


Efficacy and Efficiency in Object Detection: Exploring YOLOv9 on Limited Resources

Yangkai Zhou ^a

College of Intelligence and Computing, Tianjin University, Tianjing, China

Keywords: Deep Learning, Object Detection, YOLO, Performance Testing, Network Architecture.

Abstract: The introduction of the You Look Only Once v9 (YOLOv9) algorithm marks a significant milestone in the realm of object detection, notably tackling the inherent information bottleneck in deep learning while enhancing model accuracy and parameter efficiency across various tasks. This advancement translates into expedited and more precise detection capabilities, surpassing its predecessor, the YOLO algorithm, as well as other modelling methodologies. This paper endeavours to delve into the performance evaluation of YOLOv9 and explore strategies for effectively training the model on small datasets and with limited computational resources. Through meticulous hyperparameter tuning, careful selection of optimizers, and comparison between YOLOv9-c and YOLOv9-e model variants, this study aims to ascertain the most suitable learning rates, batch sizes, and optimizers to optimize training efficacy. The insights garnered from this research serve as a valuable guide for small research teams and individuals facing computational constraints, providing them with a robust framework to streamline experimental processes and enhance overall efficiency in model development and training procedures. Ultimately, this paper contributes to advancing the accessibility and effectiveness of object detection methodologies in diverse settings and applications.


1 INTRODUCTION

A key task named object detection which contains Object detection is an important problem in the field of computer vision and one of the most challenging problems in the field of computer vision, which is widely used in many fields such as video monitoring, automatic piloting, face recognition, to name just a few. In the last decade, thanks to the rapid development of deep learning, significant progress has been made in the field of deep learning-based modelling computer vision, especially the development of the field of object detection has been applied in a large number of applications. Many powerful system architectures and learning methods have emerged, including Convolutional Neural Networks (CNNs) (Huang, 2017), Transformers (Dosovitskiy, 2020; Tu, 2022) and Mambas (Liu, 2024).

In order to improve the model performance, the loss function (Chen, 2020; Rezatofighi, 2019) and the label assignment strategies (Wang, 2021) is proposed to increase the generalization of the objective

function. These efforts aim to accurately map the input data to the target task.

However, traditional neural network models usually have a large number of parameters, which can lead to the need for deeper networks often when dealing with large-scale datasets, where the number of network parameters rises dramatically with the size of the dataset. The computational complexity of the model will thus increase dramatically, and it is also prone to overfitting. And for long-distance-dependent tasks, traditional neural networks may face modeling difficulties. Especially when dealing with long sequence data, the model may lose the information related to long distances in the sequence. Further, most of the previous methods ignore the fact that there may be a large amount of information loss in the input data during the forward propagation process, and this information loss can lead to biased gradient flow, and updating the model on such a basis is likely to cause the deep network to establish incorrect correlations between the target and the inputs, which will make the training model produce incorrect predictions (Wang, 2024).

^a <https://orcid.org/0009-0006-8801-3640>

Yolov9, as the latest version of the You Look Only Once (YOLO) series, not only has higher detection accuracy, but also has a significant improvement in inference speed, reaching a new State-of-the-art (SOTA) on the MS COCO dataset, and becoming one of the research hotspots in the field of object detection. It addresses these challenges through the introduction of Programmable Gradient Information (PGI) and the introduction of Generalized Efficient Layer Aggregation Network (GELAN) by improving the information retention and gradient flow, thus preventing false associations between targets and inputs (Wang, 2024). In terms of object detection performance, the object detection method which base on GELAN and PGI outperform previous "train from scratch" methods, and also outperform RT DETR (Lv, 2023), which uses large datasets for pre-training, and Yolo MS, which is based on a deep convolutional design, in terms of parameter utilization (Wang, 2024; Chen, 2023). PGI's applicability spans from lightweight to large-scale models, and is able to be applied to a variety of models without the need for large pre-training. The applicability of PGI spans from lightweight to large models, and is able to train models from scratch with very good performance without the need for large pre-training datasets.

This study will focus on evaluating the performance of the Yolov9 model through a series of targeted investigations. Firstly, the model's sensitivity to batch size, a critical factor in its performance, is examined, and the interplay between batch size and learning rate is explored to optimize the training process. Secondly, the effectiveness of different optimizers in model training is compared, determining the best-fit optimizer and corresponding learning rate for Yolov9. Additionally, the performance of different scale variants of Yolov9, namely Yolov9-c and Yolov9-e, is assessed by comparing their training performance under varying conditions. Furthermore, the impact of different optimizers on the performance of these models is investigated. This study is aiming to present a straightforward yet efficient method for testing and optimizing the training and performance of the Yolov9 model on the COCO128 dataset, offering valuable insights and solutions for object detection tasks, particularly for small research teams or those

with limited computational resources. This study serves as a practical reference for optimizing the utilization of Yolov9 in target detection tasks, particularly for resource-constrained research teams.

2 METHODOLOGIES

2.1 Dataset Description and Preprocessing

In this study, the COCO128 dataset was chosen as the main data source for the study. COCO128 is the first 128 sheets of the dataset Microsoft Common Objects in Context (MS COCO), which is a large-scale multi-category object detection dataset containing more than 80 categories totaling more than one million. The MS COCO dataset is a large-scale multi-category target detection dataset that contains more than 80 categories totaling more than one million images which cover a variety of real scenes, such as indoor and outdoor scenes, and thus is highly representative and challenging. Each image of the COCO128 dataset is labeled with the bounding box of one or more objects and their category information, and also contains segmentation masks and key points for the objects, as well as a preference for images in which the target co-occurs with its scene, i.e., non-iconic images, which can reflect visual semantics and are more in line with the requirements of the image understanding task. This detailed labeling information makes it one of the ideal datasets for tasks of object detection, instance segmentation, pose estimation and so on.

2.2 Proposed Approach

This study will focus on exploring the performance of optimizing the Yolov9 model and investigating the performance of Yolov9 under each condition. In general, the experiment firstly reproduces the model and imports the dataset, then the training configuration and environment are selected, the experiment-testing model is started, and finally the images/tables are drawn based on the results and the analysis is given. The pipeline is shown in the Figure 1.

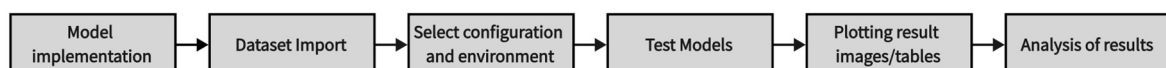


Figure 1: General flow of the experiment (Picture credit: Original).

This study will not be limited to a single Yolov9 model, but will delve into two models, Yolov9-c and Yolov9-e, to explore models with different sizes and number of parameters, and for the different performances of hyper-parameter tuning hyper and optimizer choices, this study will try to find the optimal combinations in order to make the model converge faster and get better performance during the training process. Performance. The detail is shown in the Figure 2.

The study first tests the performance of the Yolov9 model by tuning the hyperparameters, selecting the optimizer and fixing the epoch, and continuously adjusting different learning rates to train the model and record the performance.

Secondly, this experiment adjusts the batch size and learning rate to explore the most suitable and matching learning rate of the two models under different batch sizes to improve the training effect and efficiency.

On this basis, explore how many epochs after which the large model Yolov9-e outperforms the small model Yolov9-c, and explore the efficiency and effectiveness of their training, and what the change in the relationship reflects.

Finally, different optimizer choices are used to explore the impact of different optimizers on the training effectiveness of Yolov9, and the learning rate is adjusted in parallel to explore the optimizer that has the best effect on the performance of the model with which it has the best effect on the performance of the model with the optimizer and the learning rate that matches it.

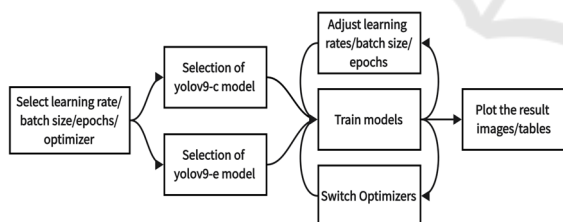


Figure 2: Experimental details flow (Picture credit: Original).

2.2.1 Yolov9 Model Structure and Features

Yolov9 builds on the foundation of Yolov7 with an updated architecture that combines the Programmable Gradient Information (PGI) concept with the Generative Latent Embedding for Object Detection (ELAN) architecture to further enhance its functionality in object detection, representing a major leap forward in accuracy, speed, and efficiency, and solidifying its position as a next-generation top-of-the-line real-time object detector.

Yolov9 maintains the signature features of the Yolo family by providing real-time object detection compared to other object detection models and its predecessors in the Yolo family. This means that it can quickly process an input image or video stream and accurately detect objects within it without compromising speed.

At the same time, Yolov9 uses for the first time the concept of Programmable Gradient Information (PGI), which allows the model to make the gradient more reliable and stable by aiding reversible branching, as shown in the Figure 3. This ensures that the deep features retain the key features required to perform the target task, addressing the problem of information loss during deep neural network feedforward. In addition, it uses the GELAN architecture, a new architecture that allows parameters to be optimized, computational complexity to be significantly reduced, and accuracy and inference speed to be improved. GELAN enhances the flexibility and efficiency of Yolov9 by allowing the user to select the appropriate computational module for different inference devices. Yolov9's architecture can be easily integrated into a variety of systems and environments due to its better accuracy with fewer parameters and computations, making it suitable for a wide range of applications including surveillance, self-driving vehicles, robotics, and more.

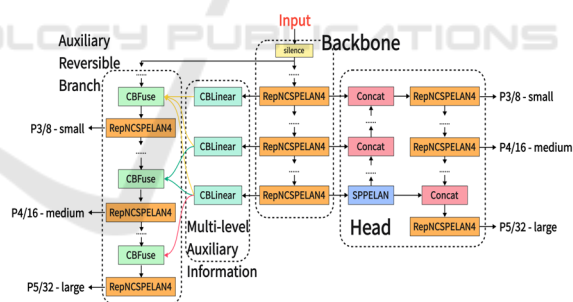


Figure 3: YOLOv9 model structure (Picture credit: Original).

2.2.2 Information Bottleneck Principle & Reversible Functions

The information bottleneck principle shows that the loss of information often occurs when data X is in the process of being transformed, as shown in the following equation:

$$I(X, X) \geq I(X, f^\theta(X)) \geq I(X, g^\sigma(f^\theta(X))) \quad (1)$$

In this equation, the mutual information is denoted by I, the transformation functions are f and g, and the

operations of the two successive layers in the neural network are denoted by as f^θ and g^φ well, respectively.

As neural networks deepen, there's a risk of information loss in each layer, affecting the model's ability to retain original data and leading to unreliable gradients during training. Yolov9 addresses this challenge by incorporating reversible architectures, ensuring that each layer preserves complete original information. By utilizing reversible functions, Yolov9 maintains reliable gradients for model updates, enabling convergence even in deep networks. Additionally, Yolov9 leverages transformer models to find inverse transformations, ensuring that features retain sufficient information despite complex mappings, thus enhancing model performance.

There are other models that use other methods to solve the above problem. For example, Diffusion Models and Variational Autoencoder (VAE), both of which have the ability to find inverse functions. However, because lightweight models will be under-parameterized during the training process, a large number of parameters will be transformed and become a large amount of raw data, and when applying these methods to lightweight models, vulnerabilities will occur. Similarly, $I(Y, X)$, which maps data X to important information about the target Y , is subject to the same problem. To address this issue, Yolov9 discusses the adoption of the concept of information bottlenecks:

$$I(X, X) \geq I(Y, X) \geq I(Y, f^\theta(X)) \geq \dots \geq I(Y, Z) \quad (2)$$

Although $I(Y, X)$ only accounts for a small portion of $I(X, X)$, it is crucial to the target task. Therefore, even if a lot of information is not lost in the forward propagation phase during model fallout, the training effect will be greatly undermined as long as $I(Y, X)$ is lost. In the down-parameterized state, it is difficult for a lightweight model not to easily lose a lot of important information in the forward propagation phase. Therefore, it is a crucial part for lightweight models to figure out how to filter $I(Y, X)$ from $I(X, X)$ accurately.

In Yolov9, the reversible architecture's operation units maintain the property of reversible transformation to ensure that the output feature maps of each operation unit layer retain the complete original information.

2.2.3 Programmable Gradient Information (PGI)

The PGI consists of 3 main parts, i.e., main branch, auxiliary reversible branch, and multilevel auxiliary information. Among them, the auxiliary reversible branch is involved in the process of neural network deepening, which improves the problem of deepening the network has a good improvement. Network deepening causes information bottlenecks that prevent the loss function from generating reliable gradients.

The concept of auxiliary reversible branching ensures minimal information loss during forward propagation by maintaining the integrity of input data. However, directly constructing the main branch into a fully reversible form can significantly increase inference time, potentially hindering higher-order semantic learning. In the PGI framework, the auxiliary reversible branch acts as a deep supervision mechanism during training, providing complementary information flow and precise gradient updates. This approach ensures that critical feature information is retained, aiding in the extraction of relevant features for the target task. Multilevel auxiliary information addresses error accumulation in deep supervision, especially for models with multiple predictive branches. It integrates an integration network between hierarchical layers of the feature pyramid and the main branch to combine gradients from different prediction heads. This aggregated gradient information, containing all target objects, updates the main branch parameters, preventing dominance of specific object information in the feature pyramid hierarchy. Yolov9's approach mitigates information leakage in deep supervision and allows flexibility in choosing the semantic levels required to guide learning in networks of various sizes.

2.2.4 Generalized Efficient Layer Aggregation Network (GELAN)

Yolov9 generalized the capabilities of ELAN, whereas the original ELAN uses only a stack of convolutional layers, GELAN can use any computational block as a base Module (Wang, 2022). GELAN is a novel architecture designed by combining two neural network architectures, i.e., combining with gradient path planning Cross Stage Partial Network (CSPNet) and ELAN to design a Generalized Efficient Layer Aggregation Network with a combination of lightweight, speed of inference, and accuracy. GELAN ensures flexibility by allowing

any computational block to extend the layer aggregation of ELAN. The architecture is designed to achieve efficient feature aggregation while maintaining competitive performance in terms of speed and accuracy. The overall design of GELAN incorporates the cross-level partial connectivity of CSPNet and the efficient layer aggregation of ELAN for effective gradient propagation and feature aggregation. The structure is shown in the Figure 4.

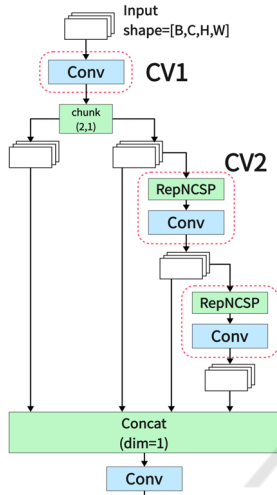


Figure 4: GELAN structure (Picture credit: Original).

2.3 Implementation Details

During model execution, key aspects include hyperparameter tuning. Initial settings involve a

learning rate of 0.01 and a batch size of 16. Exploring batch size and learning rate relationships, the learning rate decreases incrementally with halving batch size (ensuring it remains a power of 2). Initially, Stochastic Gradient Descent (SGD) optimizers are chosen for their computational speed and convergence. Subsequent optimizer exploration involves Adaptive Moment Estimation (Adam), and EvoLved Sign Momentum (LION), with simultaneous learning rate adjustments to gauge their impact on model performance. YOLOv9 preprocesses datasets by letterboxing, adding grey bars, and adjusting batch dimensions from HWC to CHW, enhancing size, diversity, and generalization ability, thus reducing overfitting. The comparison highlights two YOLOv9 models: YOLOv9-c (compact) and YOLOv9-e (extended). YOLOv9-C, the compact model, surpasses its predecessor, YOLO MS-S, by reducing parameters and computational loads while improving AP accuracy by 0.4% to 0.6%. YOLOv9-E excels in balancing model complexity and detection accuracy, markedly reducing parameters and computation requirements while enhancing accuracy, as shown in the Figure 5.

3 RESULTS AND DISCUSSION

This section thoroughly examines the performance of the YOLOv9 model, focusing on key aspects. It begins by analyzing the impact of varying learning rates on both YOLOv9-c and YOLOv9-e models.

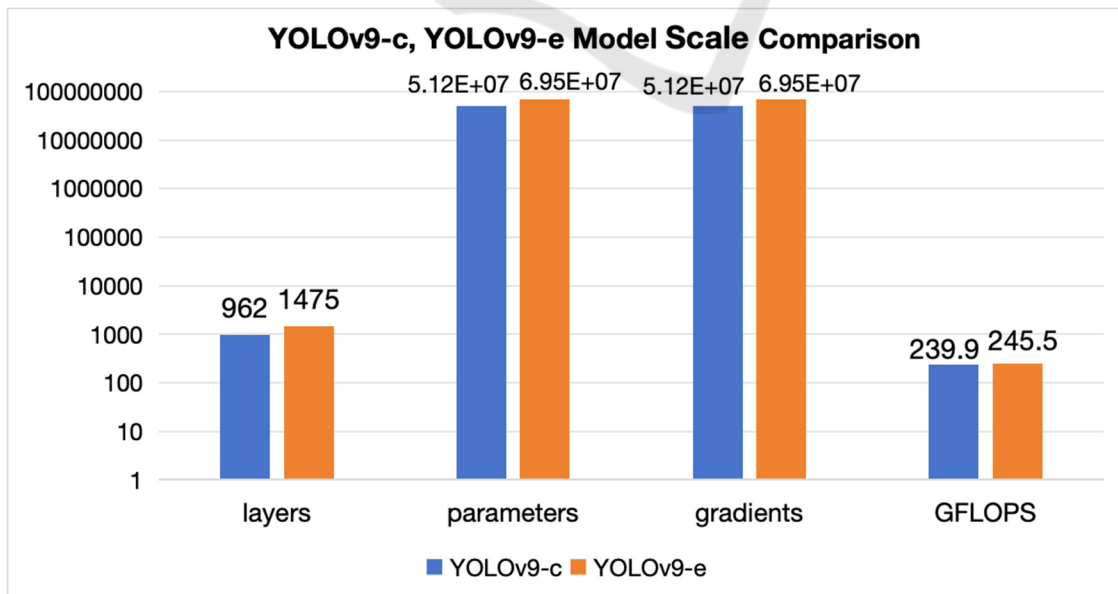


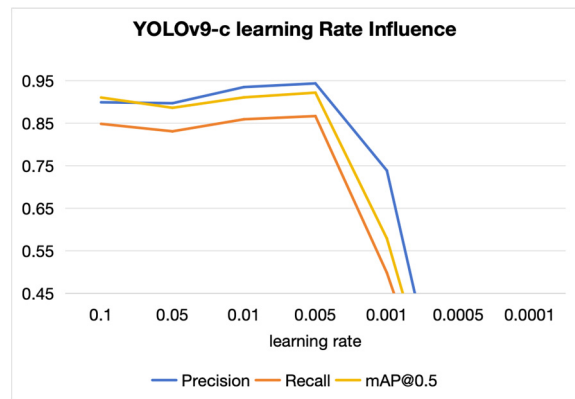
Figure 5: YOLOv9-c and YOLOv9-e size comparison (Picture credit: Original).

Next, it investigates the effects of batch size and learning rate on model performance, identifying optimal training strategies. The study then compares the training iterations required for YOLOv9-c and YOLOv9-e models and explores their differences during training. Finally, it assesses the influence of different optimizers on model performance, highlighting their advantages and disadvantages. These analyses offer valuable insights for optimizing YOLOv9 model performance and guide further research in this area.

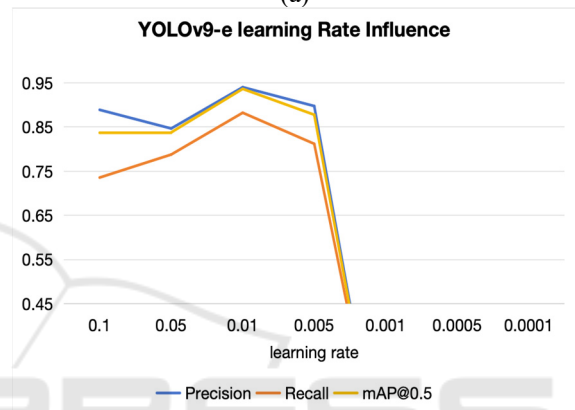
3.1 Effect of Learning Rate on the Performance of YOLOv9-c and YOLOv9-e Models

In this part, SGD is chosen as the optimizer for the experiment, because SGD possesses the advantages of low computational cost and avoidance of local minima due to stochasticity, which is conducive to the test of learning rate. Meanwhile, the epochs value is chosen to be 1500 to ensure that the model can converge sufficiently to ensure optimization at different learning rates. The trend of model performance with learning rate is shown in the following Figure 6.

The performance trend of both YOLOv9-c and YOLOv9-e models is evident across different learning rates (0.1, 0.05, 0.01, 0.005). Initially, performance gradually improves with increasing learning rates, peaking at around 0.94. However, a rapid decline occurs beyond this point, resulting in significant performance loss at learning rates of 0.0005 and 0.0001. This decline is attributed to selecting excessively small learning rates, leading to the models being trapped in local optima despite the effects of SGD, thus hindering normal training outcomes. Additionally, the experiment revealed that due to its deeper model structure and increased parameters, YOLOv9-e is more sensitive to learning rate adjustments compared to YOLOv9-c. Consequently, even small changes in learning rates have a more pronounced impact on YOLOv9-e's performance, resulting in greater fluctuations during adjustments.



(a)



(b)

Figure 6: Effect of learning rate on YOLOv9-c/e model (Picture credit: Original).

3.2 Effect of Batch Size and Learning Rate on Model Performance

In this section, SGD serves as the optimizer to better illustrate the influence of learning rate and batch size on model performance. The mean Average Precision (mAP) is selected as the performance metric due to its comprehensive evaluation of Precision and Recall, offering a nuanced assessment across different detection categories. For instance, when training YOLOv9-c with a 4090 graphics card, a batch size of 16 fully utilizes video memory without causing 'Out of Memory' errors. However, for YOLOv9-e, with its larger model size, the batch size needs to be reduced by 8 to ensure optimal performance. This disparity in batch sizes presents an intriguing discussion point regarding their impact on training outcomes.

The model performance and the corresponding learning rate in relation to epochs and batch size are shown in the Table 1.

Table 1: Batch size and learning rate matching and performance of YOLOv9-c/e.

YOLOc9-c				YOLOc9-e			
batch size	learning rate	mAP@0.5	epochs	batch size	learning rate	mAP@0.5	epochs
16	0.01	0.91069	800	8	0.01	0.93639	1200
16	0.005	0.92175	1200	8	0.005	0.87780	1200
8	0.01	0.92392	1200	4	0.01	0.90821	1200
8	0.005	0.88570	1500	4	0.005	0.90698	1500

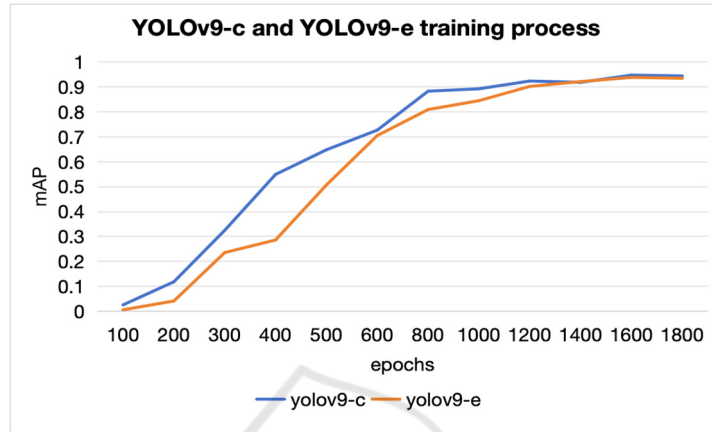


Figure 7: Training efficiency of YOLOv9-c/e (Picture credit: Original).

The experimental test includes learning rates of 0.1, 0.05, 0.01, 0.005, and 0.001. The table highlights the optimal learning rates - 0.01 and 0.005 - for discussion. When training yolov9-c with a batch size of 16, both 0.01 and 0.005 yield comparable results, but 0.005 requires over 50% more iterations. Thus, 0.01 is recommended for its balance of effectiveness and efficiency. Similarly, with a batch size of 8, 0.01 outperforms 0.005 by nearly 4% in mAP, making it the preferred choice. For yolov9-e, at a batch size of 8, 0.01 achieves a nearly 6% higher mAP compared to 0.005, reinforcing its recommendation. When the batch size reduces to 4, 0.01 and 0.005 yield similar results, but with fewer epochs required, 0.01 is preferred. Hence, 0.01 is consistently recommended as the optimal learning rate.

3.3 Training Efficiency of Yolov9-c and Yolov9-e

Due to the substantial disparity in volume between yolov9-c and yolov9-e, including their layer count, parameter quantity, and computational complexity, their selection for practical applications varies based on specific requirements. Therefore, assessing the number of iterations for both models is crucial, offering insights into their training efficiency. In this experiment, SGD serves as the optimizer, with mAP employed as the evaluation metric. To prevent memory overflow, a batch size of 8 is selected.

The trend of the training effect of the two models with the growth of epoch is shown in the Figure 7.

The training curves of yolov9-c and yolov9-e in Fig. indicate that yolov9-c exhibits notably superior training efficiency in the initial stages, consistently maintaining a higher mAP compared to yolov9-e. Although yolov9-e eventually surpasses yolov9-c after approximately 1,400 epochs, the margin of improvement is marginal. Throughout continued training, yolov9-c maintains a slight edge over yolov9-e. Therefore, for datasets with modest size and less stringent requirements on generalization and accuracy, prioritizing training efficiency and model speed, yolov9-c is recommended due to its simpler parameter count, lower computational complexity, and commendable training results.

3.4 Impact of Different Optimizers on Model Performance

Different optimizers have different impacts on different models, in different cases the selection of specific optimizers will bring improvement to the experiment, at the same time, different optimizers also have different training efficiency, in the practical application will be based on the demand and the specific model to choose different optimizers. Therefore, it is of great interest to discuss the impact of the optimizer on the model performance. In this part of the experiment, the learning rate of 0.1-

0.00001 and the corresponding epochs are chosen, and the batch size of yolov9-c is 16, and that of yolov9-e is 8, to make full use of the video card memory.

The best performance of the two models with different optimizers is shown in the Figure 8.

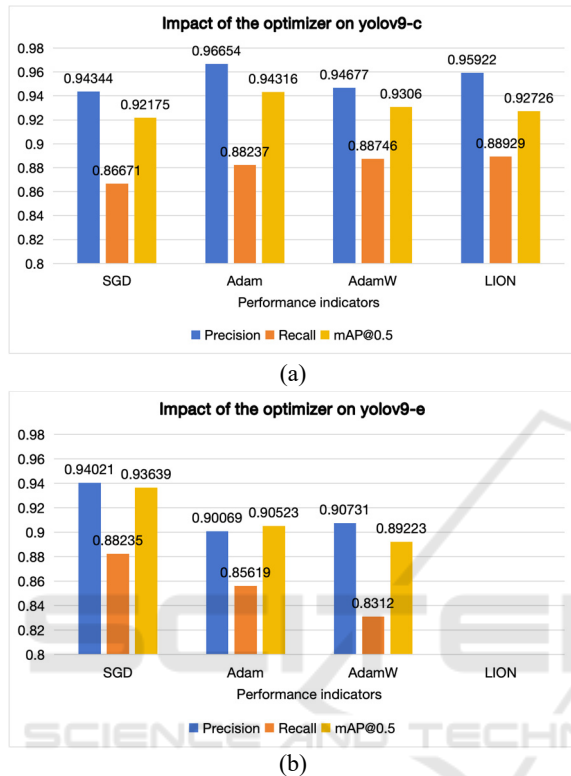


Figure 8: The different results of two models (Picture credit: Original).

In this experiment, the impact of different optimizers on yolov9-c and yolov9-e models is assessed. For yolov9-c, all four optimizers yield satisfactory results, with Adam demonstrating optimal performance. Conversely, for the larger yolov9-e model, SGD proves to be the most effective optimizer. Notably, the LION optimizer struggles to converge, particularly on yolov9-e, indicating its suitability for larger batch sizes. Given these findings, the LION optimizer may not be suitable for experiments with limited GPU memory, and SGD is recommended as a more suitable alternative.

The learning rates for the above experimental results for yolov9-c and yolov9-e are shown in the Table 2.

Table 2: Learning rate of YOLOv9-c/e for different optimizer.

Method/optimizer	SGD	Adam	AdamW	LION
YOLOv9-c	0.005	0.001	0.001	0.0001
YOLOv9-e	0.01	0.001	0.001	None

4 CONCLUSIONS

This study investigates the training efficacy and performance of YOLOv9 models, crucial for resource-constrained research teams venturing into object detection. By training and testing on the COCO128 dataset with limited GPU memory, optimal learning rates and batch sizes for yolov9-c and yolov9-e are determined. Findings reveal that yolov9-e is more sensitive to learning rate adjustments, particularly with SGD as the optimizer. Yolov9-c performs best with a batch size of 16, while yolov9-e prefers a batch size of 8, both with a learning rate of 0.01. Moreover, yolov9-c demonstrates superior training efficiency and comparable performance to yolov9-e on small datasets. The study recommends Adam for yolov9-c and SGD for yolov9-e on small datasets, cautioning against using the LION optimizer due to batch size limitations. Future research will delve into exploring yolov9's performance with larger datasets and computational resources, focusing on maximizing detection speed without compromising accuracy.

REFERENCES

- Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp: 4700-4708.
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., & Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv :2010.11929.
- Tu, Z., Talebi, H., Zhang, H., Yang, F., Milanfar, P., Bovik, A., & Li, Y. (2022, October). Maxvit: Multi-axis vision transformer. In European conference on computer vision, pp: 459-479.
- Liu, Y., Tian, Y., Zhao, Y., Yu, H., Wang, Y., & Liu, Y. (2024). Vmamba: Visual state space model. arXiv:2401.10166.
- Chen, K., Lin, W., Li, J., See, J., Wang, J., & Zou, J. (2020). AP-loss for accurate one-stage object detection. IEEE

- Transactions on Pattern Analysis and Machine Intelligence, vol. 43(11), pp: 3782-3798.
- Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., & Savarese, S. (2019). Generalized intersection over union: A metric and a loss for bounding box regression. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 658-666.
- Wang, J., Song, L., Li, Z., Sun, H., Sun, J., & Zheng, N. (2021). End-to-end object detection with fully convolutional network. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp: 15849-15858.
- Wang, C. Y., Yeh, I. H., & Liao, H. Y. M. (2024). YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information. arXiv: 2402.13616.
- Lv, W., Xu, S., Zhao, Y., Wang, G., Wei, J., Cui, C., ... & Liu, Y. (2023). Detsr beat yolos on real-time object detection. arXiv:2304.08069.
- Chen, Y., Yuan, X., Wu, R., Wang, J., Hou, Q., & Cheng, M. M. (2023). YOLO-MS: rethinking multi-scale representation learning for real-time object detection. arXiv:2308.05480.
- Wang, C. Y., Liao, H. Y. M., & Yeh, I. H. (2022). Designing network design strategies through gradient path analysis. arXiv:2211.04800.

