across different iterations.

By averaging the performance metrics obtained from each fold, we obtain a more reliable estimate of the model's generalization ability, thereby reducing the risk of overfitting to the training data. This approach enables us to assess the stability and consistency of our models across different subsets of the data, providing valuable insights into their performance under varying conditions.

In all cases, official pre-trained weights are chosen to apply transfer learning. The final weights of the models are uploaded to the local machine to be used in the evaluation phase.

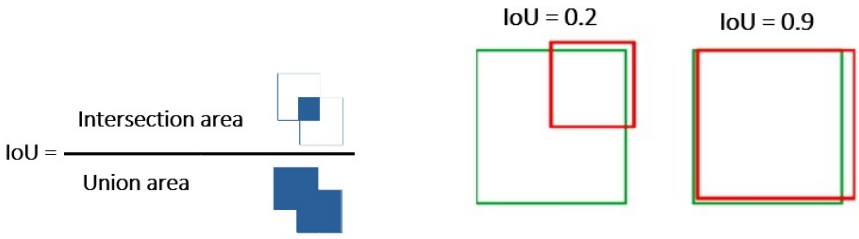
5 Model evaluation, results and discussion

The evaluation is conducted through a two-part process. In the initial stage, we assess the models' capacity for generalization in URL detection by considering their overall performance, which involves the utilization of the entire test dataset. In the subsequent stage, we subject the three models to testing under various conditions commonly encountered in photos taken with mobile phones, thereby evaluating their stability.

In the field of object detection, the evaluation of model performance relies on several crucial metrics that provide a comprehensive assessment of a model's object detection capabilities. Hereafter, we define the main metrics for object detection:

The Intersection over Union (IoU), also known as the Jaccard Index, quantifies the similarity between predicted bounding boxes and actual bounding boxes. Formally, IoU equals the intersection between the real and predicted bounding boxes divided by their union. Figure 9a clearly illustrates this concept of IoU.

IoU ranges from 0 to 1; the closer the actual and predicted bounding boxes, the closer the IoU measure is to 1 (see Figure 9b).



(a) Intersection over Union (IoU). (b) Examples of the IoU metric.

Fig. 9: Intersection over Union (IoU) metric.

Precision and recall. Precision assesses the proportion of correct predictions among all positive predictions, while recall measures the proportion of true positives identified among all actual objects.

Further, the precision-recall curve illustrates the trade-off between precision and recall for different confidence thresholds, providing an overall view of the model's performance across a range of confidence thresholds.

Another way to compare the performance of object detectors is by calculating the Average Precision (AP), a numerical measure corresponding to the area under the Precision-Recall curve. In practical terms, the AP serves as a metric that combines precision and recall, providing a summary of the Precision-Recall curve by averaging precision values across the recall range from 0 to 1. The computation of AP follows the formula:

$$AP = \Sigma_n (R_n - R_{n-1}) \times P_n \tag{1}$$

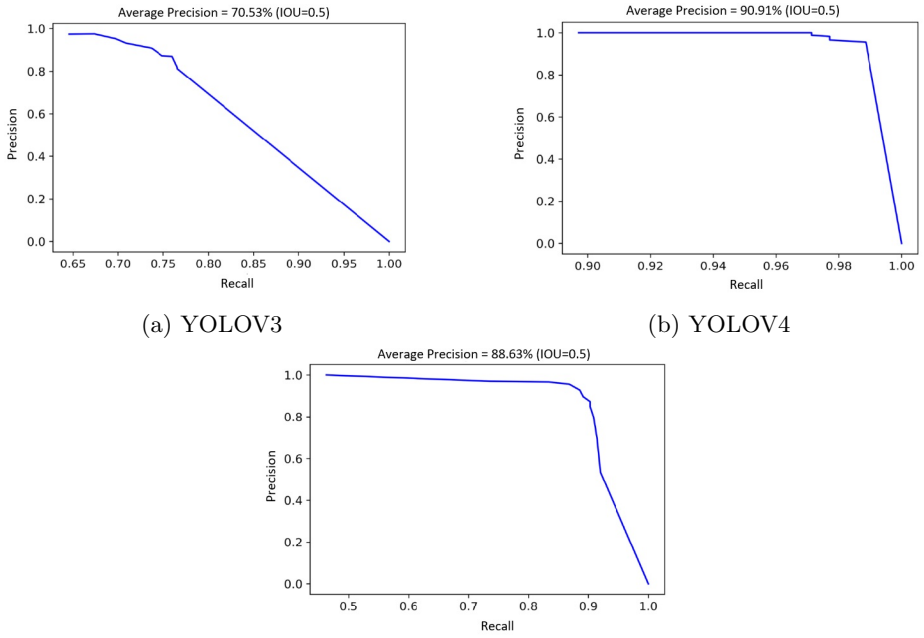
Where:

- (R_n) is the recall at position n in the sorted list of proposals
- (P_n) is the corresponding precision at (R_n).

The formula involves the summation over distinct recall levels (R_n) in the sorted list of proposals. For each recall level, the corresponding precision (P_n) is multiplied by the change in recall $(R_n - R_{n-1})$, and the results are summed. This formula captures the average precision across various recall levels and is commonly employed in assessing the performance of object detection models.

5.1 Generalization ability evaluation

The evaluation involves a subset of 20% of the complete dataset, comprising 33 images. We have selected an IoU threshold of 0.5 for this assessment. Results are depicted in Figure 10.



(a) YOLOV3

(b) YOLOV4

(c) YOLOV5

Fig. 10: Precision-Recall curve.

In the evaluation of our object detection models, distinct performance characteristics emerge. YOLOv3, for instance, achieved an average accuracy of **70.53%**. Notably, precision begins to decline once recall surpasses 75%.

Conversely, YOLOv4 demonstrates a notably higher average accuracy of **90.91%**, with a subsequent drop in precision observed after reaching a recall rate of 98%. As for YOLOv5, it achieves an average accuracy of **88.63%**, with precision exhibiting a decrease once recall exceeds 91%.

Despite the limited dataset size, the above observations underscore the model's ability to generalize effectively.

5.2 Evaluation in different conditions

In this section, we assess the model's performance using images that present challenges not encountered in the training dataset. These challenges include:

- Distinct typestyles, such as Algerian, Bradley Hand ITC, and Jokerman.
- Different background colors and character colors.
- Rotation of images of 90° and 180°.
- URLs prefixed with the `https://` protocol tag.
- Handwritten URL characters.
- Images with Gaussian Noise.

The table below showcases the models' performance as measured by the average precision (AP) for each difficulty category.

Table 1: Average Precision (AP) of YOLOv3, YOLOv4, and YOLOv5 for different difficulties

	Model		
Difficulty	***YOLOv3***	***YOLOv4***	***YOLOv5***
Colors	31.17%	54.55%	27.27%
Font	72.73%	84.29%	88.07%
HTTP Protocol	12.12%	28.9%	13.64%
Handwritten characters	0.0%	0.0%	0.0%
Rotation (180°)	27.27%	18.18%	18.18%
Rotation (90°)	0.0%	0.0%	0.0%
Gaussian noise	36.36%	37.36%	83.98%

From results presented in Table 1, we can draw the following conclusions:

Testing with distinct typestyles: This challenge had a relatively minor impact on the models' performance, as all three models demonstrated closely aligned average accuracies. YOLOv3 yielded the lowest average accuracy of 72.7%, while YOLOv5 achieved the highest average accuracy of 88.07%, and YOLOv4 84.2%. Furthermore, Figures 11, 12, and 13 provide a comprehensive representation of URLs successfully identified within text samples rendered in three different fonts.

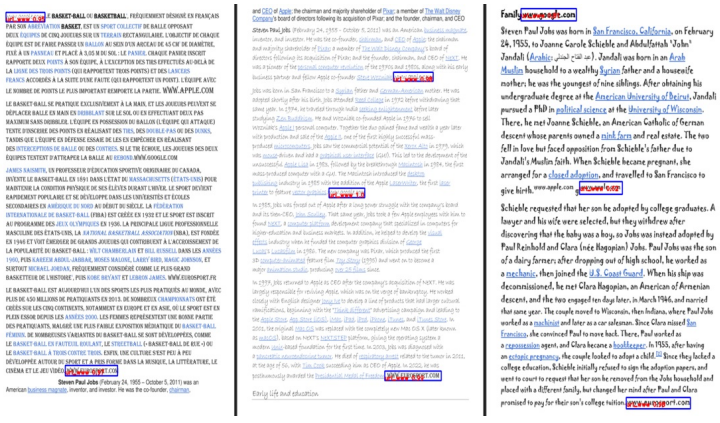


Fig. 11: Different character polices (YOLOV3).

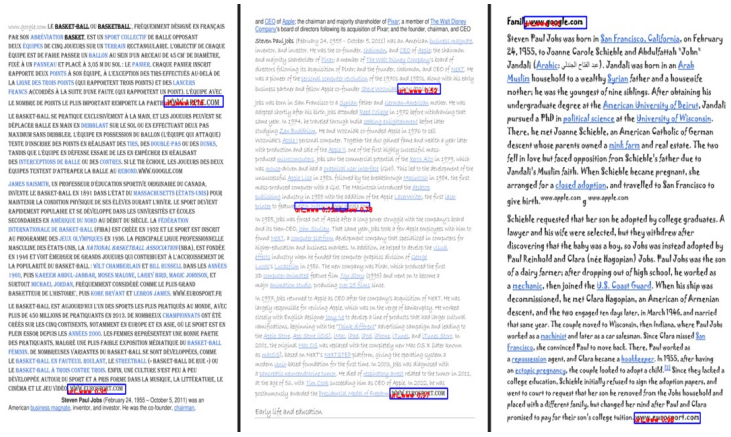


Fig. 12: Different character polices (YOLOV4).

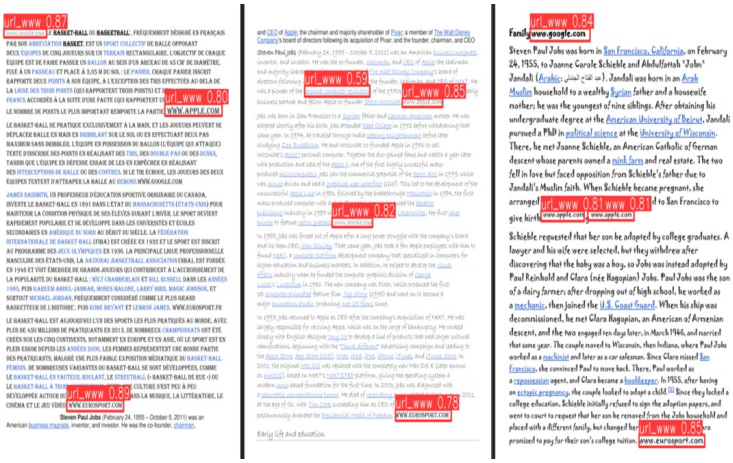


Fig. 13: Different character polices (YOLOV5).

Testing with different color polices and backgrounds: This challenge posed a significant impact on the models' performance. YOLOv3, for instance, exhibited the inability to detect URLs against a colored background, yet it demonstrated success in detecting URLs with colored characters, achieving an average accuracy of 31.71%. In contrast, YOLOv4 emerged as the top-performing model in this context, attaining an average accuracy of 54.44%. YOLOv4 managed to successfully detect all URLs with colored characters and a majority of URLs against colored backgrounds. Conversely, YOLOv5 displayed the weakest performance with an average accuracy of 27.27%, struggling to detect URLs with colored objects, as well as those against colored backgrounds. Additionally, Fig-

ures 14, 15, and 16 provide a visual representation of the identification of URLs within text samples that feature colored characters and are placed against colored backgrounds.

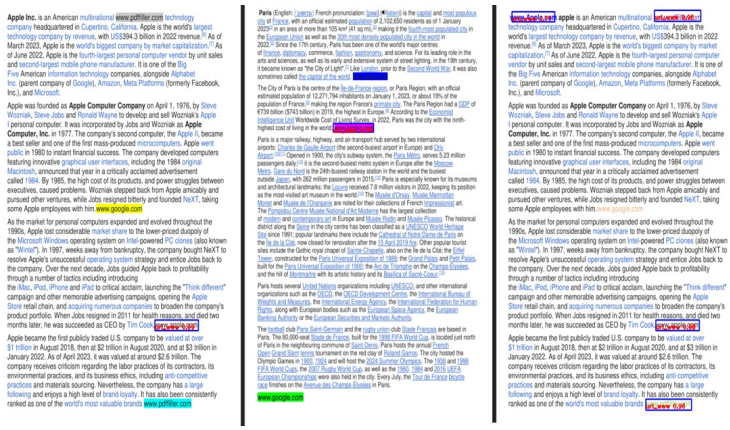


Fig. 14: Different color polices and backgrounds (YOLOV3).

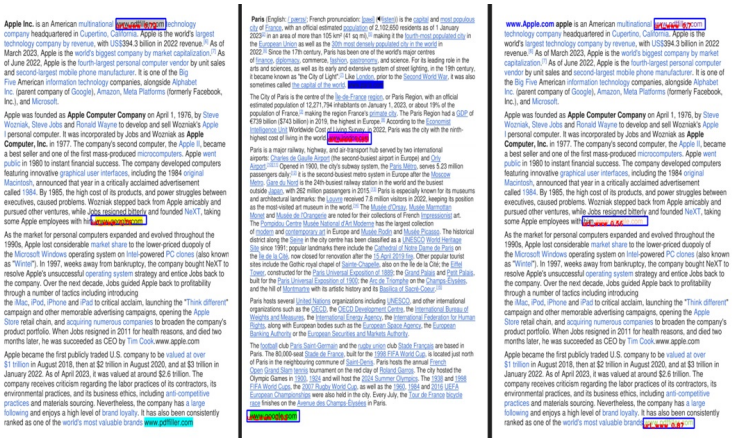


Fig. 15: Different color polices and backgrounds (YOLOV4).

Testing with different image rotations (90° and 180°): Rotating the images by 180° had no notable influence on the models' performance. However, when rotated by 90°, the models' performance was significantly decreased, with

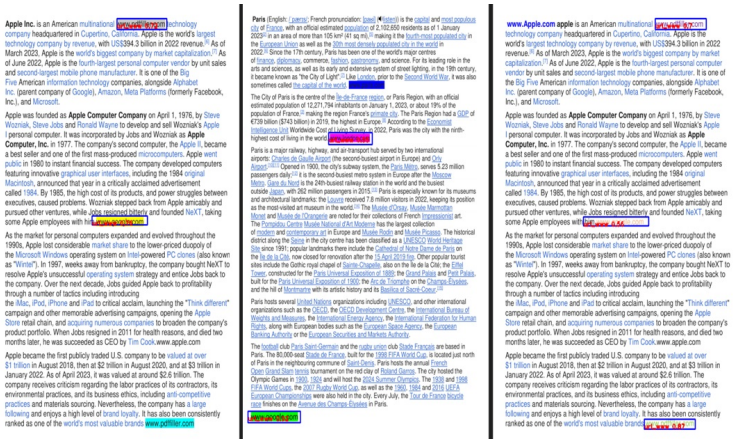


Fig. 16: Different color polices and backgrounds (YOLOV5).

all three models achieving an average accuracy of 0%. As evident in Figures 17, 18, and 19.

Fig. 17: Image rotations -90°, 180° (YOLOV3).

Fig. 18: Image rotations -90°, 180° (YOLOV4).

Fig. 19: Image rotations -90°, 180° (YOLOV5).

URL prefixed with the http:// protocol tag: In the case of URLs prefixed with the http:// tag, the models encountered difficulties in their detection, with some models failing to identify the complete URL, recognizing only the domain name. YOLOv3 exhibited an average accuracy of 12.12%, YOLOv4 outperformed the others with the highest accuracy at 28.9%, and YOLOv5 achieved an accuracy of 13.64%. Examples of detections for this particular challenge are illustrated in Figures 20, 21 and 22.

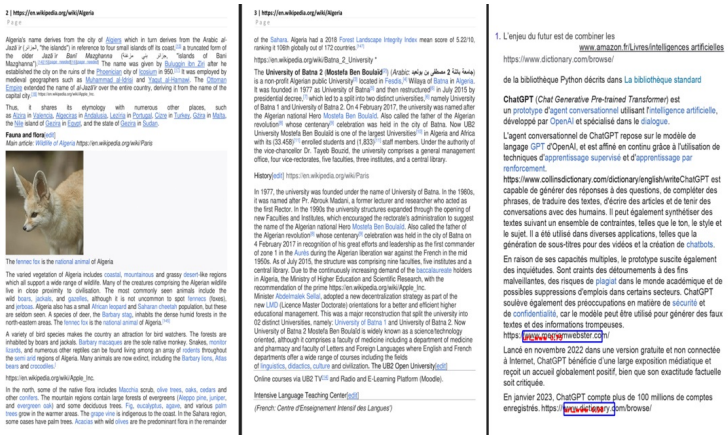


Fig. 20: URL prefixed with the `http://` protocol tag (YOLOV3).

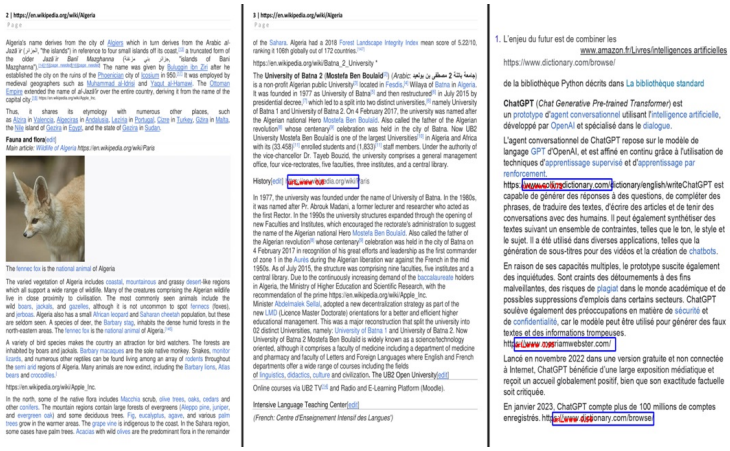


Fig. 21: URL prefixed with the `http://` protocol tag (YOLOV4).

Handwritten URL characters: In this case, all models were unable to identify the URL, resulting in an average accuracy of 0% across the board. Figure 23 provides an image depicting a sheet with various handwritten URLs that remained unrecognized by all three models.

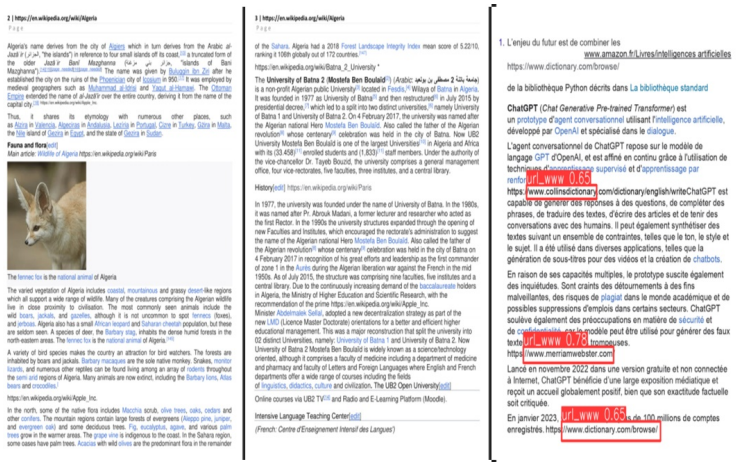


Fig. 22: URL prefixed with the `http://` protocol tag (YOLOV5).

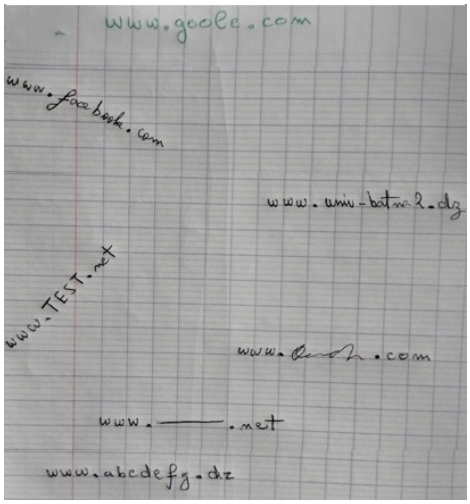


Fig. 23: Handwritten URL characters (YOLOv3, YOLOv4, and YOLOv5).

Gaussian noise addition: The introduction of noise had a profound impact on the models' accuracy, leading to the detection of false objects. YOLOv3's accuracy dropped to 36.36%, and YOLOv4 also experienced a decrease, with an accuracy of 37.36%. In contrast, YOLOv5 achieved the highest accuracy of 83.98% under these conditions (see Figures 24, 25, and 26).

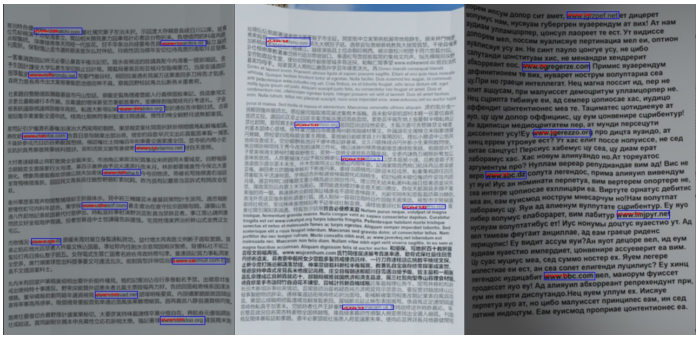


Fig. 24: Images with Gaussian noise (YOLOv3).

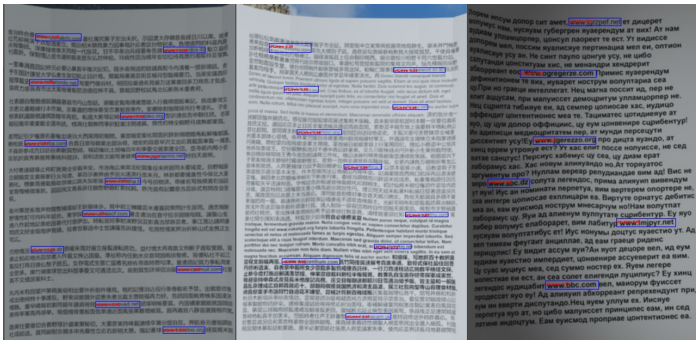


Fig. 25: Images with Gaussian noise (YOLOv4).

6 Conclusion

In this study, we investigated a very interesting topic in the field of computer vision, specifically focusing on object detection, a pivotal stage in recognition processes. Our primary goal was to assess the generalization capability and robustness of three distinct models—YOLOv3, YOLOv4, and YOLOv5—in the context of URL detection within mobile phone-captured images.

- The experimental results, expressed in terms of average precision, allowed us to deduce the following conclusions:
- The three models gave very satisfactory generalization results, and the best is YOLOv4.

Concerning stability for several difficulties, the 3 models did not completely recognize URLs rotated by a 90° rotation angle, where the average precision achieved is 0:0%. Also, for handwritten URLs, all three models provided an average accuracy of 0.0%.

To improve these results, we propose to:

- increase the size of the dataset.

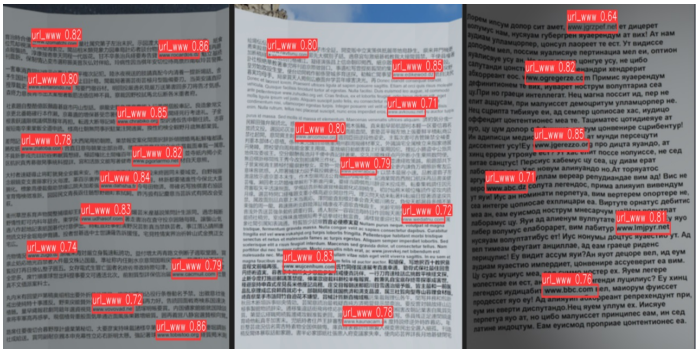


Fig. 26: Images with Gaussian noise (YOLOv5).

– Augment the image set with images containing different difficulties for the training dataset.
– Test other versions of the YOLO family, even other models of the two-stage detector family.

References

1. Zuguo Chen, Yanglong Liu, Chaoyang Chen, Ming Lu, and Xuzhuo Zhang. Malicious url detection based on improved multilayer recurrent convolutional neural network model. *Security and Communication networks*, 2021:1–13, 2021.
2. Jianming Hu, Xiyang Zhi, Tianjun Shi, Wei Zhang, Yang Cui, and Shenggang Zhao. Pag-yolo: A portable attention-guided yolo network for small ship detection. *Remote Sensing*, 13(16):3059, 2021.
3. Yankai Ma, Jun Yang, Zhendong Li, and Ziqiang Ma. Yolo-cigarette: An effective yolo network for outdoor smoking real-time object detection. In *2021 Ninth International Conference on Advanced Cloud and Big Data (CBD)*, pages 121–126. IEEE, 2022.
4. Zhao Feng, Wang Jianzong, and Xiao Jing. Yolo-based image target recognition method and apparatus, electronic device, and storage medium. `https://patents.google.com/patent/WO2020164282A1/en`.
5. Jing Wang, Peng Yang, Yuansheng Liu, Duo Shang, Xin Hui, Jinhong Song, and Xuehui Chen. Research on improved yolov5 for low-light environment object detection. *Electronics*, 12(14):3089, 2023.
6. Michael Shenoda. Lighting and rotation invariant real-time vehicle wheel detector based on yolov5. `https://arxiv.org/pdf/2305.17785`, 2023.
7. Baokai Liu, Fengjie He, Shiqiang Du, Jiacheng Li, and Wenjie Liu. An advanced yolov3 method for small object detection. *Journal of Intelligent & Fuzzy Systems*, (Preprint):1–13, 2022.
8. Object detection for blind people using yolov3. *International Journal For Science Technology And Engineering*, 11(5):7172–7181, 2023.
9. Renduchinthala Sai Praneeth, Kancharla Chetan Sai Akash, Bommisetty Keerthi Sree, P Ithaya Rani, and Abhishek Bhola. Scaling object detection to the edge with yolov4, tensorflow lite. In *2023 7th International Conference on Computing Methodologies and Communication (ICCMC)*, pages 1547–1552. IEEE, 2023.

10. Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, and Jose Garcia-Rodriguez. A review on deep learning techniques applied to semantic segmentation. `https://arxiv.org/pdf/1704.06857`, 2017.
11. Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. *Proceedings of the IEEE*, 2023.
12. Paul Viola and Michael 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*, volume 1, pages I–I. Ieee, 2001.
13. Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.
14. Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2009.
15. Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.
16. Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. `https://arxiv.org/pdf/1804.02767`, 2018.
17. Seokyong Shin, Hyunho Han, and Sang Hun Lee. Improved yolov3 with duplex fpn for object detection based on deep learning. *The International Journal of Electrical Engineering & Education*, page 0020720920983524, 2021.
18. Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. `https://arxiv.org/pdf/2004.10934`, 2020.
19. Chien-Yao Wang, Hong-Yuan Mark Liao, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh, and I-Hau Yeh. Cspnet: A new backbone that can enhance learning capability of cnn. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 390–391, 2020.
20. Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8759–8768, 2018.
21. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015.
22. Glenn Jocher. Ultralytics yolov5. `https://github.com/ultralytics/yolov5`, 2020.
23. Yiming Fang, Xianxin Guo, Kun Chen, Zhu Zhou, and Qing Ye. Accurate and automated detection of surface knots on sawn timbers using yolo-v5 model. *BioResources*, 16(3):5390, 2021.