國立陽明交通大學

科技管理研究所

碩士論文

Institute of Management of Technology

National Yang Ming Chiao Tung University

Master Thesis Dissertation

基於 CPA 模組之 Lite UPANets 與注意力機制之物件偵測問題處理

Object Detection Processing Based on CPA Module for Lite UPANets and Attention Mechanism

研究生：李品陞（Li, Pin-Sheng）

指導教授：李昕潔（Lee, Shin-Jye）

中華民國一一二年八月

August 2023

基於 CPA 模組之 Lite UPANets 與注意力機制之物件偵測問題處理

Object Detection Processing Based on CPA Module for Lite UPANets and Attention Mechanism

研 究 生：李品陞　　Student：Pin-Sheng Li

指導教授：李昕潔 博士　　Advisor：Dr, Shin-Jye Lee

國立陽明交通大學

科技管理研究所

碩士論文

A Thesis
Submitted to Institute of Management of Technology
College of Management
National Yang Ming Chiao Tung University
In partial Fulfillment of the Requirements
for the Degree of
Master of Business Administration
in
Management of Technology

August 2023
Taiwan, Republic of China

中華民國 一一二年八月

誌謝

感謝我的父母這一路以來對我的栽培，讓我在能夠全心專注於課業不用為生活所迫外，也讓從你們身上學習人生路上的一些做人的智慧。

感謝我的指導教授李昕潔博士帶領我進入人工智慧的領域，讓我從原本只有數學系的相關背景，沒有任何程式語言的經驗，督促並指導我學習 python 語言與深度學習相關的知識，並且在撰寫論文的過程中，幫助我釐清論文的脈絡，並在論文的撰寫遇到困難時，適時地給我建議，並不斷地給我加油打氣。

感謝博士班學長曾敬勳對我的幫助，在我跑模型與修改模型遭遇卡頓時，引導我解決在編寫代碼時的盲點，讓我後續在跑實驗時減少了不少的阻礙。

最後，感謝我所有的家人、同學及朋友，在我就讀研究所時給我的支持與鼓勵，讓我得到滿滿的動力，使我能夠順利完成畢業論文。

李品陞 謹誌

中華民國一一二年八月

於新竹陽明交通大學

基於 CPA 模組之 Lite UPANets 與注意力機制之物件偵測問題處理

研究生: 李品陞　　指導教授: 李昕潔 博士

國立陽明交通大學科技管理研究所碩士班

摘要

在比較了許多物件偵測的相關模型會發現，模型本體的部分往往過於龐大，其參數量往往高達數千萬，造成單位參數所帶來的貢獻相對低下，因此想嘗試能否在降低參數量的情況下，盡量保持相當程度的精度。我們提出了三種關於 YoloX 的模型修改來檢視不同模型的偵測效果，並討論實驗結果背後的可能原因。第一種模型是將 YoloX 模型的骨幹網路 CSPDarknet 替換成較低參數量、帶有 CPA 的 Lite UPANets，其透過 Channel Pixel Attention（CPA）模組來有效的混和不同特徵層的特徵。第二種模型與第三種模型都是在前一種的模型上加入與注意力機制相關的常用模組，分別是 SE、CBAM 與 ECA。為了評估不同模組添加在網路不同位置上所帶來的效果，第二種模型選擇在 Backbone 與 Neck 之間添加，而第三種則是在 Neck 當中添加。此外，為了比較所提出的模型在大模型與小模型之間的泛用性，我們會將所提出的模型與 YoloX 模型的三種變體進行對應。

根據我們的實驗結果，第一種模型成功達成了提高參數貢獻率的目標，意味著透過 CPA 模組可以幫助特徵圖在不丟失原始特徵的情況下融合更複雜的特徵圖。另一方面，添加額外注意力機制模組的第二種與第三種模型大致上都能有效地在前一種的基礎上些微改進模型表現。其中以 CBAM 模組表現最好、SE 模組次之，而 ECA 模組則沒有帶來明顯的進步。

關鍵字：物件偵測、Channel Pixel Attention、UPANets、注意力機制、深度學習

Object Detection Processing Based on CPA Module for Lite UPANets and Attention Mechanism

Student: Pin-Sheng Li Advisor: Dr. Shin-Jye Lee

Institute of Management of Technology

National Yang Ming Chiao Tung University

ABSTRACT

After comparing many models related to object detection, it can be found that the part of the model itself is often too large, with a number of parameters often reaching tens of millions, resulting in a relatively low contribution of unit parameter. Therefore, this work wants to try to maintain a considerable degree of accuracy while reducing the number of parameters. This work proposes three model modifications for YoloX to examine the detection performance of different models and discuss the possible reasons behind the experimental results. The first model replaces the backbone network CSPDarknet of YoloX model with lower-parameter Lite UPANets with CPA, which effectively mixes the features of different feature layers through the Channel Pixel Attention (CPA) module. The second model and third model both add commonly used modules related to attention mechanism to the previous model, namely SE, CBAM, and ECA. In order to evaluate the effect of adding different modules at different parts of network, the second model chooses to add them between the Backbone and the Neck, while the third model adds them in the Neck. In addition, to compare the generalization of the proposed model between large and small models, this work will correspond the proposed model to three variants of the YoloX model.

According to our experimental results, the first model successfully achieves the goal of

increasing parameter contribution rate, which means that the CPA module can help feature maps fuse more complex feature maps without losing the original features. On the other hand, the second and third models that add additional attention mechanism modules can generally effectively improve performance slightly on the basis of the previous one. Among them, the CBAM module performs the best, followed by the SE module, while the ECA module brings no significant progress.

Keywords: Object Detection, Channel Pixel Attention, UPANets, Attention mechanism, Deep Learning

Contents

List of Figures

List of Tables

1. Introduction

1.1 Background

In the wave of artificial intelligence, object detection has been a research topic for many experts in the past 20 years, which is an important computer vision task for detecting certain types of visual objects (such as humans, animals or cars) in digital images. Zou et al. (2019) suggest that if expecting to use a simple sentence to describe what object detection is, the following sentence is very appropriate:

What objects are where?

Looking back at the history of object detection in the past, with 2014 as the watershed, it can be roughly divided into two stages: traditional object detection methods and deep learning based object detection methods. Most early object detection algorithms were based on handcrafted features, including the Viola Jones detector (Viola & Jones, 2001), HOG detector (Dalal & Triggs, 2005), and DPM (Felzenszwalb et al., 2008). As the performance of handcrafted features tended to saturate, object detection reached a plateau after 2010. But in 2012, the world witnessed the rebirth of convolutional neural network (Krizhevsky et al., 2012), so object detection officially stepped into the era of deep learning.

However, in the era of deep learning, object detection has generated two branches: “one-stage detection” and “two-stage detection”. The two-stage detection is represented by R-CNN (Girshick et al., 2014), followed by Fast R-CNN (Girshick, 2015), which introduced the concept of SPPNet (He et al., 2014) to improve R-CNN. Later, Faster R-CNN (Ren et al., 2015) was further improved

into a complete end-to-end CNN architecture. One-stage detection is represented by Yolo (Redmon et al., 2015), which is famous for its fast detection speed. Later Single Shot MultiBox Detector (SSD) (Liu et al., 2015) improved the accuracy of small object detection in Yolo, while RetinaNet (Lin et al., 2017) subsequently introduced a new loss function called “focal loss”, which makes the detector pay more attention to examples that are difficult to classify. The development process of object detection over the past 20 years (Zou et al., 2019) can be illustrated as follows:

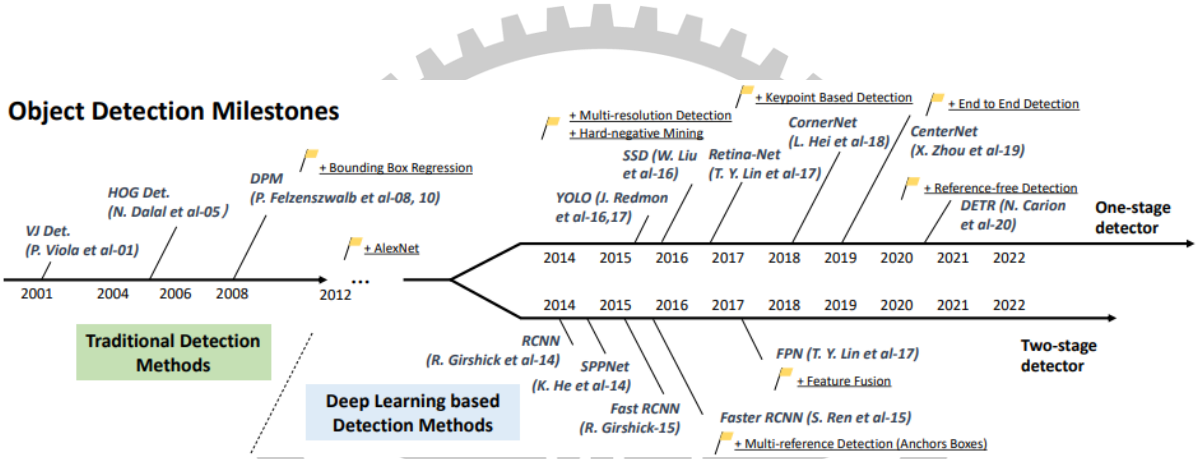


FIGURE 1. A Roadmap for the Development of Object Detection. Figure is taken from Zou et al. (2019).

In recent years, the rapid development of deep learning technology has brought new blood to object detection, leading to significant breakthroughs and pushing it to an unprecedented research hotspot. Nowadays, object detection has a wide range of applications in the real world, such as automatic parking and assisted driving of cars in daily life, and the widespread application of this technology in industrial production lines to detect product yield rate. The following figure (Zou et al., 2019) shows the increasing trend in the number of publications in object detection, highlighting that this field is still thriving.

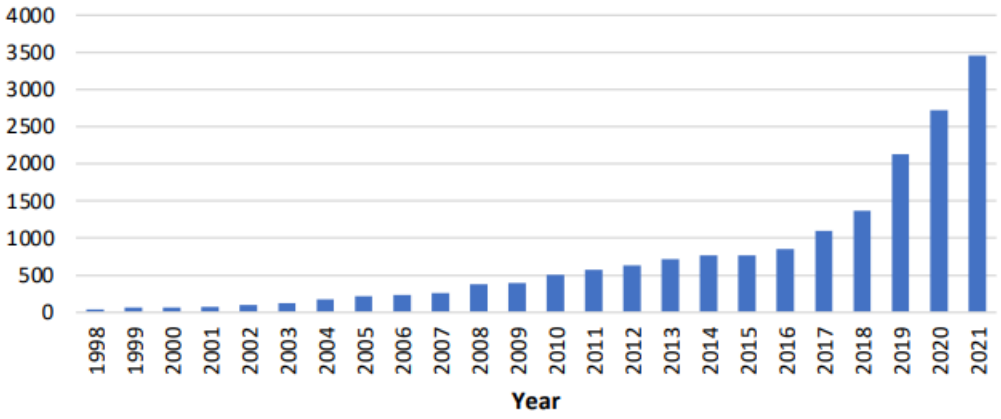


FIGURE 2. Number of Publications in Object Detection from 1998 to 2021. Figure is taken from Zou et al. (2019).

1.2 Research Motivation

After comparing many models related to object detection, it can be found that the network of the model is often very large. With the continuous updating of object detection models, the parameters are often reach tens of millions, resulting in a relatively low contribution of unit parameter. Therefore, I have the idea to solve this problem.

In addition, the backbone of object detection models in the past are mostly traditional image classification models. With the rise of the concept of attention mechanism, it has been mainly used to deal with natural language processing models in the past, but now more and more research has put this mechanism into the field of computer vision. However, after reviewing the relevant papers of Yolo object detection series, there is no example of putting the attention mechanism into the object detection backbone network, which brings the motivation of this research.

1.3 Research Objectives

The objectives of this paper are twofold:

(1) In order to improve the efficiency of parameter utilization and add attention mechanisms to the model, inspired by the CPA modules in UPANets (Tseng et al., 2021), this work develops Lite UPANets to learn and integrate cross channel information. In order to correspond to some variants of the original model, a total of three versions will be developed.

(2) In addition, this work will add commonly used attention mechanism modules to the model for adaptation and summarize the best model.

This study is based on YoloX network (Ge et al., 2021), and its architecture mainly consists of three components: Backbone, Neck, and Head. Backbone is mainly used for feature extraction, usually pre-trained on large datasets (e.g. ImageNet, COCO, etc.), which is a convolutional neural network with pre-trained parameters, e.g. ResNet-50, Darknet53, etc. Head is mainly used to predict the class and location of objects. While Neck is the network layers between Backbone and Head, used to integrate feature maps in different stages.

The main contributions of this paper are:

(1) Replacing the Backbone in yoloX network with a lightweight version of UPANets with attention mechanism will not perform poorly when the number of parameters decreases.

(2) Modify the Neck in yoloX network by adding small modules with attention mechanism to improve model performance with only a few parameters added.

(3) Compare and analyze the results of the proposed models, and provide summary opinions.

2. Related Work

In this chapter, this work will first briefly review the evolution of deep learning and representative models in the field of object detection over the past 20 years. In addition, models related to attention mechanisms, including UPANets and commonly used modules, will be introduced, explaining the structure and mechanisms.

2.1. Deep Learning Review

With the gradual enhancement of computer computing power, in order to pursue higher accuracy, the field of artificial intelligence has gradually developed from traditional machine learning to deep learning. For example, for the common classification problems in life, traditionally experts may use Support Vector Machine (Evgeniou & Pontil, 2001) and Random forest (Louppe, 2014) algorithms to deal with, but now they prefer to use deep learning frameworks like neural networks to solve such problems.

In the early stage of deep learning, there were three common models, namely, the fully-connected neural network for classification and regression, the Convolutional Neural Networks (O'Shea & Nash, 2015) especially applied to the image field, and the Recurrent Neural Networks series (Schmidt, 2019) for natural language processing. With the further development of deep learning, its division of labor is becoming more and more distinct, among which the computer vision field is particularly concerned, such as image classification, image generation, object detection, semantic segmentation, etc., while the object detection field is also increasingly widely used in life, such as assisted driving, Medical imaging

judgement. The following sections will briefly introduce the development history of object detection.

2.2. Traditional Object Detection Methods

Due to the lack of effective image representation in early object detection algorithms, people had to design complex feature representations and various acceleration techniques to exhaust limited computing resources. Moreover, the feature representations would be completely different depending on the difference of the target, so a lot of tedious manpower would be spent beforehand to establish handcrafted features.

2.2.1. Viola Jones Detector (VJ Detector)

More than 20 years ago, VJ detector (Viola & Jones, 2001) was the first object detection method that could detect objects in real time and with good accuracy, following one of the most straightforward detection methods, i.e. sliding windows: going through all possible positions and scales in the image to see if any windows contain objects. VJ detector combines three important techniques: "integral image", "feature selection", and "detection cascades" to greatly increase detection speed. The following figure shows the features used for VJ Detector:

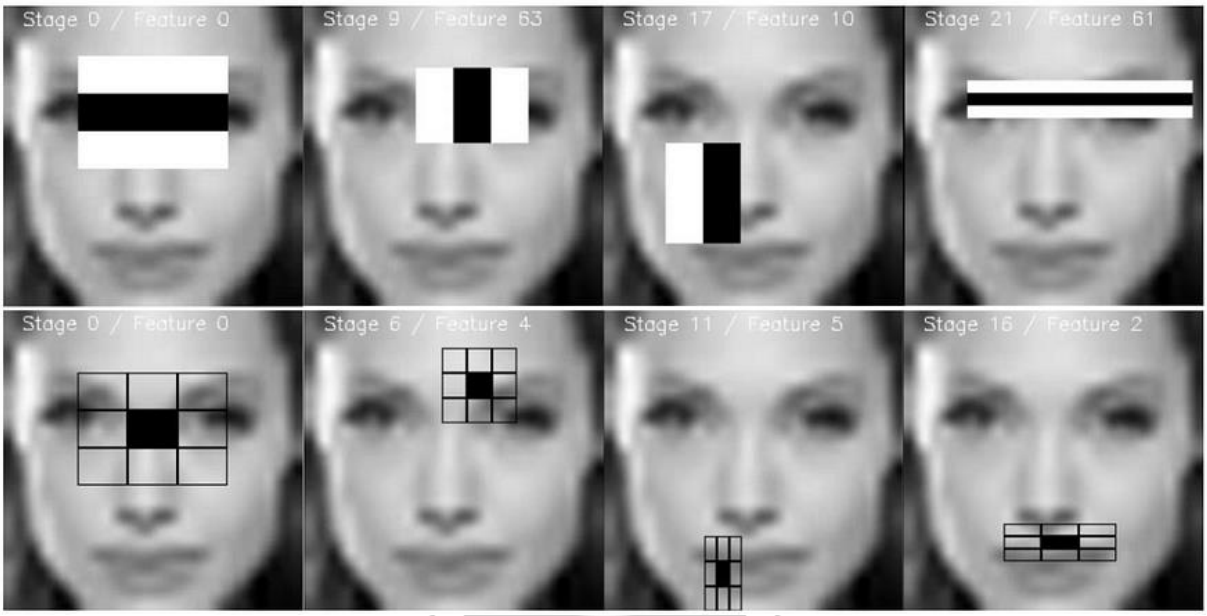


FIGURE 3. Haar Features Used for VJ detector[1]

2.2.2. Deformable Part-based Model (DPM)

DPM was initially proposed by P. Felzenszwalb as an extension of the HOG detector in 2008, and then various improvements were made by R. Girshick. DPM successfully won the championship in VOC-07, VOC-08, and VOC-09 object detection challenges, marking the pinnacle of traditional object detection methods.

DPM follows the principle of "divide and conquer" detection, where training can be simply seen as learning a method for decomposing objects appropriately, while inference can be viewed as an integration of local detection of objects. For example, detecting a car can be treated as detecting the windows, body, and tires. Although so far many object detectors have far exceeded DPM in detection accuracy, many of them are still deeply influenced by its valuable insights, such as mixture models, hard negative mining, bounding box regression, etc. The following figures (Felzenszwalb et al., 2010) show the side and front view of the

[1] Figure 3 Source: https://medium.com/@Andrew_D./computer-vision-viola-jones-object-detection-d2a609527b7c

bicycle model in DPM:

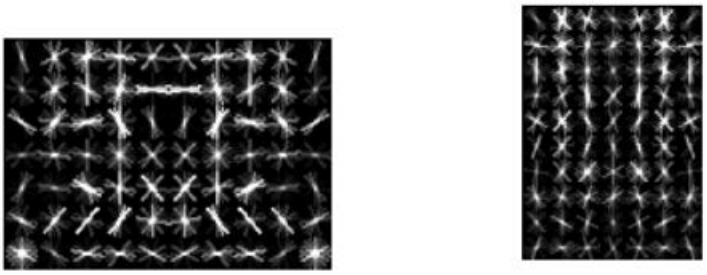


FIGURE 4. The Side and Front View of the Bicycle Model in DPM. Figure is taken from Felzenszwalb et al. (2010).

2.3. R-CNN Series

With the development of deep learning, Convolutional Neural Network have been widely used in the field of images. Research also shows that the deep convolutional network can learn the robust and high-level feature representation of images. Therefore, some people tried to apply it to the field of object detection, which also gave birth to R-CNN. In this section, this work will review relevant papers in the R-CNN series, including R-CNN, Fast R-CNN, and Faster R-CNN. Although Mask R-CNN also belongs to the R-CNN series, it belongs to the field of semantic segmentation and has applications such as facial recognition, virtual fitting rooms, and geological exploration. Therefore, it will not be included in the discussion here.

2.3.1. R-CNN

R-CNN model first uses Selective Search (Uijlings et al., 2013) to find 2000-3000 region proposals, compresses the extracted region proposals to the same size,

and then feeds them into CNN to extract features. Finally, SVM is used for classification and linear regression is performed on the bounding box.

R-CNN model achieved significant performance improvement on the VOC-07 dataset, with mAP significantly increasing from 33.7% (DPM-v5) to 58.5%. However, R-CNN model also has obvious drawbacks, due to the excessive number of region proposals and the need to feed each region proposal into the CNN separately, resulting in high computational complexity and slow detection speed. The R-CNN model algorithm (Girshick et al., 2014) can be illustrated as follows:

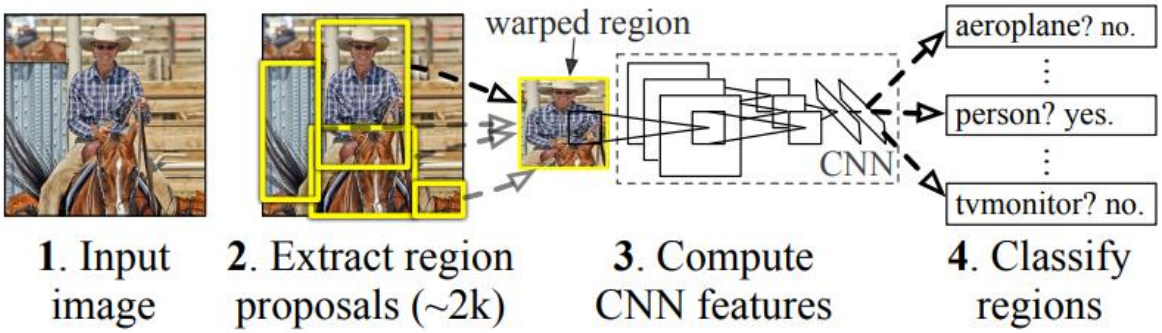


FIGURE 5. The Process of R-CNN Model Algorithm. Figure is taken from Girshick et al. (2014).

2.3.2. Fast R-CNN

In order to address the issue of R-CNN model computational efficiency, Fast R-CNN model introduced the concept of SPPNet model and made improvements. Fast R-CNN model first feeds the entire image instead of the region proposals into the CNN, and then maps the over 2000 region proposals generated by Selective Search onto the feature map. However, due to the need to feed into the fully-connected layers in the end, the size of each region proposal on the feature map

needs to be kept consistent, so a RoI pooling process will be done before this.

On the VOC-07 dataset, Fast R-CNN model increased mAP from 58.5% (R-CNN model) to 70.0%, while detecting more than 200 times faster than R-CNN model. The Fast R-CNN model algorithm (Girshick, 2015) and the RoI pooling process can be illustrated as follows:

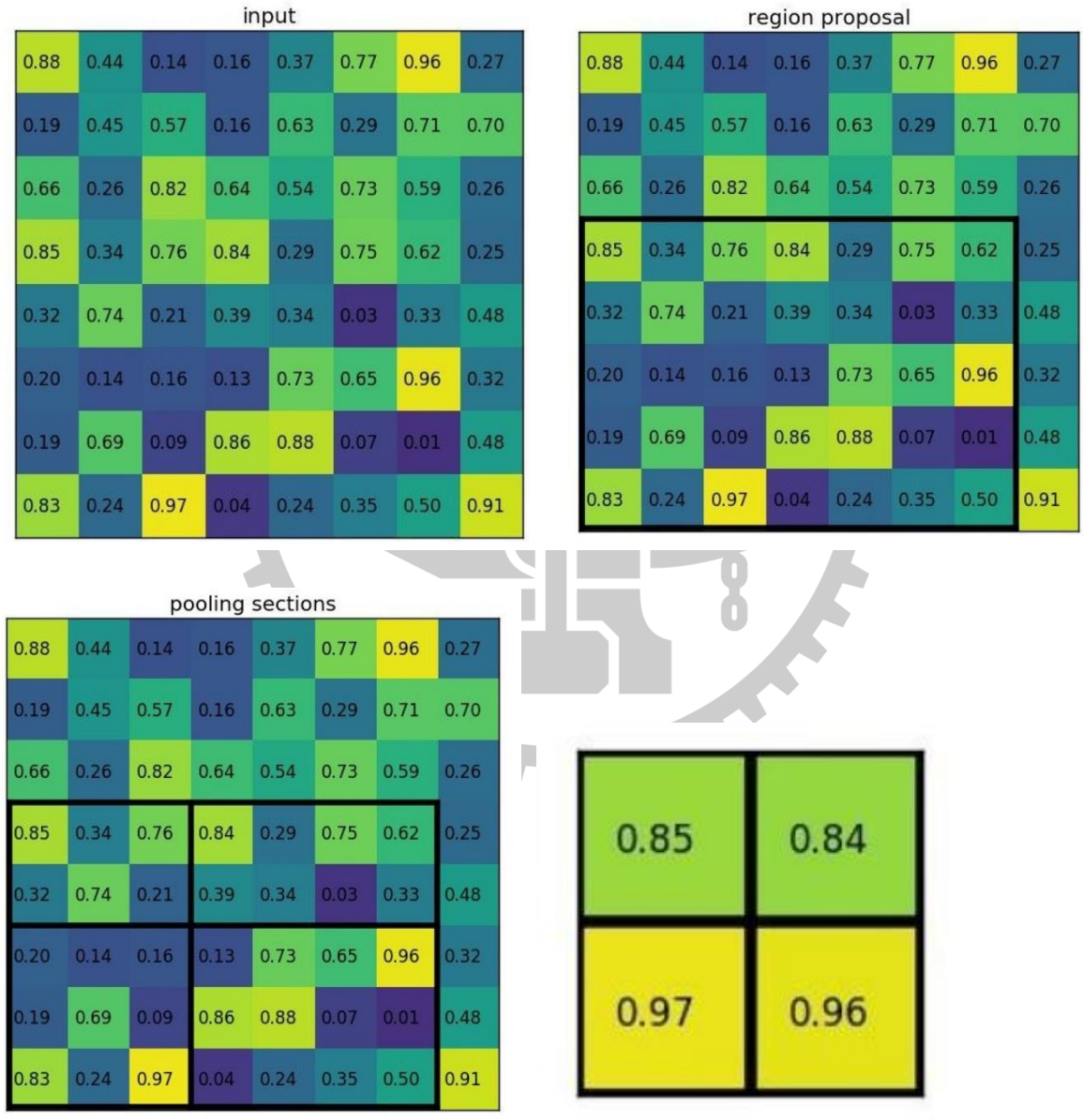


FIGURE 6. The RoI Pooling Process[2]

[2] Figure 6 Source: https://blog.csdn.net/qq_35586657/article/details/97885290

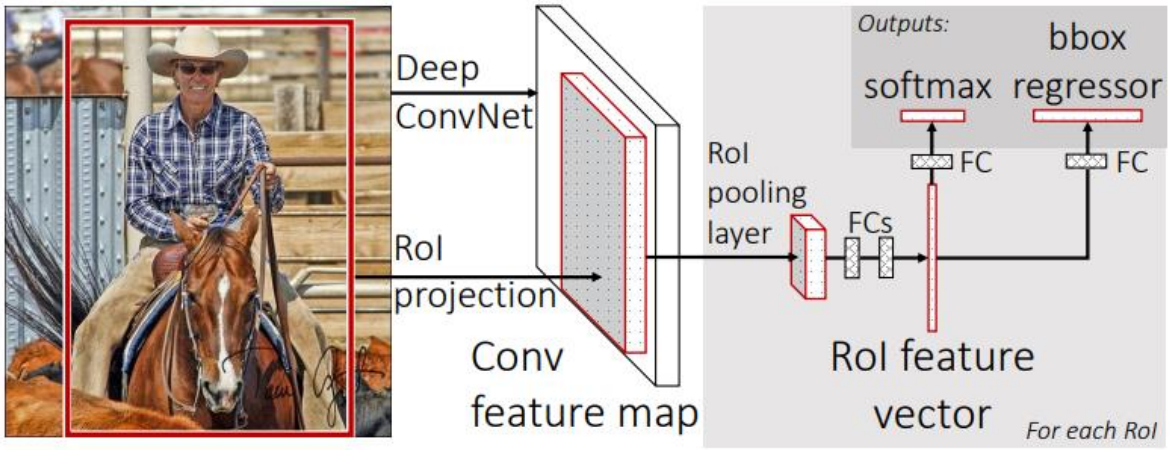


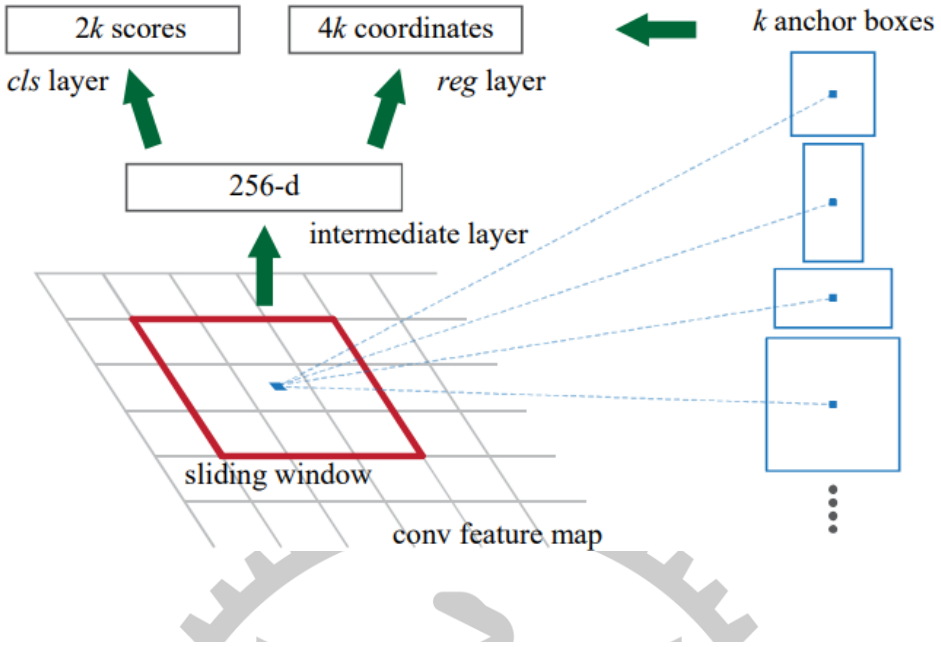
FIGURE 7. The Process of Fast R-CNN Model Algorithm. Figure is taken from Girshick (2015).

2.3.3. Faster R-CNN

Although Fast R-CNN model significantly improves detection speed, there are still speed bottlenecks in using Selective Search to generate region proposals. Therefore, people attempted to implement the generation of region proposals using CNN, and invented Faster R-CNN model by combining it with the original CNN framework.

Faster R-CNN model is similar to Fast R-CNN model, but the difference is that there is an additional branch of Region Proposal Network (RPN) after the images are fed into the CNN, and the RPN uses pre-defined anchor boxes of different sizes to generate the region proposal by sliding window. From R-CNN model to Faster R-CNN model, most of the individual modules of object detection systems, such as region proposal, feature extraction, bounding box regression, etc., have gradually been integrated into a unified, end-to-end framework. The Faster R-CNN model algorithm and the RPN process (Ren et al., 2015) can be illustrated as follows:

(a)



(b)

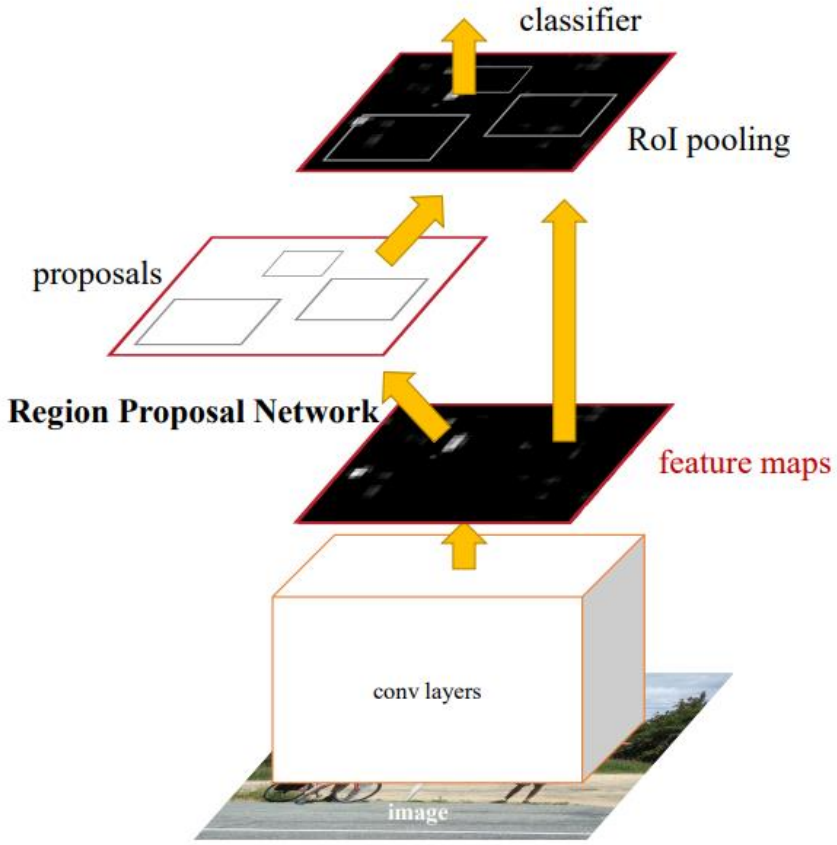


FIGURE 8. (a) The Region Proposal Network Process. (b) The Process of Faster R-CNN Model Algorithm. Figure is taken from Ren et al. (2015).

2.3.4. Summary

From the evolution of R-CNN model to Faster R-CNN model, it can be seen that the progress of model usually does not require improvements in every position of the model, but rather focuses on solving current thorny problems. From focusing on improving performance in R-CNN era, to the reduction of unnecessary calculations in Fast R-CNN era, and finally to the end-to-end framework in Faster R-CNN era, it needs to be achieved step by step, and all these can not be achieved in one day, which is still the case today.

2.4. Yolo Series

Yolo is the abbreviation for "You only look once". From its name, it can be seen that the author has completely abandoned the previous detection mode of "region proposal + verification". On the contrary, it follows a completely different principle: applying a single neural network to the entire image. This network divides the image into multiple regions and simultaneously predicts the bounding boxes and probabilities of each region. The characteristic of Yolo series is its fast detection speed. Compared with two-stage detectors, their localization accuracy may decrease, but with continuous research, this problem has also been solved to a considerable extent.

In this work, the models of Yolo series are selected for the study for the following two main reasons:

(1) They are one of the representative models of single-stage detectors, and as two-stage detectors are gradually being phased out, the study of fast and accurate

models is the trend nowadays.

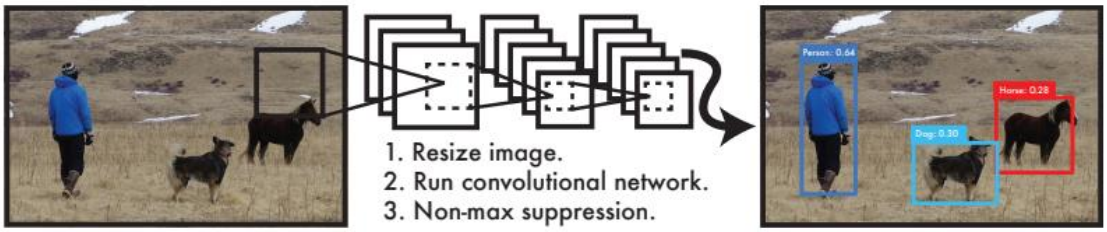
(2) They have undergone multiple generations of changes, from the earliest Yolo-v1 to the current Yolo-v8, and are still under development until this year. Through research, we can understand the current trends in object detection over the past decade, making it convenient for various comparisons.

2.4.1. Yolo-v1

The following two figures (Redmon et al., 2015) can briefly illustrate the entire process of Yolo-v1 algorithm. Firstly, the image will be scaled to a size of 448 × 448, and then feeds into the CNN network structure shown in the figure below to generate a tensor of 7 × 7 × 30. Finally, non-maximum suppression (Hosang et al., 2017) will be performed to remove excess bounding boxes.

In Yolo-v1, if the center point of an object falls on a certain grid, then this grid will be responsible for predicting the object. Because the final result is 7 × 7, it represents that the entire image will be divided into 49 grids, with each grid responsible for predicting up to two objects. There are several predictions for the bounding box generated by Yolo-v1 network, including the center position of the bounding box (x, y), the width and height of the bounding box (w, h), and the confidence that the bounding box contains an object. Plus, there are a total of 20 classes, so 30 is obtained by (4+1) × 2+20. Although Yolo-v1 network may not perform as accurately as Faster R-CNN model, it can process images in real-time at a speed of 45 frames per second.

(a)



(b)

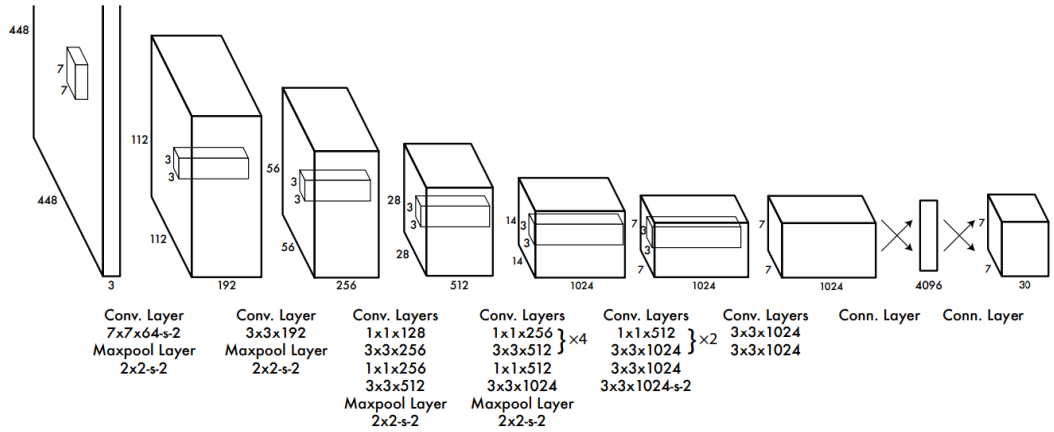


FIGURE 9. (a) The Process of Yolo-v1 Algorithm. (b) The Yolo-v1 Network Structure. Figure is taken from Redmon et al. (2015).

2.4.2. Yolo-v2

Yolo-v1 network has the disadvantage that only one category and at most two objects can be predicted in one grid, which makes it hard to detect when there are many dense and different objects in the image, so Yolo-v2 (Redmon & Farhadi, 2017) had made a series of improvements. Firstly, Yolo-v2 added the anchor box and batch normalization (Ioffe & Szegedy, 2015) strategy to make the model more accurate and fast. In order to improve the localization ability, Yolo-v2 had imposed some limitations on the use of sigmoid functions for the added anchor boxes.

In addition, Yolo-v2 adopted a multi-scale input training strategy, specifically

changing the input image size of the model after a certain interval of iterations during the training process. Finally, due to changes in the output layer structure of Yolo-v2 network, the problem of only being able to predict one category within the same grid was solved. After making the above modifications, the accuracy of Yolo-v2 network has been improved. The structure of Yolo-v2 network can be illustrated as follows:

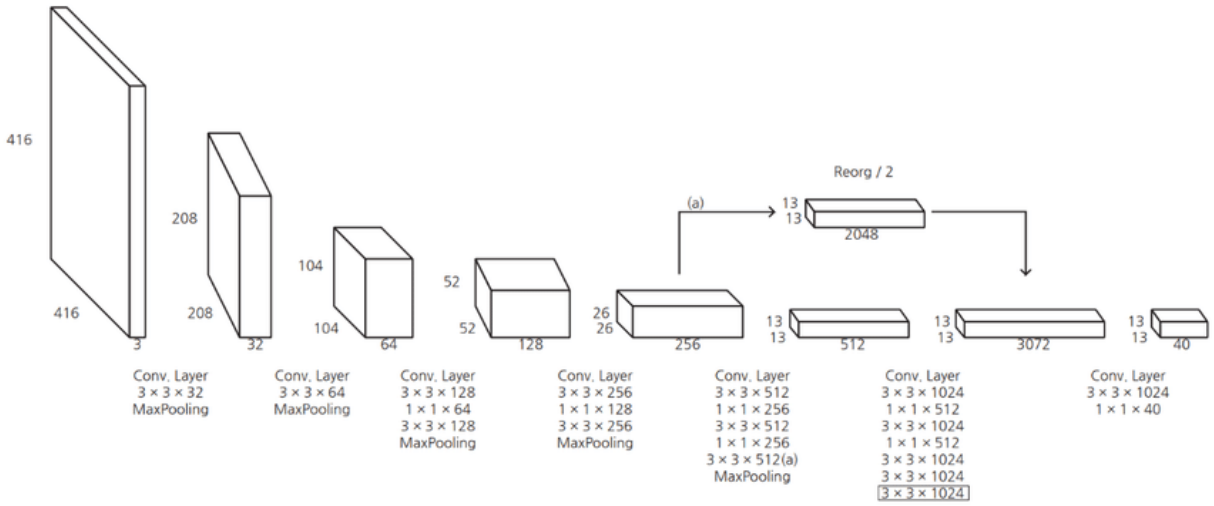


FIGURE 10. The Yolo-v2 Network Structure[3]

2.4.3. Yolo-v3

Yolo-v3 (Redmon & Farhadi, 2018) did not make revolutionary innovations, but instead referred to other papers to optimize its own model, with significant results. It has two main characteristics. Firstly, it uses a new backbone network called Darknet-53, which has changed from the original 19 layers of Yolo-v2 network to 53 layers. Since the layers are deeper, a ResNet (He et al., 2015) structure is used to solve the gradient problem, which is commonly used for

[3] Figure 10 Source: https://www.maskaravivek.com/post/yolov2/

deepening the general neural networks.

Secondly, Yolo-v3 network introduced the concept of Feature Pyramid Networks (Lin et al., 2016) and used the FPN multi-layer prediction architecture to improve the prediction ability of small objects. The feature layer changed from single-layer 13x13 to multi-layer 13x13, 26x26, and 52x52, and bounding boxes changed from single-layer prediction of 5 bounding boxes to three-layer prediction of 3 bounding boxes per layer (total 9 types). The use of FPN architecture allows for the fusion of better target positions at lower levels and better semantic features at higher levels, and independent prediction at different feature layers, resulting in significant improvement in small object detection. The structure of Yolo-v3 network can be illustrated as follows:

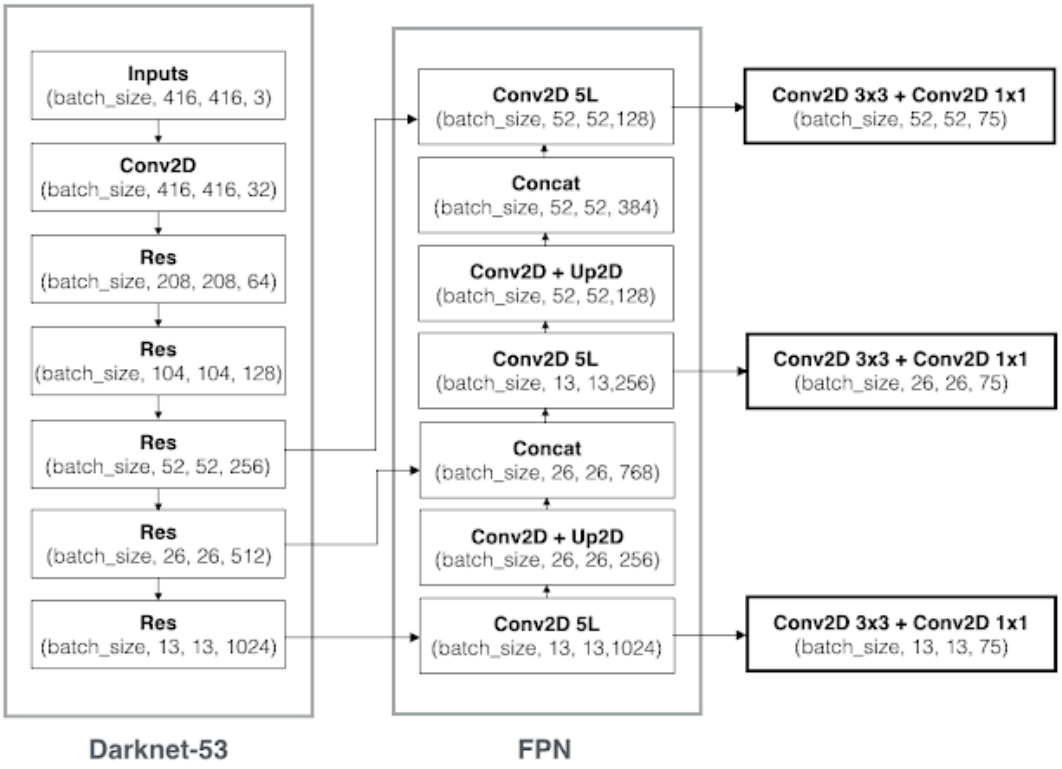


FIGURE 11. The Yolo-v3 Network Structure[4]

[4] Figure 11 Source: https://aip.cm.nsysu.edu.tw/index.php/documents/

2.4.4. Yolo-v4

Yolo-v4 (Bochkovskiy et al., 2020) made certain improvements to various parts of the network structure based on Yolo-v3. First of all, Mosaic data augmentation (Perez & Wang, 2017) was used in the input part. Secondly, it also introduced CSP module (Wang et al., 2019), SPP module, PAN architecture (Liu et al., 2018), mish activation function, etc. to change the model architecture. Compared to Yolo-v3 network, Yolo-v4 network is an efficient model that can train a fast and good model on a 1080 Ti or 2080 Ti GPU. The structure of Yolo-v4 network can be illustrated as follows:

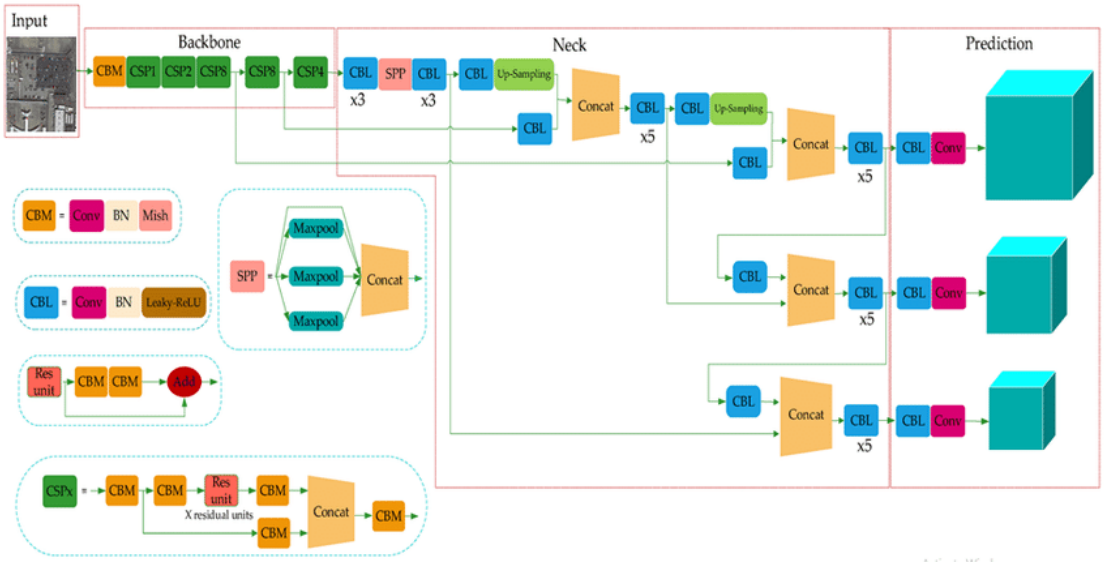


FIGURE 12. The Yolo-v4 Network Structure[5]

2.4.5. Yolo-v5

Yolo-v5 had made some improvements based on Yolo-v4. Firstly, it can adaptively adjust the anchor box. Yolo-v5 embedded this feature into the code and

[5] Figure 12 Source: https://www.researchgate.net/figure/YOLOv4-structure-diagram_fig4_357639383

adjusted the best anchor boxes in different training sets during each training session without manual adjustment. In addition, Yolo-v5 network also introduced the Focus structure to perform slicing operations on input images. The biggest innovation of Yolo-v5 network is that the author makes the network structure a configurable option. For example, the backbone network structure can be divided into versions such as Yolo-v5s, Yolo-v5m, Yolo-v5l, and Yolo-v5x based on the width and height of each network. The structure of Yolo-v5 network can be illustrated as follows:

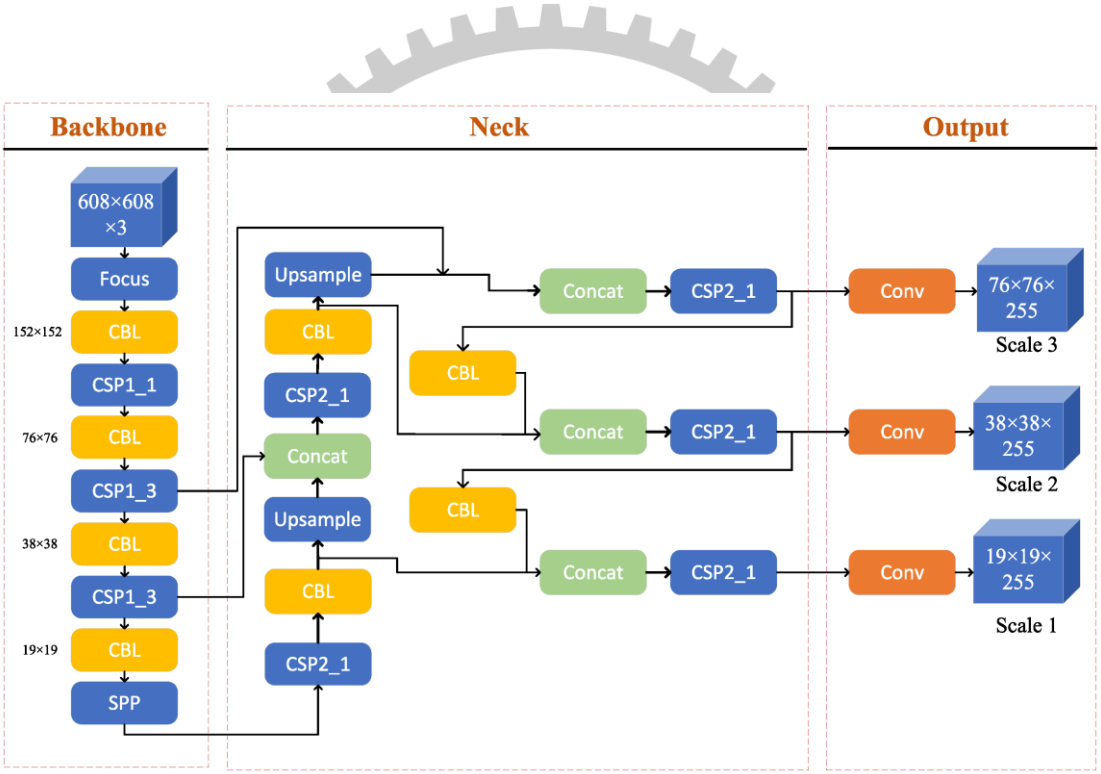


FIGURE 13. The Yolo-v5 Network Structure[6]

2.4.6. YoloX

Although YoloX network (Ge et al., 2021) has not made significant changes in the backbone and neck compared to Yolo-v5 network, it made a series of

[6] Figure 13 Source: https://www.mdpi.com/2072-4292/13/18/3776

improvements in the input layer and prediction layer. In the input layer, YoloX inherited the Mosaic data augmentation used by Yolo-v5 network, and additionally introduced MixUp data augmentation (Zhang et al., 2021). In the prediction layer, the author modified the Yolo head of the traditional Yolo series to the Decoupled head. Based on experiments, it was found that the detection head used in the Yolo series in the past may lack good expression ability. After using Decoupled head, not only did the accuracy improve, but the convergence speed of the network also accelerated.

In addition, unlike previous generations of Yolo models, YoloX adopted the concept of anchor-free, which no longer used anchor boxes to associate with ground-truth boxes. In addition, it used the concept of SimOTA label assignment strategy, which solved the problem of anchor-free training with too many negative samples, which made the network training difficult. Similar to Yolo-v5 network, the author had made the network structure configurable, and the structure of YoloX network can be illustrated as follows:

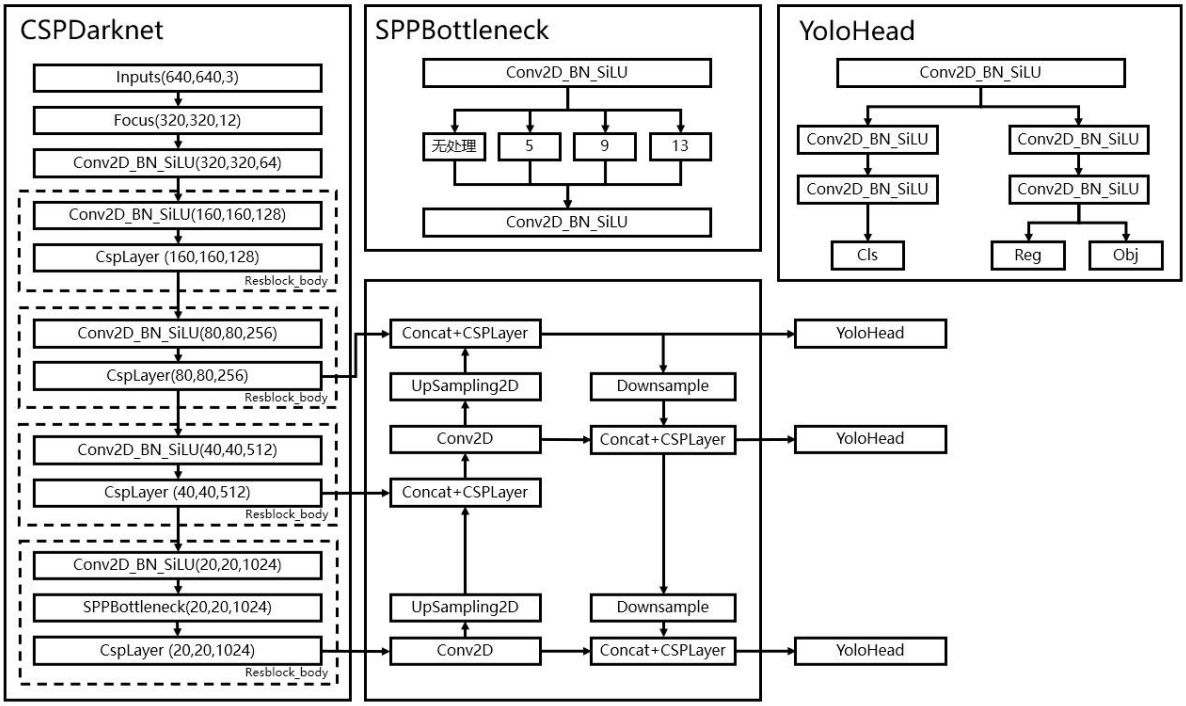


FIGURE 14. The YoloX Network Structure[7]

2.4.7. Yolo-v6

Yolo-v6 network (Li et al., 2022) retains the advantages of YoloX's anchor-free and label assignment strategies, as well as Yolo-v5's data augmentation method, and makes improvements based on this. Firstly, the backbone of Yolo-v6 network no longer used CSPDarknet, but instead used a more efficient EfficientRep (Weng et al., 2023). In the neck section, the structure was changed to Rep-PAN constructed based on Rep (Ding et al., 2021) and PAN, and the head, like YoloX network, adopted the Decoupled head and incorporated a more efficient structure, reducing the computational complexity of the Decoupled head. The structure of Yolo-v6 network (Li et al., 2022) can be illustrated as follows:

[7] Figure 14 Source: https://blog.csdn.net/weixin_44791964/article/details/120476949

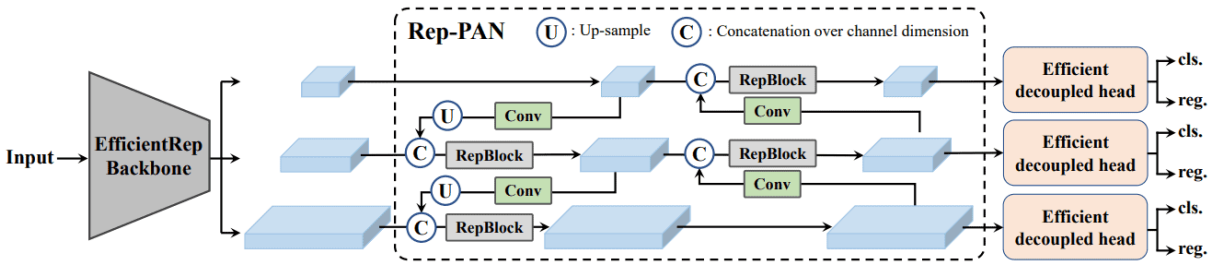


FIGURE 15. The Yolo-v6 Network Structure. Figure is taken from Li et al. (2022).

2.4.8. Yolo-v7

Yolo-v7 network (Wang et al., 2022) reduces approximately 40% of the parameters and approximately 50% of the FLOPs in the current real-time object detection State-of-the-Art field. When published, its speed and accuracy in the range of 5 FPS to 160 FPS exceeded all known object detectors.

Yolo-v7 network proposed an Extended efficient layer aggregation network for optimizing model architecture. In addition to considering the factors such as parameters and FLOPs, it also analyzes gradient paths, allowing weights of different layers to learn more diverse features. Moreover, Yolo-v7 further improves the performance of the model through the use of model re-parameterized and dynamic label assignment strategies during the training process. The structure of Yolo-v7 network can be illustrated as follows:

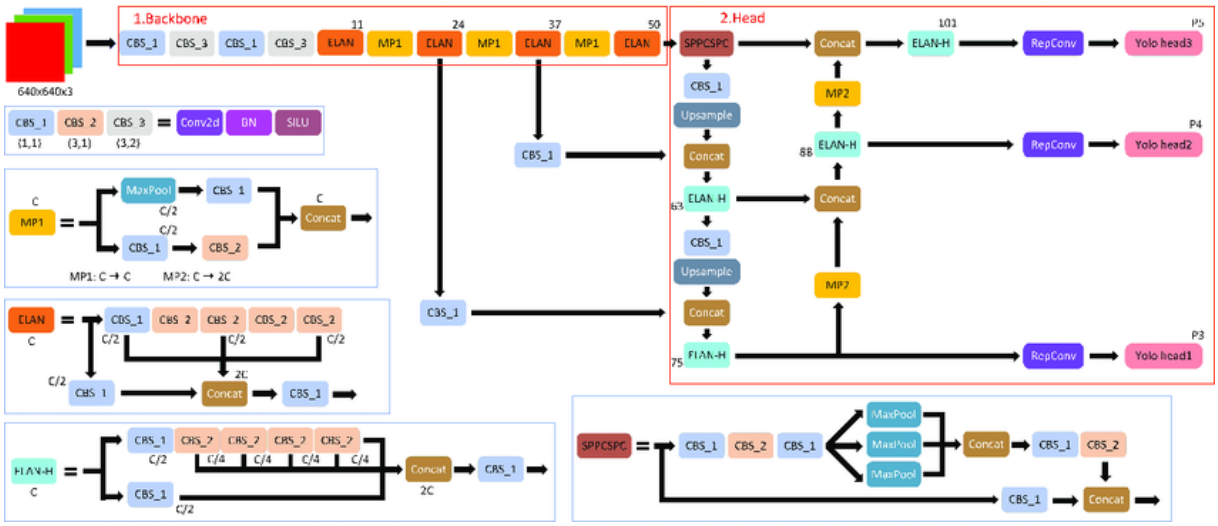


FIGURE 16. The Yolo-v7 Network Structure[8]

2.4.9. Yolo-v8

Yolo-v8 (Reis et al., 2023) network is the latest version of the Yolo series, with few innovations in itself. It mostly drew on the advantages of previous generations of Yolo models and made minor adjustments, including strategies such as anchor-free, Decoupled head, and data augmentation. One of the biggest characteristic of Yolo-v8 is its code scalability, which not only supports one algorithm, but is compatible with all previous Yolo algorithm frameworks. It can easily modify the Yolo algorithm used for training by modifying configuration parameters. The structure of Yolo-v8 network can be illustrated as follows:

[8] Figure 16 Source: https://www.researchgate.net/figure/The-structure-of-YOLOv7_fig4_365691457

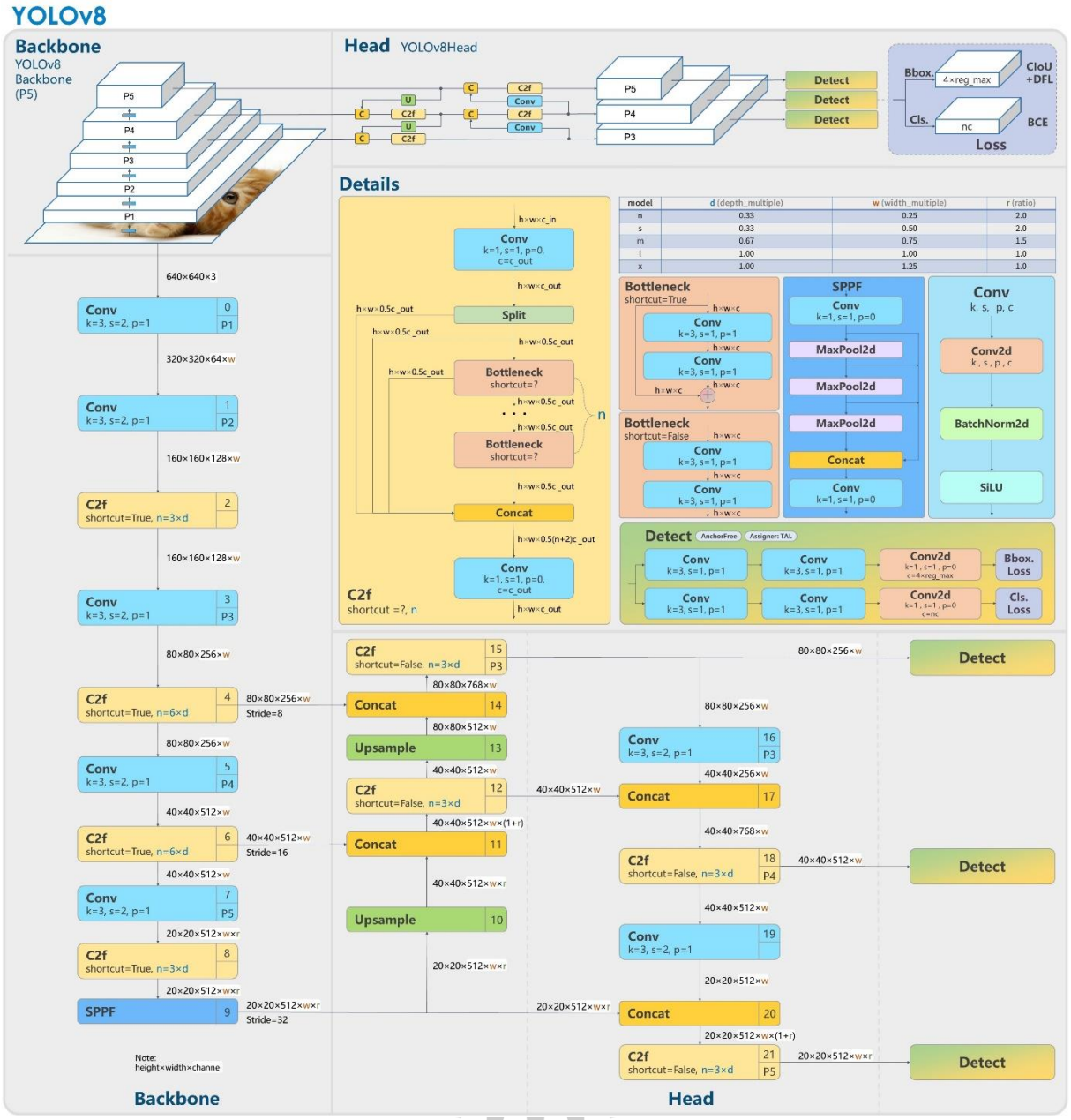


FIGURE 17. The Yolo-v8 Network Structure[9]

2.4.10. Summary

Looking back at the relevant papers in the Yolo series, it will be found that each version has its own characteristics. Yolo-v1 focused on how to make detection faster and developed the first one-stage detector. From Yolo-v2 to

[9] Figure 17 Source: https://github.com/ultralytics/ultralytics/issues/189

Yolo-v4, in order to make the model more accurate, many useful tricks were used to modify the model. In the training process, multi-scale input training strategy and anchor based method were introduced; In the network section, including the use of data augmentation methods in the input layer, batch normalization, skip connection, CSP modules, and FPN structures were introduced in the backbone and neck sections. YoloX has indicator significance, leading the Yolo series towards the era of anchor-free, and also introduced new label assignment methods to solve the problems brought by anchor-free. However, Yolo-v6 and Yolo-v7 had mainly made changes to the network structure to make the network more efficient and perform better. And Yolo-v5 and Yolo-v8 focused on code changes, making it more convenient for relevant personnel to make modifications.

The models of Yolo series have been developed for nearly a decade now, and although there are many innovative points, many models have been modified based on the well-known algorithms at that time. By standing on the shoulders of giants, we can avoid more detours in the process of algorithm development and become more efficient in improving models.

2.5. Related Models with Attention Mechanism

Attention mechanism is a commonly used technique in deep learning, although it can be implemented in various ways, its core is to make the network pay attention to the areas that it needs to pay more attention to. Generally speaking, attention mechanisms can be divided into channel attention mechanisms, spatial attention mechanisms, and a combination of the two.

In deep learning, common implementation methods of attention mechanisms include SE, CBAM, ECA, and so on. Therefore, in this section, this work will first

introduce UPANets with attention mechanisms, and then introduce these common modules with attention mechanisms one by one, and try to apply them in subsequent experiments.

2.5.1. Universal Pixel Attention Networks (UPANets)

In order to enable CNN to process global and local information, a channel pixel attention (CPA) mechanism was designed in each UPA block of UPANets (Tseng et al., 2021). UPANets are mainly composed of UPA layers, and each UPA layer is mainly composed of densely connected UPA blocks, which contain CPA modules. The CPA modules can help feature maps fuse more complex feature maps without losing the original features. The following section describes the components of a UPA block and a UPA layer:

UPA Blocks

According to Wide ResNet (Zagoruyko & Komodakis, 2016), the combination of two 3×3 convolutional layers provides the most reliable accuracy, so the CNN in each UPA block is composed of two 3×3 convolutional layers. There are two types of UPA blocks, stride 1 version is used when the CNN output size is the same as the x input, then the x goes through a CPA consisting of MLP and batch normalization, and after adding with the CNN output and doing layer normalization (Ba et al., 2016), the final output does densely-connection with the feature maps from the previous blocks.

When the output size of the CNN is different from the input x, the Stride 2 version is used, and then both the x and CNN outputs go through a CPA, after adding the two CPA outputs and doing the layer normalization, and finally using

the average pooling. The UPA block structure (Tseng et al., 2021) can be illustrated as follows:

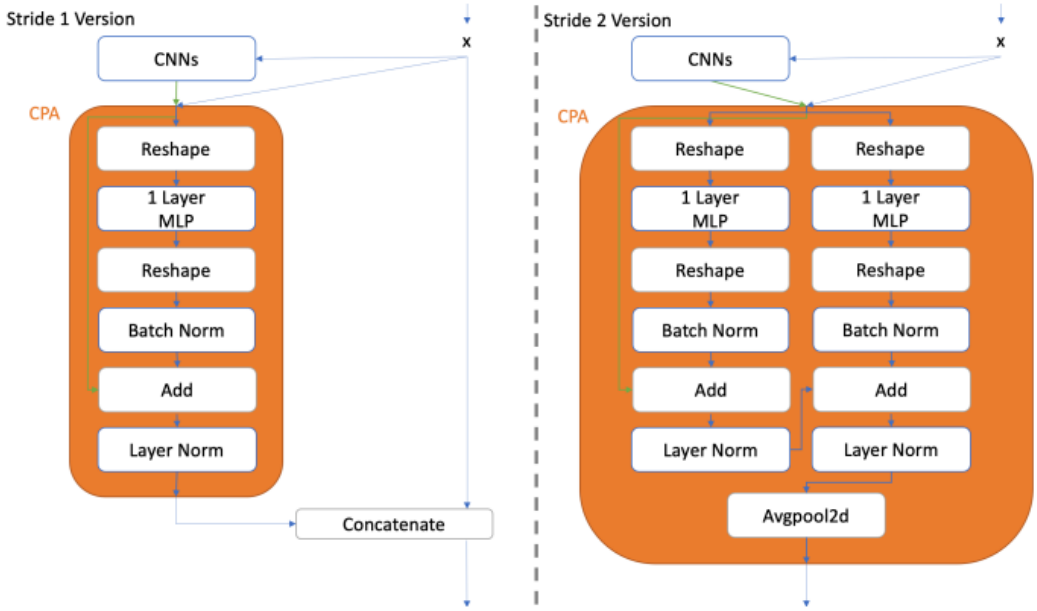


FIGURE 18. The UPA Block Structure. Figure is taken from Tseng et al. (2021).

UPA Layers

A UPA layer is composed of several UPA blocks. Except for the first UPA block that follows the Stride 2 operation, all the other UPA blocks are Stride 1 blocks, which means that the second to last UPA blocks will undergo a concatenate operation after passing through CPA. By focusing on important pixels through the CPA in the UPA blocks, this model can further improve the learning performance of the image. The UPA layer structure (Tseng et al., 2021) can be illustrated as follows:

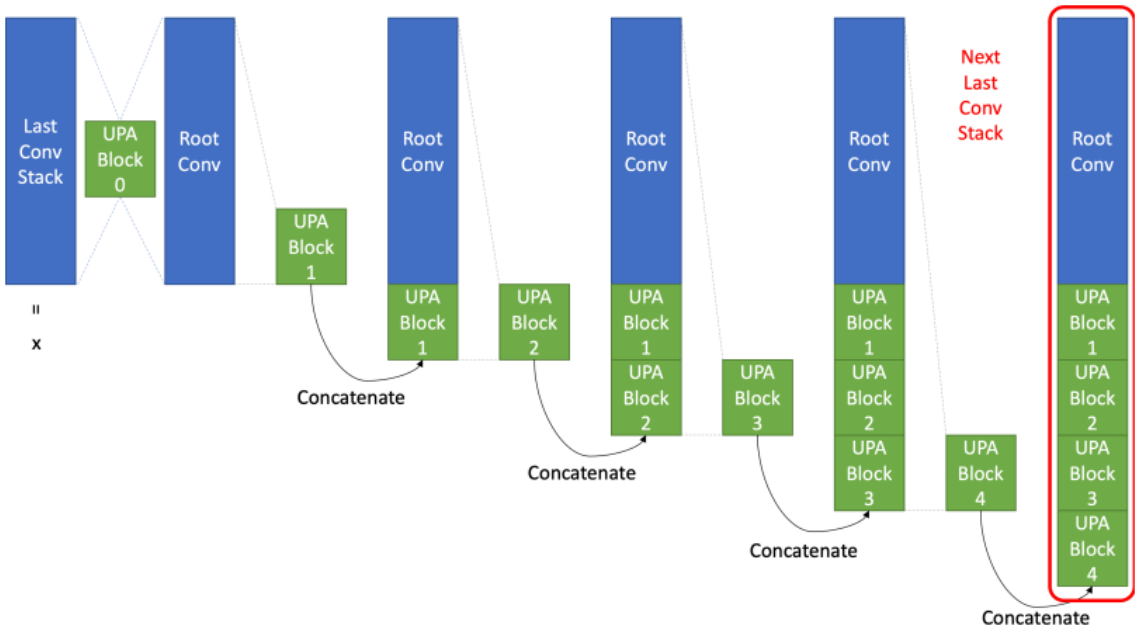


FIGURE 19. The UPA Layer Structure. Figure is taken from Tseng et al. (2021).

2.5.2. Squeeze-and-Excitation Networks (SENet)

SENet (Hu et al., 2017) is a typical implementation of channel attention mechanism. The SENet proposed in 2017 was the champion of the last ImageNet competition. The basic idea of SENet is to focus on the weight of each channel for the input feature layer, so that the network can focus on the channel it needs to pay attention to the most.

Its specific implementation method is to first perform global average pooling on the input feature layer, and then feed it into two fully-connected layers. The first time, there are fewer fully-connected neurons, and the second time, the number of fully-connected neurons is the same as the input feature layer. After completing two fully-connected layers, we take sigmoid function to fix the value between 0 and 1. At this point, we obtain the weights of each channel in the input feature layer (between 0 and 1). After obtaining these weights, we multiply them

by the original input feature layer. The structure of SE block (Hu et al., 2017) can be illustrated as follows:

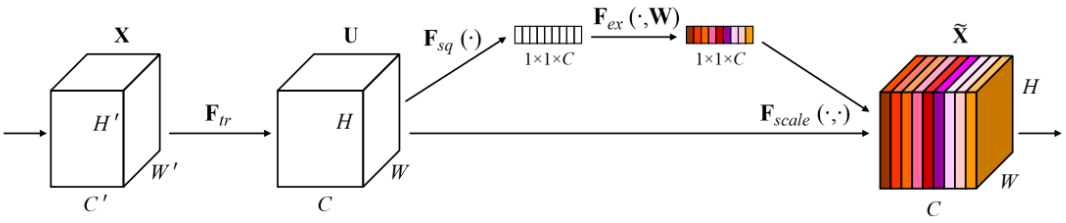


FIGURE 20. The Structure of SE Block. Figure is taken from Hu et al. (2017).

2.5.3. Convolutional Block Attention Module (CBAM)

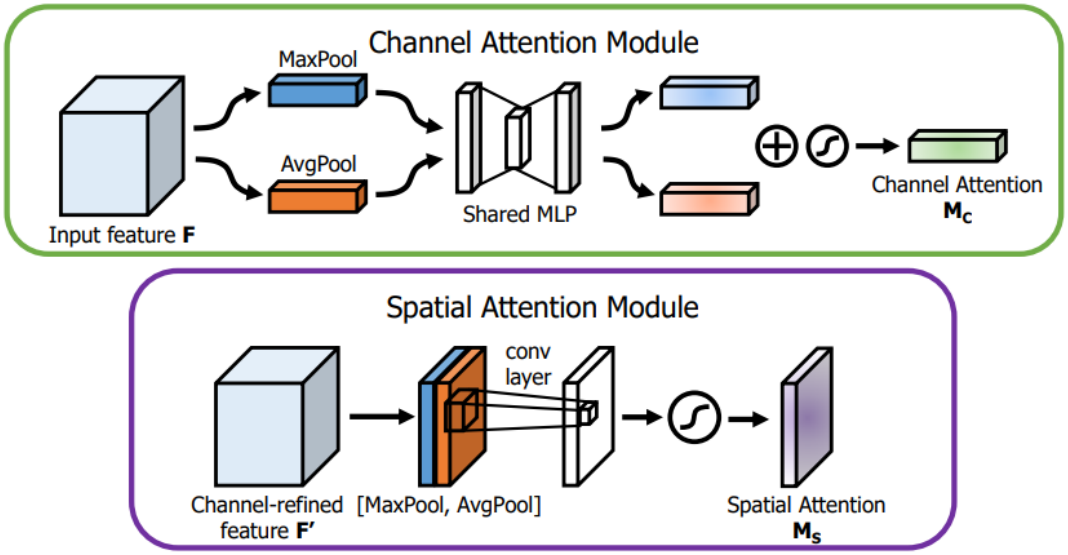
Compared to SENet, which only focused on channel attention mechanism, CBAM (Woo et al., 2018) combined the channel attention mechanism with the spatial attention mechanism. CBAM processed the input feature layer through channel attention mechanism and spatial attention mechanism, respectively.

In the section of channel attention mechanism, we will perform global average pooling and global maximum pooling on the input feature layer, respectively. Afterwards, the results of average pooling and maximum pooling are processed using shared fully-connected layers. We add the two processed results, take a sigmoid function, and finally obtain the weights of each channel in the input feature layer (between 0-1).

And the spatial attention mechanism, we will take the maximum and average values on the channel of each feature point on the input feature layer. Afterwards, we stack these two results and adjust the number of channels by a convolutional layer with one channel. Then, we take a sigmoid function, and obtain the weights

of each feature point in the input feature layer (between 0 and 1). After obtaining these weights, we multiply them by the original input feature layer. The CBAM process and structure of CBAM (Woo et al., 2018) can be illustrated as follows:

(a)



(b)

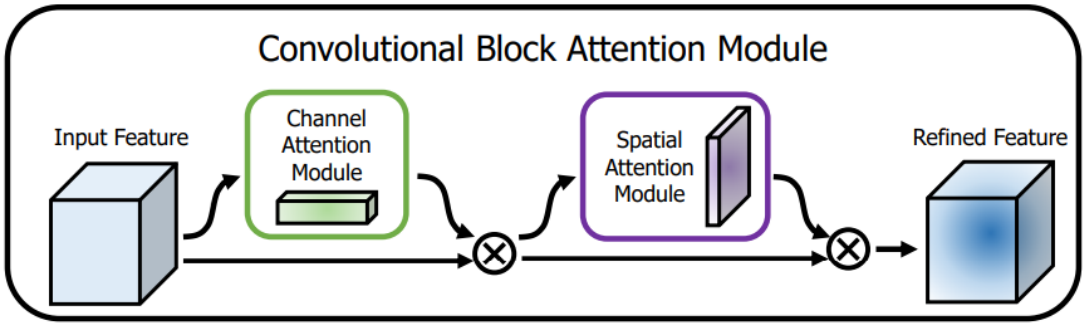


FIGURE 21. (a) The structure of CBAM. (b) The CBAM process. Figure is taken from Woo et al. (2018).

2.5.4. Efficient Channel Attention Networks (ECANet)

ECANet (Wang et al., 2019) is also an implementation form of channel

attention mechanism, and its author believes that SENet is inefficient and unnecessary in capturing the dependencies of all channels, so it has been improved. In ECANet paper, the author believes that the convolutional layer has good cross channel information acquisition ability.

The idea of the ECA module is very simple, as it removes the fully-connected layers from the original SE block feeds the feature layer after global average pooling into the 1D convolutional layer for learning. Since 1D convolution is used, the selection of kernel size for 1D convolution becomes very important, as its size affects the number of channels to be considered in calculating each weight of the attention mechanism. The structure of ECA module (Wang et al., 2019) can be illustrated as follows:

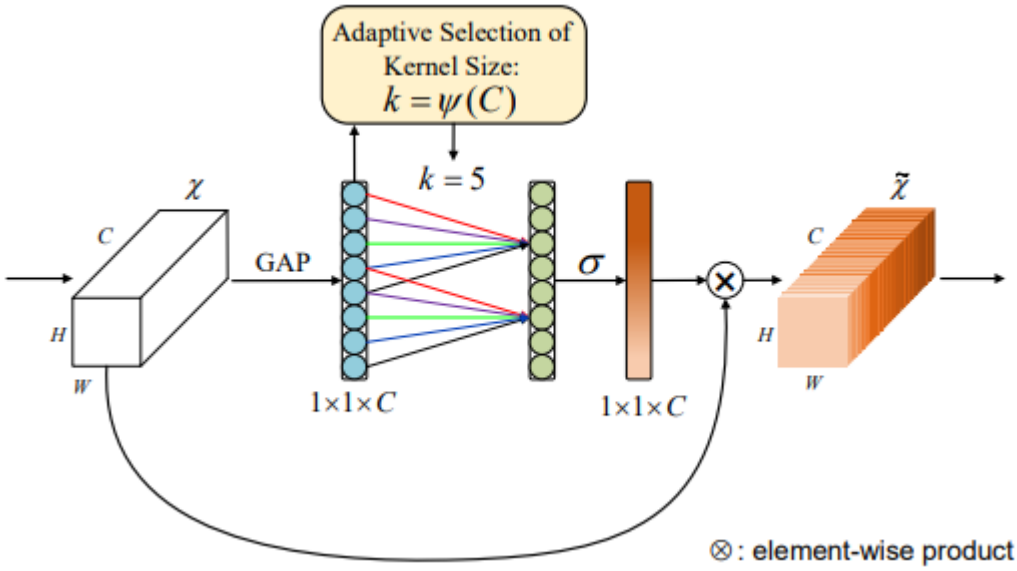


FIGURE 22. The Structure of ECA Module. Figure is taken from Wang et al. (2019).

2.6. Summary

In this chapter, after reviewing the changes in object detection field from its

early stages to the present, it is truly a very complex field. In addition, according to the different times, they have their own problems to solve. For example, the traditional object detection methods need to conceive effective feature representation, while the object detection methods in the deep learning era, although having the advantages of computer computing power, they still need to consider how to build a more efficient network structure, even more in line with the current field of loss function, in order to pursue excellence in the current development. Even though object detection field has been developed for over 20 years, there is still great room for development based on the performance of COCO datasets commonly used for evaluating the model ability in object detection. Perhaps in the future, computer capabilities will reach new heights, and it may bring about another stage of growth.

In addition, after familiarizing with the models related to the attention mechanism, this work will combine them with the Yolo model and propose several improved network structures. For detailed information, please refer to Chapter 3. Finally, the performance of the model will be evaluated through the indicator mAP, and the best model will be summarized.

3. Proposed Models

Firstly, this work choose the YoloX model of the Yolo series as our experimental target for two main reasons:

(1) The YoloX network is mainly optimized for the input layer and detection head, and lacks modifications in the backbone and neck parts of the network. Therefore, adjustments are needed at these positions.

(2) The YoloX model officially brought the Yolo series into the era of anchor-free, and subsequent models no longer require the use of anchor boxes to generate prediction boxes, which has indicator significance.

Therefore, in this chapter, this work will propose three model modifications for the YoloX network. The first one is to modify the backbone part of the network, and the latter two are to optimize the neck part. After running the proposed models and its mAP score, this work will compare the results and come up with the most robust modified model.

3.1. Replace the Backbone Part of YoloX Network

Although the YoloX network has better performance in mAP compared to previous generations of Yolo models, it still has the problem of low contribution rate per unit parameter. To address this issue, adding modules to the existing CSPDarknet backbone may further improve the performance of mAP, but there is still a problem of the model being too cumbersome. In addition, inspired by UPANets, through the operation of CPA in UPANets, the feature map output by CPA can effectively combine the original feature itself and useful information from other features. Compared with deep network structures, CPA helps shallow

networks more easily form complex patterns. Therefore, this work attempts to apply UPANets with CPA modules to the backbone of the YoloX network, and makes some lightweight modifications in the details to achieve the original expectations.

First Type - YoloX with Lite UPANets Backbone

The proposed Lite UPANets consist of two parts. The upper part first feeds the input image into two convolutional layers, and then going through the CPA module to do a down-sampling process to obtain the result of the first stage. The lower part is composed of UPA layers, and the result obtains in the first stage are processed by four UPA layers before the model enters the Neck section. The structure of Lite UPANets can be illustrated as follows:

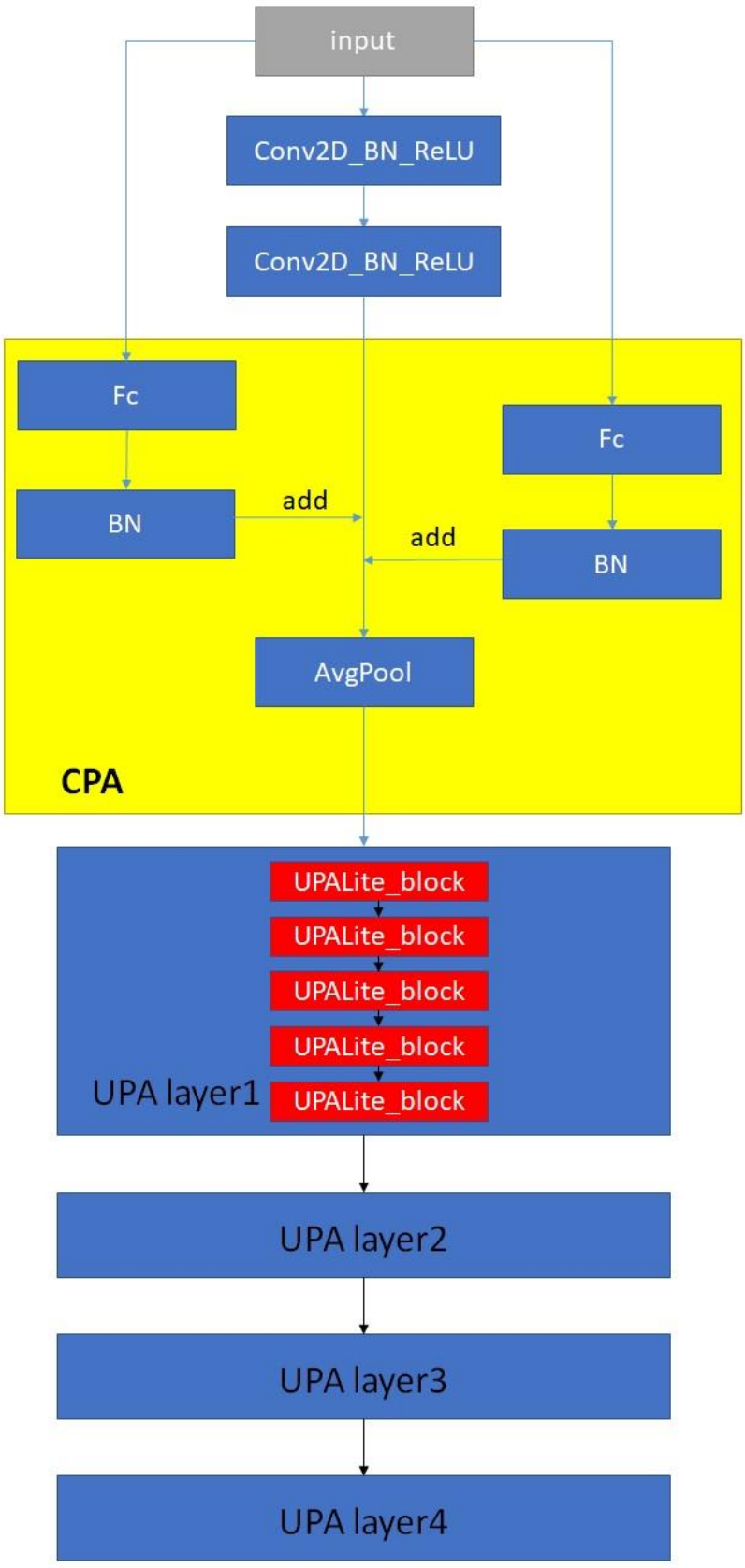


FIGURE 23. The Structure of Lite UPANets

As described in Chapter 2, each UPA layer is composed of 5 UPALite blocks, and each UPALite block contains a CPALite module. The first UPALite block is responsible for down-sampling the feature map, while the second to fifth UPALite blocks are responsible for increasing the number of channels in the feature map. The main purpose of the CPA module before the UPA layer and the CPALite module in the UPA layer is to effectively fuse the feature maps of different layers, and the CPA module will perform an additional fusion. The structure of UPA layer can be illustrated as follows:

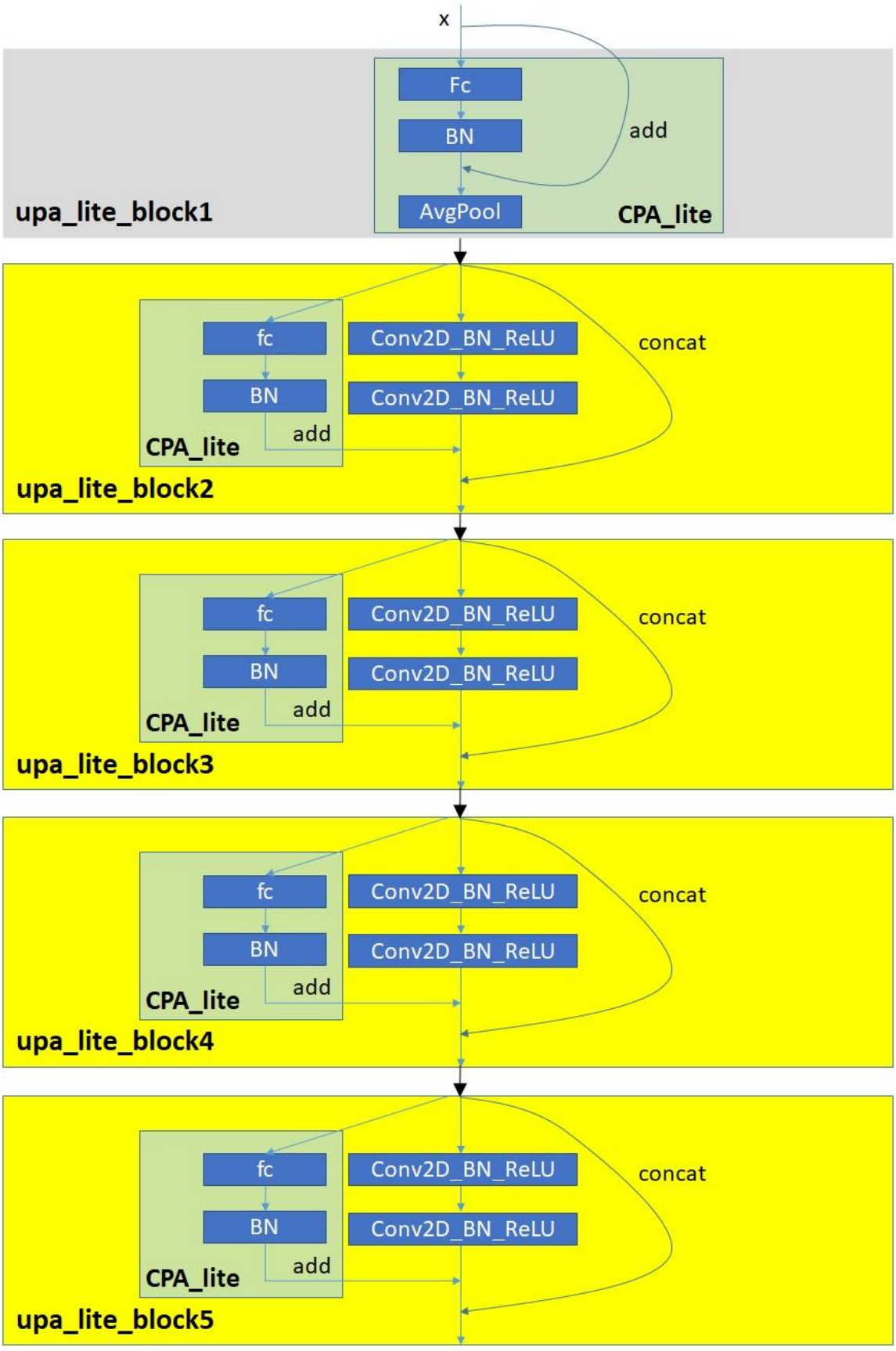


FIGURE 24. The Structure of UPA Layer

Combining Lite UPANets with our original YoloX network is the first model this work proposed. Considering that YoloX networks have multiple variants, this work will propose three versions of Lite UPANets, namely Lite UPANets-64, Lite UPANets-32, and Lite UPANets-16 to correspond to YoloX-l, YoloX-s, and YoloX-nano. The structure of these three versions of the model is roughly the same, mainly due to differences in the number of output channels in the convolutional layer and the fully connected layer, which also results in a difference of approximately 40 times in the parameters of these three models. This is to compare whether there will be any differences in the performance of Lite UPANets between the large model and the small model. The structure of YoloX with Lite UPANets backbone can be illustrated as follows:

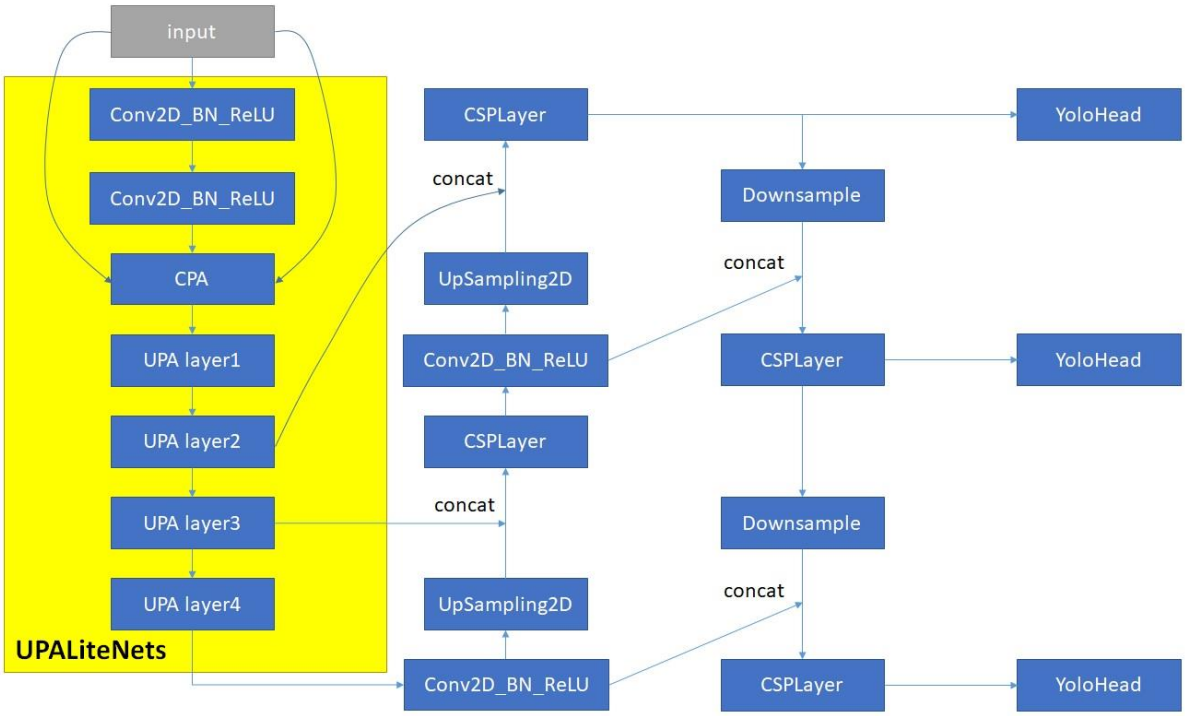


FIGURE 25. The Proposed Model 1 - YoloX with Lite UPANets Backbone

3.2. Optimize the Neck Part of YoloX Network

After improving the backbone part of the YoloX network, this work aims to further optimize the neck part. Due to the use of lightweight Lite UPANets in the backbone network to replace the original model's CSPDarknets, it is foreseeable that the performance will decrease with a certain degree of parameter reduction. Therefore, this work would like to add some attention mechanism modules, while adding a few parameters, this work can also slightly improve the model's performance, reducing the gap between the proposed model and the original model in mAP scores.

Through the related work in Chapter 2, we know that there are three commonly used small modules with attention mechanisms to improve model performance, namely SE, CBAM, and ECA. Although these attention mechanism modules are plug and play that can theoretically be placed behind any feature layer, the effects brought by placing them in different parts of the network must be different. Therefore, in this section, I will add these modules to different parts of the Neck, and finally compare the effects of adding different modules in different areas during the experimental stage.

3.2.1. Second Type - Adding Attention Module outside FPN Based on the First Type Model

Since this work wants to evaluate the effect of adding attention mechanisms between the backbone and FPN Neck layer, this work chooses to add attention mechanism modules as our second model on the two feature layers that are

extracted from the backbone and will soon enter the FPN Neck for concatenation with other feature layers. As there are a total of three attention mechanisms, there will be three variants of the model here. The structure of model after adding attention mechanisms outside FPN can be illustrated as follows:

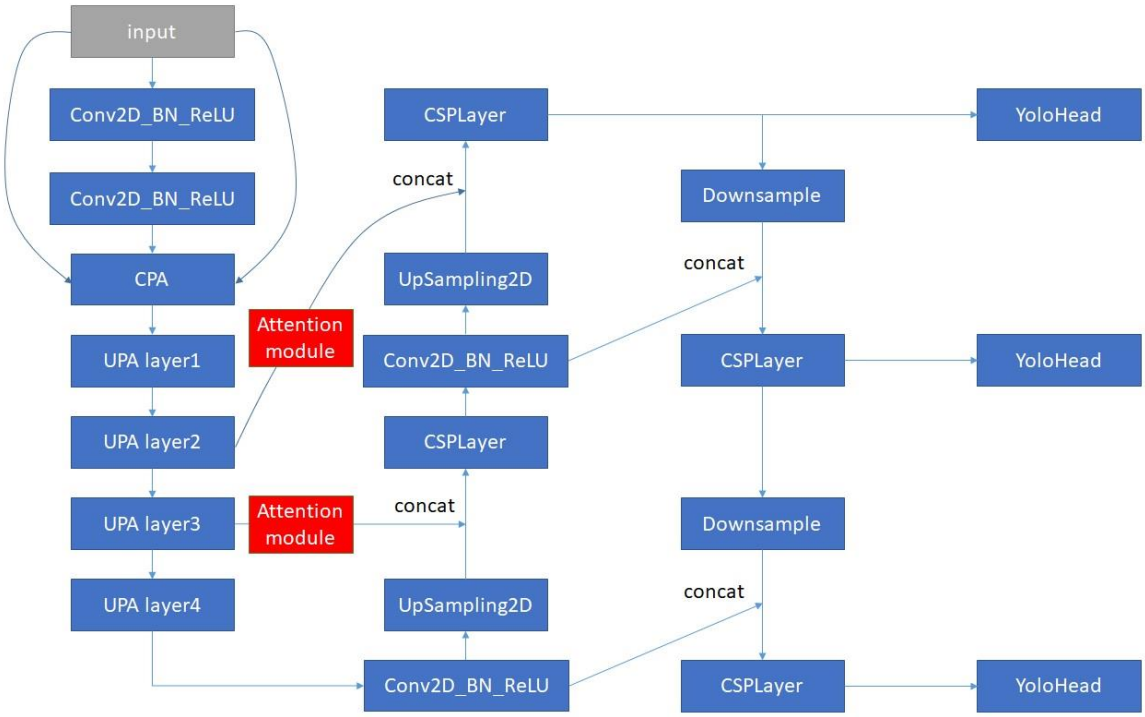


FIGURE 26. The Proposed Model 2 - Adding Attention Module outside FPN Based on the First Type Model

3.2.2. Third Type - Adding Attention Module in FPN Based on the Second Type Model

Next, this work wants to further evaluate the changes bring about by adding attention mechanisms in FPN. For the third model, this work adds attention mechanisms modules to the two feature layers within FPN that are about to be concatenated with other feature layers. Since there are three types of attention

mechanisms, there are also three variants of the model here, and the structure of model after adding the attention mechanisms within FPN can be illustrated as follows:

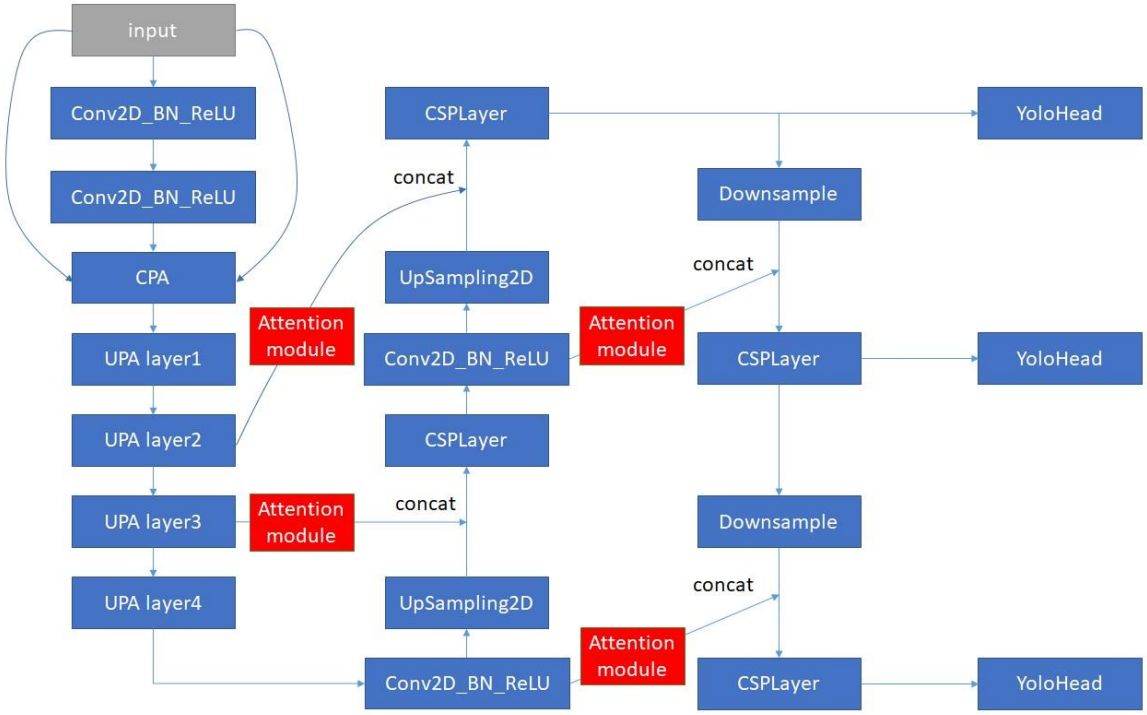


FIGURE 27. The Proposed Model 3 - Adding Attention Module in FPN Based on the Second Type Model

4. Experiments

4.1. Dataset and Experimental Description

After evaluating the experimental equipment and time cost, PASCAL VOC-07 is used as our dataset in this experiment. This work uses the preset splitting of the VOC-07 dataset to split the data into training sets, validation sets, and testing sets. The training and validation sets contain 2501 and 2510 images, respectively, while the testing sets contain 4951 images. The training and validation sets are used to train the most robust model, while the testing sets are used to evaluate the performance of the model. All experiments are deployed on a 24GB RTX Titan, and due to the lack of pre-trained weights, the model needs to be trained from scratch until convergence.

As mentioned in the previous chapter, in order to evaluate the performance of the proposed models under different network scales, this experiment will use three variants of the YoloX network, namely YoloX-l, YoloX-s, and YoloX-nano, as the experimental group, and compare the three models this work proposed as the control groups. The following tables show the differences between the original model and the three proposed models:

Table 1. The comparison of the structure and parameter quantities between the **YoloX-nano** model and the three proposed models.

Backbone	Between Backbone and Neck	Neck	Parameters
YoloX-nano CSPDarknet	**N/A**	FPN	900,459
The First Type Model			

Lite UPANet16	**N/A**	FPN	815,803
The Second Type Model			
Lite UPANet16	SE	FPN	818,363
Lite UPANet16	CBAM	FPN	826,239
Lite UPANet16	ECA	FPN	815,811
The Third Type Model			
Lite UPANet16	SE	FPN+ SE	820,923
Lite UPANet16	CBAM	FPN+ CBAM	836,675
Lite UPANet16	ECA	FPN+ ECA	815,819

Table 2. The comparison of the structure and parameter quantities between the **YoloX-s** model and the three proposed models.

Backbone	**Between Backbone and Neck**	**Neck**	**Parameters**
YoloX-s CSPDarknet	**N/A**	FPN	8,945,035
The First Type Model			
Lite UPANet32	**N/A**	FPN	6,227,435
The Second Type Model			
Lite UPANet32	SE	FPN	6,237,675
Lite UPANet32	CBAM	FPN	6,268,591
Lite UPANet32	ECA	FPN	6,227,445
The Third Type Model			
Lite UPANet32	SE	FPN+ SE	6,247,915
Lite UPANet32	CBAM	FPN+	6,309,747

		CBAM	
Lite UPANet32	ECA	FPN+ ECA	6,227,455

Table 3. The comparison of the structure and parameter quantities between the **YoloX-l** model and the three proposed models.

Backbone	Between Backbone and Neck	Neck	Parameters
YoloX-l CSPDarknet	**N/A**	FPN	54162635
The First Type Model			
Lite UPANet64	**N/A**	FPN	33054603
The Second Type Model			
Lite UPANet64	SE	FPN	33095563
Lite UPANet64	CBAM	FPN	33218639
Lite UPANet64	ECA	FPN	33054613
The Third Type Model			
Lite UPANet64	SE	FPN+ SE	33136523
Lite UPANet64	CBAM	FPN+ CBAM	33382675
Lite UPANet64	ECA	FPN+ ECA	33054623

The above three tables show the differences in structure and parameter quantities between the three variants of the YoloX network and the proposed models. The first model only makes changes to the backbone network, while the second and third models add different attention mechanism modules at different positions in Neck based on the previous model. In terms of parameter quantity,

our proposed model has little difference from the original model in the case of the small model, with about 90% of the original model. However, as the network expands, the difference becomes more significant, and in the case of the large model, the parameter quantity is only about 60% of the original model.

4.2. Mean Average Precision (mAP)

In order to evaluate the error between the predicted bounding boxes and the ground-truth boxes, Mean Average Precision (mAP) is a commonly used indicator in the field of object detection to measure the quality of the model. The mAP formula is not complex, but it involves the concept of Confusion Matrix and IoU indicators. Next, this work will introduce them one by one:

Confusion Matrix

The typical confusion matrix is a table with two rows and two columns, which can be divided into four situations based on ground truth and model prediction: TP, TN, FP, and FN. Taking the field of object detection as an example:

True Positive (TP): This model detects an object and correctly matches the ground truth.

True Negative (TN): This model doesn't detect an object and matches the ground truth.

False Positive (FP): This model detects an object, but the object doesn't exist (Type I Error).

False Negative (FN): The model doesn't detect an object, but the object exists (Type II Error).

The composition of the confusion matrix can be illustrated as follows:

		Predicted	
		Negative **(N)** -	Positive **(P)** +
Actual	Negative -	True Negative **(TN)**	**False Positive (FP) Type I Error**
	Positive +	**False Negative (FN) Type II Error**	True Positive **(TP)**

FIGURE 28. The Confusion Matrix

In the field of object detection, what we usually care about is how many objects predicted by the models actually exist. Therefore, what we are concerned about is the Precision indicator. The formula for Precision is as follows:

$$Precision = \frac{\text{TP}}{\text{TP+FP}} \tag{1}$$

Intersection over Union (IoU)

The Intersection over Union (IoU) indicates the degree of overlap between the predicted bounding box and the ground-truth box. A higher IoU indicates that the predicted bounding box coordinates are very similar to the ground-truth box coordinates, thus showing that the prediction is more accurate. The formula for IoU is as follows:

$$IoU = \frac{\text{area of overlap}}{\text{area of union}} \tag{2}$$

The following figure can briefly illustrate the relationship between prediction accuracy and IoU. It is obvious that the IoU in the right image is higher, therefore the accuracy of the prediction is also higher.

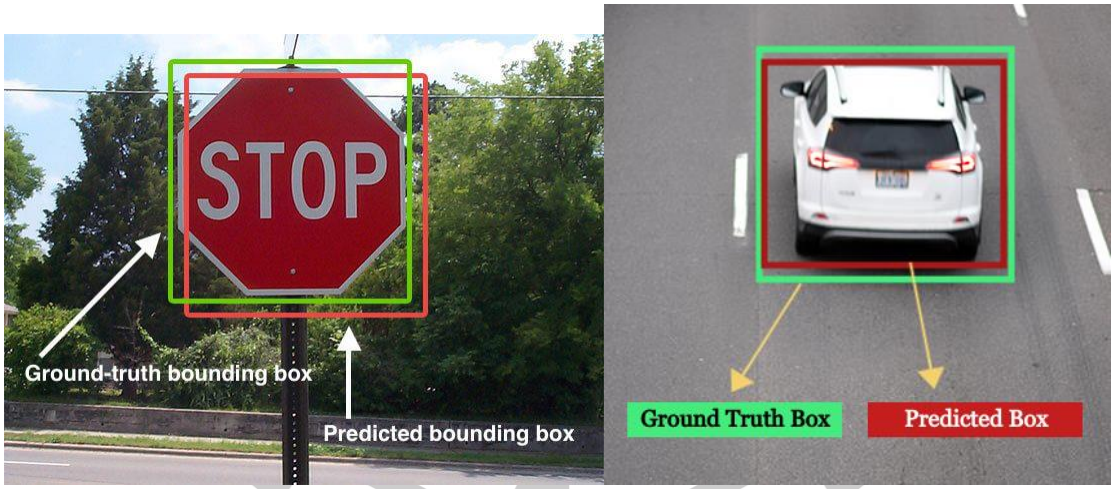


FIGURE 29. The Comparison of Different IoU

With the above two indicators, the concept of mAP is to set an IoU threshold, usually set 0.5 (Henderson & Ferrari, 2016). If the IoU between the predicted bounding box and the ground-truth box is greater than 0.5, it's TP, and vice versa, it is FP. Therefore, the mAP is calculated by finding the Average Precision (AP) of each class and then average over all classes. The formula for mAP is as follows:

$$\text{mAP} = \frac{1}{N}\sum_{i=1}^{N} AP_i \qquad (3)$$

where i indicates the i-th class and N indicates the number of all classes.

4.3. Discussion of Experimental Result

After running the proposed models in Chapter 3, this work obtains the

following conclusions: firstly, this work finds that replacing the backbone from CSPDarknet to Lite UPANets has better performance, i.e., it maintains about the same performance with much fewer parameters, implying that the CPA is actually effective in mixing the feature maps of different layers and conveying useful information to the subsequent layers.

In addition, comparing the impact of modules with different attention mechanisms on model performance, it can be found that compared to SE and CBAM, ECA seems unable to further improve the model. The analysis may be due to the use of 1D convolution in ECA, which may not be suitable for image processing, or the failure to select a suitable kernel size, resulting in lower performance than expected. Finally, comparing SE with CBAM, CBAM show higher performance growth, possibly due to the additional consideration of spatial attention within CBAM, thus maximizing mAP scores compared to SE with additional useful information. The experimental results of the three models are shown in the tables below:

Table 4. The comparison of experimental results between the **YoloX-nano** model and the three proposed models.

Model	**mAP0.5 score**	**Parameters**	**The proportion of parameter quantity to the original model**
YoloX-nano	**44.31%**	900,459	100%
The First Type Model			
Lite UPANet16	**44.77%**	815,803	90.60%

The Second Type Model			
Lite UPANet16 + SE	45.14%	818,363	90.88%
Lite UPANet16 + CBAM	**45.15%**	826,239	91.76%
Lite UPANet16 + ECA	44.71%	815,811	90.60%
The Third Type Model			
Lite UPANet16 + SE + SE	45.20%	820,923	91.17%
Lite UPANet16 + CBAM + CBAM	**45.27%** **(+0.96%)**	836,675	92.92%
Lite UPANet16 + ECA + ECA	44.80%	815,819	90.60%

Table 5. The comparison of experimental results between the **YoloX-s** model and the three proposed models.

Model	**mAP0.5 score**	**Parameters**	**The proportion of parameter quantity to the original model**
YoloX-s	**52.13%**	8,945,035	100%
The First Type Model			
Lite UPANet32	**48.32%**	6,227,435	69.62%
The Second Type Model			
Lite UPANet32 + SE	48.91%	6,237,675	69.73%
Lite UPANet32 + CBAM	**49.26%**	6,268,591	70.08%
Lite UPANet32 + ECA	48.42%	6,227,445	69.62%
The Third Type Model			
Lite UPANet32 + SE + SE	49.35%	6,247,915	69.85%

Lite UPANet32 + CBAM + CBAM	**50.05%** **(-2.08%)**	6,309,747	70.54%
Lite UPANet32 + ECA + ECA	48.28%	6,227,455	69.62%

Table 6. The comparison of experimental results between the **YoloX-l** model and the three proposed models.

Model	**mAP0.5 score**	**Parameters**	**The proportion of parameter quantity to the original model**
YoloX-l	**58.72%**	54,162,635	100%
The First Type Model			
Lite UPANet64	**51.45%**	33,054,603	61.03%
The Second Type Model			
Lite UPANet64 + SE	51.82%	33,095,563	61.10%
Lite UPANet64 + CBAM	**52.10%**	33,218,639	61.33%
Lite UPANet64 + ECA	51.57%	33,054,613	61.03%
The Third Type Model			
Lite UPANet64 + SE + SE	52.13%	33,136,523	61.18%
Lite UPANet64 + CBAM + CBAM	**52.74%** **(-5.98%)**	33,382,675	61.63%
Lite UPANet64 + ECA + ECA	51.6%	33,054,623	61.03%

Comparing the differences between our proposed models in large and small models, it can be found that our proposed models achieve better parameter contribution rate, and even achieve results with lower parameter quantity but

better accuracy in the small model section. Although the final performance of the medium model and the large model is still worse than the original model, it is speculated that it is due to a significant reduction in the number of parameters, which is only about 70% and 60% of the original model, respectively. If the parameter quantity can be increased to about 90% as in the case of the small model, there is a possibility of outperforming the original model. Anyway, our proposed model demonstrates the effectiveness of Lite UPANets and attention mechanism modules CBAM and SE, which may be applied to other fields in the future.

4.4. Summary

This work uses YoloX network as the main structure, and attempt to reduce the number of parameters while maintaining the performance of the model by replacing the backbone network and adding attention mechanism modules. From the experimental results, it can be seen that our proposed Lite UPANets meet our expectations, proving that CPA can effectively mix information from different feature layers. Therefore, through CPA, the model can achieve the same level with fewer parameters. After comparing all the attention mechanism modules, it is proven that they have a generally positive impact on the performance of the model, with CBAM performing the best, followed by SE, and ECA having no significant positive impact. It is speculated that CBAM, compared to SE, contains additional spatial attention mechanisms, which can transmit more effective information to subsequent layers, and ECA may be due to the 1D convolution, the choice of the kernel size is more sensitive, or due to the fact that 1D convolution itself is not suitable for application in the image field.

In addition, comparing the generalization ability of our proposed model

between large and small models, all of them have better parameter contribution rate compared with the original model, and even achieve better performance with fewer parameters in the small model section, proving that CPA can perform effectively regardless of the model size. However, if the accuracy is to be maintained at a similar level, the number of parameters in the model should not be too much less than the original model, otherwise the performance may be too poor and not as expected.

5. Conclusion

In this paper, this work first reviews the relevant papers on object detection and find that there exists a problem of the model being too cumbersome, so this work wants to solve it. In addition, inspiring by UPANets and attention mechanisms, this work modifies the YoloX model to more effectively utilize parameters by replacing the backbone network and adding different attention mechanism modules at different locations in the network. Finally, three different types of models are proposed. According to our research results, replacing the original backbone network CSPDarknet with Lite UPANets can effectively improve the parameter contribution rate, and adding attention mechanisms to different parts of the Neck can also slightly improve the model performance. Moreover, this work also discusses some possible reasons why some attention modules may not perform as expected:

(1) SE is not as good as CBAM due to its lack of spatial attention mechanism.

(2) ECA is not conducive to image learning due to its inclusion of 1D convolution and the more sensitive setting of kernel size.

On the other hand, experiment shows that the performance of the model at different network sizes has a certain level, with little difference from the original model. However, it is still important to note that the parameter quantities of the proposed model and the original model should not differ too much, otherwise it may cause the problem of CPA not being able to effectively carry enough information, leading to poor performance.

5.1. Research Limitation

Due to the hardware limitations, considering time costs, this work is unable to perform on other larger datasets (such as the COCO dataset). In a compromise, this work can only choose the relatively smaller PASCAL VOC-07 dataset. In addition, because the backbone network is replaced in this experiment, there are no weights available for pre-trained on ImageNet, resulting in all models being trained from scratch. Even if the model converges after training, it is still inferior to the model with pre-trained weights in accuracy, so it is impossible to compare with the model on the leaderboard, but it could give a general trend.

5.2. Future Work

Although Lite UPANets with CPA modules have better parameter contribution rate compared to the original YoloX network, they require about twice the time spent on training compared to the YoloX model. This means that there are parts that need to be optimized in structure, perhaps due to the impact of excessive concatenate operations, which is a topic that can be studied.

The other direction is that to improve the parameter contribution rate more effectively, this work can try to study how to apply CPA to the prediction head of the YoloX model, change it to a lighter and more efficient structure or let CPA focus on objects that are difficult to detect. However, it may involve the redesign of the loss function. How to maintain the model performance when the prediction head is changed is another important issue. We hope to apply these ideas and techniques to model modification in the near future.

Reference

Ba, J., Kiros, J.R., & Hinton, G.E. (2016). Layer Normalization. ArXiv, abs/1607.06450.

Bochkovskiy, A., Wang, C., & Liao, H.M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. ArXiv, abs/2004.10934.

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), 1, 886-893 vol. 1.

Ding, X., Zhang, X., Ma, N., Han, J., Ding, G., & Sun, J. (2021). RepVGG: Making VGG-style ConvNets Great Again. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 13728-13737.

Evgeniou, T., & Pontil, M. (2001). Support Vector Machines: Theory and Applications. Machine Learning and Its Applications.

Felzenszwalb, P.F., Girshick, R.B., McAllester, D.A., & Ramanan, D. (2010). Object Detection with Discriminatively Trained Part Based Models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 32, 1627-1645.

Felzenszwalb, P.F., McAllester, D.A., & Ramanan, D. (2008). A discriminatively trained, multiscale, deformable part model. 2008 IEEE Conference on Computer Vision and Pattern Recognition, 1-8.

Ge, Z., Liu, S., Wang, F., Li, Z., & Sun, J. (2021). YOLOX: Exceeding YOLO Series in 2021. ArXiv, abs/2107.08430.

Girshick, R.B. (2015). Fast R-CNN. 2015 IEEE International Conference on Computer Vision (ICCV), 1440-1448.

Girshick, R.B., Donahue, J., Darrell, T., & Malik, J. (2013). Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. 2014 IEEE Conference on Computer Vision and Pattern Recognition, 580-587.

He, K., Zhang, X., Ren, S., & Sun, J. (2014). Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 37, 1904-1916.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770-778.

Henderson, P., & Ferrari, V. (2016). End-to-End Training of Object Class Detectors for Mean Average Precision. ArXiv, abs/1607.03476.

Hosang, J.H., Benenson, R., & Schiele, B. (2017). Learning Non-maximum Suppression. 2017 IEEE Conference on Computer Vision and Pattern

Recognition (CVPR), 6469-6477.
Hu, J., Shen, L., Albanie, S., Sun, G., & Wu, E. (2017). Squeeze-and-Excitation Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 42, 2011-2023.
Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. International Conference on Machine Learning.
Krizhevsky, A., Sutskever, I., & Hinton, G.E. (2012). ImageNet classification with deep convolutional neural networks. Communications of the ACM, 60, 84 - 90.
Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Ke, Z., Li, Q., Cheng, M., Nie, W., Li, Y., Zhang, B., Liang, Y., Zhou, L., Xu, X., Chu, X., Wei, X., & Wei, X. (2022). YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications. ArXiv, abs/2209.02976.
Lin, T., Dollár, P., Girshick, R.B., He, K., Hariharan, B., & Belongie, S.J. (2016). Feature Pyramid Networks for Object Detection. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 936-944.
Lin, T., Goyal, P., Girshick, R.B., He, K., & Dollár, P. (2017). Focal Loss for Dense Object Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 42, 318-327.
Liu, S., Qi, L., Qin, H., Shi, J., & Jia, J. (2018). Path Aggregation Network for Instance Segmentation. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 8759-8768.
Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S.E., Fu, C., & Berg, A.C. (2015). SSD: Single Shot MultiBox Detector. European Conference on Computer Vision.
Louppe, G. (2014). Understanding Random Forests: From Theory to Practice. arXiv: Machine Learning.
O'Shea, K., & Nash, R. (2015). An Introduction to Convolutional Neural Networks. ArXiv, abs/1511.08458.
Perez, L., & Wang, J. (2017). The Effectiveness of Data Augmentation in Image Classification using Deep Learning. ArXiv, abs/1712.04621.
Redmon, J., & Farhadi, A. (2016). YOLO9000: Better, Faster, Stronger. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 6517-6525.
Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. ArXiv, abs/1804.02767.
Redmon, J., Divvala, S.K., Girshick, R.B., & Farhadi, A. (2015). You Only Look

Once: Unified, Real-Time Object Detection. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 779-788.

Reis, D., Kupec, J., Hong, J., & Daoudi, A. (2023). Real-Time Flying Object Detection with YOLOv8. ArXiv, abs/2305.09972.

Ren, S., He, K., Girshick, R.B., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39, 1137-1149.

Schmidt, R.M. (2019). Recurrent Neural Networks (RNNs): A gentle Introduction and Overview. ArXiv, abs/1912.05911.

Tseng, C., Lee, S., Feng, J., Mao, S., Wu, Y., Shang, J., Tseng, M., & Zeng, X. (2021). UPANets: Learning from the Universal Pixel Attention Neworks. Entropy, 24.

Uijlings, J.R., Sande, K.E., Gevers, T., & Smeulders, A.W. (2013). Selective Search for Object Recognition. International Journal of Computer Vision, 104, 154-171.

Viola, P.A., & Jones, M.J. (2001). Rapid object detection using a boosted cascade of simple features. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, 1, I-I.

Wang, C., Bochkovskiy, A., & Liao, H.M. (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. ArXiv, abs/2207.02696.

Wang, C., Liao, H.M., Yeh, I., Wu, Y., Chen, P., & Hsieh, J. (2019). CSPNet: A New Backbone that can Enhance Learning Capability of CNN. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 1571-1580.

Wang, Q., Wu, B., Zhu, P.F., Li, P., Zuo, W., & Hu, Q. (2019). ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 11531-11539.

Weng, K., Chu, X., Xu, X., Huang, J., & Wei, X. (2023). EfficientRep: An Efficient Repvgg-style ConvNets with Hardware-aware Neural Network Design. ArXiv, abs/2302.00386.

Woo, S., Park, J., Lee, J., & Kweon, I. (2018). CBAM: Convolutional Block Attention Module. European Conference on Computer Vision.

Zagoruyko, S., & Komodakis, N. (2016). Wide Residual Networks. ArXiv, abs/1605.07146.

Zhang, H., Cissé, M., Dauphin, Y., & Lopez-Paz, D. (2017). mixup: Beyond

Empirical Risk Minimization. ArXiv, abs/1710.09412.
Zou, Z., Shi, Z., Guo, Y., & Ye, J. (2019). Object Detection in 20 Years: A Survey. Proceedings of the IEEE, 111, 257-276.