A Survey on ”You Only Look Once(YOLO): Unified, Real-Time Object Detection” and Comparative Evaluation of Machine Learning Techniques

Abhirup Ranjan, Niharika Khurana,Viutika Rathod

Advisor: Dr. Adel Abusitta

Abstract

You Only Look Once: Unified, Real-Time Object Detection, a study by Redmon, J., Divvala, S., Girshick, R., and Farhadi that was published in the 2016 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, is thoroughly examined in this survey paper. The purpose of the essay is to critically analyze and summarize the original work's strengths and weaknesses. Additionally, it compares various machine learning methods, implements the YOLO algorithm from the research article, and suggests a fresh strategy to overcome current drawbacks. This paper aims to contribute to a better understanding of real-time object detection and its developments in the field of computer vision by providing a thorough survey.

Keywords: You Only Look Once(YOLO), Real-Time Object Detection, Computer vision, Strengths, Weaknesses

1. Introduction

You Only Look Once: Unified, Real-Time Object Detection, a study by Redmon, J., Divvala, S., Girshick, R., and Farhadi (1) that was published in the 2016 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, is thoroughly examined in this survey paper. The purpose of the essay is to critically analyze and summarize the original work's strengths and weaknesses. Additionally, it compares various machine learning methods, implements the YOLO algorithm from the research article, and suggests a fresh strategy to overcome current drawbacks. This paper aims to contribute to a better understanding of real-time object detection and its developments in the field of computer vision by providing a thorough survey.

*1.1. **Background and Significance***

The fundamental computer vision task of real-time object detection has numerous applications, including video surveillance, autonomous vehicles, and robotics. Canny frameworks should have the option to recognize protests precisely and continuously to understand and collaborate with their environmental elements. Over time, a variety of strategies have been developed to address this issue, with advances in machine learning being particularly significant. ”You Only Look Once:” is a study by Redmon et al. 's ”Unified, Real-Time Object Detection” 2016) has gotten a ton of consideration in the field among the different techniques. This review fostered the Just go for its calculation, which consolidated object discovery and characterization into a solitary relapse issue. Real-time object recognition was revolutionized by this algorithm. For real-time applications, the YOLO algorithm is a popular choice due to its remarkable speed without sacrificing accuracy.

2. Overview of ”You Only Look Once(YOLO): Unified, Real-Time Object Detection”

*2.1. **Introduction to the YOLO Algorithm***

The YOLO (You Only Look Once) algorithm is a groundbreaking method for object detection and computer vision. It was first presented in the research paper ”You Only Look Once: Unified, Real-Time Object Detection” by Redmon, J., Divvala, S., Girshick, R., and Farhadi, A (1). By proposing a unified framework that combines object localization and classification into a single regression problem, it revolutionized real-time object detection. This novel method significantly advances the field by enabling faster and more effective object detection in real-time applications.

*2.2. **Key Concepts and Methodologies***

The YOLO (You Only Look Once) algorithm, incorporates several important concepts and methodologies that enhance its efficiency in real-time object detection. Its main features are as follows:

*2.2.1. **Unified Detection***

The YOLO algorithm uses a unified strategy to detect objects and classify them in the same regression problem. Instead of using multiple stages or region proposals like conventional methods do, YOLO predicts bounding boxes and class probabilities in one pass, which speeds up inference times.

*2.2.2. **Grid-based Division***

YOLO divides the input image into a grid of cells, and each cell oversees forecasting the bounding boxes and related class probabilities for the objects situated in its area. YOLO can effectively handle objects of various scales and aspect ratios thanks to this grid-based division.

2.2.3. Bounding Box Prediction

For each grid cell, YOLO predicts a few bounding boxes, their confidence scores, and their class probabilities. The algorithm uses anchor boxes, which are pre-defined bounding box shapes of varying sizes, to improve localization accuracy.

2.2.4. Feature Extraction

The YOLO algorithm relies on deep convolutional neural networks (CNNs) for feature extraction. The network extracts high-level features from the input image for object detection and classification.

2.2.5. Loss Function

YOLO makes use of a specialized loss function that combines the loss of class probability, confidence, and localization. The algorithm can optimize for precise class probabilities, confident object detections, and precise bounding box predictions thanks to this multi-component loss function.

2.2.6. Non-Maximum Suppression

YOLO employs non-maximum suppression to reduce redundant detections and enhance localization accuracy. Only the most reliable and non-overlapping predictions are kept after this post-processing step removes bounding boxes that overlap.

3. Comparative Evaluation of Machine Learning Techniques

3.1. Implementation of YOLO Algorithm

When comparing different machine learning techniques in implementing the YOLO algorithm, we can consider how easy they are to use and how well they work. Some techniques require more coding and customization, while others provide ready-made tools and libraries that make implementation simpler.

3.1.1. Data Preprocessing

Different techniques may have different ways of preparing the data for YOLO. Some may require manually drawing boxes around objects, while others have automated tools or use special software to help with this. The quality and accuracy of the data preparation can affect how well the YOLO algorithm performs.

3.1.2. Model Architecture

Machine learning techniques can be used for different designs for the YOLO model. These designs can vary in complexity and may include extra features to improve performance. The choice of model architecture can impact how well the YOLO algorithm can detect objects quickly and accurately.

3.1.3. Training Process

Teaching the YOLO algorithm to recognize objects is part of the training phase. This may be accomplished using a variety of strategies, including changing the learning rate or expanding the training set with additional data. The algorithm's final performance may vary depending on how it was trained.

3.1.4. Inference Methodologies

Different strategies may operate more quickly and efficiently than others when the YOLO algorithm is really used. To analyze new photos or movies, some techniques might be quicker or require less processing resources. The algorithm's ability to recognize things in real time can vary depending on how it is applied.

In conclusion, when evaluating machine learning techniques for YOLO, we need to consider how simple they are to use, how well the data is prepared, the model's design, the training process, and how quickly and effectively they work in practice. The effectiveness of the YOLO method in real-world applications is influenced by these variables.

3.2. Performance Evaluation Metrics

3.2.1. Accuracy

A key performance indicator used to evaluate the object detection system's overall effectiveness is accuracy. It calculates the percentage of accurately classified objects among all the objects that were found. When using the YOLO algorithm, accuracy is typically determined by comparing the predicted bounding boxes to the ground truth bounding boxes while taking intersection over union (IoU) threshold into consideration.

Accuracy Calculation By comparing the predicted and actual bounding boxes and gauging the degree of overlap using an IoU (Intersection over Union) threshold, accuracy is typically calculated. The area of intersection between the predicted and ground truth bounding boxes is divided by the area of their union to calculate the IoU. The predicted bounding box is regarded as a true positive if the IoU exceeds the threshold; otherwise, it is a false positive.

3.2.2. False Positive Rates

The rate at which the system incorrectly detects objects that are not present in the image is referred to as the false positive rate (FPR). It is determined by separating the complete number of negative examples by the all-out number of misleading upsides. False positives are bounding boxes that are mistakenly identified as objects in object detection. The algorithm's capacity to reduce unnecessary computational load and eliminate false detections is evaluated using the false positive rate.

False Positive Rate Calculation The number of false positives (bounding boxes that were incorrectly detected) is divided by the total number of negative samples (bounding boxes that did not contain objects) to determine the false positive rate (FPR). The algorithm's capacity to reduce false detections is assessed with the aid of FPR.

3.2.3. Other Relevant Metrics

F1 Rating The harmonic mean of recall and precision is the F1 score. It is especially useful when the dataset is imbalanced because it provides a balanced measure that considers both metrics.

Precision and Recall Calculation By dividing the total number of true positives (bounding boxes that were correctly detected) by the total number of true positives and false positives, precision is calculated. It evaluates how well the optimistic predictions turned out. Recall is calculated by dividing the total of true positives and false negatives (missed detections) by the number of true positives. Recall gauges how well an algorithm can locate.

Mean Average Precision (mAP) Calculation mAP is commonly computed by calculating the average precision (AP) for each class and then taking the mean across all classes. AP is calculated by plotting the precision-recall curve and integrating the area under the curve. It provides an overall assessment of the algorithm's performance across different object categories.

4. Advantages and Disadvantages of the YOLO Algorithm

4.1. Advantages

4.1.1. Simplicity and Efficiency

The YOLO algorithm offers simplicity and efficiency in object detection. It utilizes a single neural network to directly predict bounding boxes and class probabilities for objects in an image. This streamlined architecture simplifies the detection process and results in faster inference times compared to other object detection algorithms.

4.1.2. Real-Time Performance

One of the major strengths of the YOLO algorithm is its real-time performance. By optimizing the architecture and employing efficient techniques such as grid-based spatial partitioning, YOLO achieves fast processing speeds, enabling it to perform object detection in real-time applications such as video surveillance, autonomous driving, and robotics

4.1.3. Unified Approach

The YOLO algorithm takes a unified approach to object detection by simultaneously performing object localization and classification. Instead of separating these tasks into multiple stages, YOLO combines them into a single end-to-end process. This unified approach leads to better overall performance and simplifies the implementation of object detection systems.

4.2. Disadvantages

4.2.1. Localization Accuracy

One limitation of the YOLO algorithm is its lower localization accuracy compared to some other object detection methods. YOLO predicts bounding boxes based on predefined anchor boxes, which may result in less precise localization, especially for small objects or objects with complex shapes.

4.2.2. Detection of Small Objects

The YOLO algorithm faces challenges in accurately detecting and localizing small objects. Due to the nature of its grid-based approach, YOLO may struggle to detect objects that have a small visual footprint or limited spatial context. This limitation can impact the algorithm's performance in scenarios where small objects are of particular importance.

4.2.3. Lack of Contextual Information

Another drawback of YOLO is its limited ability to consider contextual information surrounding objects. Since YOLO analyzes objects independently within each grid cell, it may miss out on valuable contextual cues that could aid in accurate detection and classification. This lack of contextual information can lead to reduced performance, especially in complex scenes or cases where object discrimination relies on contextual understanding.

4.2.4. Imbalanced Class Handling

YOLO may face challenges in handling imbalanced class distributions effectively. If the dataset used for training contains significant class imbalances, where certain object classes have a much smaller representation compared to others, the algorithm's performance may be biased towards the dominant classes. This imbalance can result in reduced accuracy for underrepresented classes during object detection.

While the YOLO algorithm offers several advantages, such as simplicity, efficiency, real-time performance, and a unified approach, it also has limitations in terms of localization accuracy, detection of small objects, lack of contextual information, and imbalanced class handling. These considerations are important when applying the YOLO algorithm to specific use cases, and further research and enhancements can be pursued to address these limitations.

4.3. Results and Comparative Analysis

4.3.1. Performance Comparison with Original Paper

The performance of the implemented techniques will be compared to the original research article's findings in this section. The performance can be evaluated using metrics like accuracy, false positive rates, precision, recall, and mAP. The comparison will highlight any similarities or differences in the YOLO algorithm's performance when applied to various datasets or settings. It will shed light on the algorithm's reproducibility and generalizability of results.

4.3.2. Strengths and Limitations of Implemented Techniques

We'll take into account the advantages and drawbacks of each method. Better management of explicit issues like small article discovery and uneven classes, increased accuracy, and lower rates of false positives and negatives that are misleading are among the benefits. These advantages will be emphasized to emphasize the improvements in execution. Then again, we'll discuss the techniques' disadvantages, for example, that they are so hard to utilize, that they are so delicate to limit settings, and that overseeing specific sorts of articles is so troublesome. A comprehensive comprehension of the methodologies' benefits and drawbacks will be provided by the analysis.

5. Proposed Technique

5.1. Introduction to the Proposed Technique

One alternative to the YOLO algorithm that has gained significant attention and shown promising results is the EfficientDet algorithm. EfficientDet(2) is an advanced object detection algorithm that focuses on achieving a better trade-off between accuracy and computational efficiency.

EfficientDet is based on a scalable and efficient compound scaling technique that optimizes both the depth and width of the network architecture. By systematically scaling up the model, EfficientDet achieves higher accuracy compared to YOLO while maintaining computational efficiency.

5.2. *Implementation and Performance Evaluation*

EfficientNet introduces a family of CNN architectures, denoted as EfficientNet-B0, B1, B2, and so on. Each architecture within the EfficientNet family is defined by a scaling factor that uniformly scales the depth, width, and resolution of the network.

The scaling factor, denoted as "phi" (ϕ), controls the level of network expansion. A higher value of phi corresponds to a larger and more powerful network, while a lower value corresponds to a smaller and more compact network.

EfficientNet-B0 serves as the base architecture, and subsequent variants, such as B1, B2, B3, etc., progressively scale up the network. The scaling is achieved by applying compound scaling to the depth, width, and resolution dimensions of the network.

Specifically, the depth of the network is scaled by a factor of alpha (α), the width (number of channels) is scaled by a factor of beta (β), and the input image resolution is scaled by a factor of gamma (γ). These scaling factors are interrelated to maintain a balance between network capacity and computational efficiency.

EfficientNet-B0 represents the baseline model with modest depth, width, and resolution. As the model index increases (B1, B2, B3, and so on), the network becomes deeper, wider, and operates at higher resolutions, enabling it to capture more complex features and improve performance.

The compound scaling technique ensures that the depth, width, and resolution are scaled uniformly while maintaining a balance between model capacity and computational efficiency. This scaling strategy allows EfficientNet to achieve state-of-the-art performance with optimal resource utilization.

It's important to note that the specific scaling factors and architectural details may vary between implementations and versions of EfficientNet. However, the general concept of uniform scaling across depth, width, and resolution remains consistent.

When integrating EfficientNet into YOLO or any other algorithm, the choice of the EfficientNet variant (e.g., B0, B1, B2) depends on the desired trade-off between accuracy and computational efficiency. Higher variants provide greater capacity but require more computational resources, while lower variants offer reduced complexity but may sacrifice some performance.

Experimentation and evaluation with different EfficientNet variants are essential to identify the optimal configuration that aligns with the requirements and constraints of the specific object detection task.

Some advantages of EfficientDet over YOLO include: Improved Accuracy: EfficientDet has demonstrated improved performance and higher accuracy compared to YOLO on various benchmark datasets(3). It achieves state-of-the-art results in terms of mean Average Precision (mAP) while considering different object scales and aspect ratios.

Efficient Architecture: EfficientDet is designed to achieve better computational efficiency compared to YOLO. It optimizes the model's depth and width, allowing for faster inference times and reduced memory requirements.

Scalability: The compound scaling technique in EfficientDet enables the algorithm to adapt to different resource constraints and target performance requirements. It can be efficiently scaled up or down to accommodate different computing platforms.

Flexibility: EfficientDet supports various backbone architectures, such as EfficientNet, to extract high-level features effectively. This flexibility allows for better adaptation to different datasets and domains.

While EfficientDet offers advantages over YOLO, it is important to note that the choice of algorithm depends on the specific requirements of the application and the trade-off between accuracy and computational efficiency(4). Conducting a comparative evaluation between YOLO and EfficientDet, considering specific datasets and evaluation metrics, can provide valuable insights into their respective strengths and weaknesses.

5.3. *Future Research Directions*

To incorporate EfficientNet into the YOLO algorithm, several changes need to be made to adapt the YOLO architecture and leverage the capabilities of EfficientNet. Here are some key modifications:

Replace the Backbone Network: The first step is to replace the original backbone network of YOLO with the EfficientNet architecture. EfficientNet's convolutional layers will serve as the feature extraction backbone for YOLO. This change involves integrating EfficientNet's blocks, such as convolutional layers, pooling layers, and skip connections, into the YOLO architecture.

Input Resolution Adjustment: EfficientNet operates at different input resolutions compared to YOLO. Adjust the input resolution of YOLO to match the input size expected by the selected EfficientNet variant. This ensures that the input resolution is consistent with EfficientNet's training and inference requirements.

Anchor Box Design: The anchor boxes used in YOLO may need to be adjusted to align with the different scales and aspect ratios that EfficientNet is optimized for. The anchor boxes should be chosen or generated based on the scales and ratios appropriate for the feature maps produced by the EfficientNet backbone.

Feature Pyramid Integration: EfficientNet employs a feature pyramid network (FPN) to capture multi-scale information. Integrate the FPN concept into YOLO to ensure that the detection layers in YOLO receive features from multiple scales. This allows YOLO to better handle objects of varying sizes and improve detection performance.

Training and Fine-tuning: Initialize the EfficientNet backbone with pre-trained weights, such as those obtained by training EfficientNet on a large-scale image classification dataset like ImageNet. Fine-tune the combined YOLO-EfficientNet model on object detection datasets using appropriate loss functions and optimization techniques.

Hyperparameter Tuning: Some hyperparameters in YOLO, such as learning rate, regularization, and augmentation strategies, may need adjustment to achieve optimal performance when combined with EfficientNet. Experimentation and tuning are necessary to determine the ideal hyperparameters for the integrated YOLO-EfficientNet model.

Evaluation and Comparison: Conduct a thorough evaluation of the integrated YOLO-EfficientNet model using appropriate evaluation metrics, comparing its performance to the original YOLO algorithm. Assess improvements in accuracy, computational efficiency, and handling of various object scales to determine the benefits of incorporating EfficientNet.(5)

It's important to note that the specific implementation details and adjustments may vary depending on the frameworks used and the specific EfficientNet variant chosen. Consult the literature, official implementation repositories, or relevant code examples for more detailed guidance on integrating EfficientNet into the YOLO algorithm.

6. Conclusion

6.1. *Summary of the Survey*

In order to achieve unified, real-time object detection, this survey paper carefully analyzes the You Only Look Once (YOLO) algorithm. It implements YOLO, critically assesses the algorithm's advantages and disadvantages, contrasts it with other machine learning techniques, and suggests a plan to get around its drawbacks. By fusing object localization and classification into a single regression problem, the YOLO algorithm revolutionized real-time object detection by providing simplicity, efficiency, and real-time performance. However, it has issues with class handling, small object detection, contextual information, and localization accuracy. This survey suggests using the EfficientDet algorithm as an alternative because it uses scalable and effective compound scaling techniques to achieve improved accuracy and computational efficiency.

6.2. *Key Findings and Contributions*

YOLO algorithm: The YOLO algorithm achieved real-time performance and simplicity by introducing a unified framework for object detection.(1)
Limitations Of YOLO: Yolo's shortcomings include difficulties with accurate localization, detecting small objects, taking into account contextual information, and handling unequal class distributions.
This study suggests using EfficientDet(2) as an alternative because it combines EfficientNet and YOLO and offers increased accuracy, computational efficiency, scalability, and flexibility.YOLO's backbone network must be replaced, input resolutions must be changed, anchor boxes must be redesigned, and feature pyramid integration must be added.For optimum performance, the YOLO-EfficientNet model must be trained with suitable loss functions and undergo hyperparameter tuning.

6.3. *Closing Remarks*

This research paper offers a thorough analysis of the YOLO algorithm, points out its drawbacks, and suggests using EfficientDet as a promising substitute. The survey makes a contribution to the field of real-time object detection by addressing the shortcomings of YOLO and utilizing the developments of EfficientDet. The difficulties with localization accuracy, small object detection, contextual understanding, and imbalanced class handling are highlighted, underlining the need for more research and development in computer vision. The survey's findings are intended to inform future developments in real-time object detection algorithms and advance knowledge in the area.

References

[1] D. S. G. R. . F. A. Redmon, J., "You only look once: Unified, real-time object detection. in proceedings of the ieee conference on computer vision and pattern recognition," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, 2016.

[2] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10781–10790, 2020.

[3] J. S. May 1 and M. Read, "Yolov3 versus efficientdet for state-of-the-art object detection," 2023.

[4] DagsHub Blog, "Yolov6: next generation object detection - review and comparison," 2022.

[5] Kaggle, "Efficientdet inference (better than yolov5)," 2023.