Crowd Counting and People Density Detection: An Overview

Fuqiang Jin[1,a], Zhaoguo Zhang[2,b], Yi Ning[3,c], Yi Lu[2,d], Wei Song[2,e], Xingguo Qin[4,f], *Jinlong Chen[1,g]

[1]Guilin University of Electronic Technology, School of Computer Science and Information Security, Guilin, China
[2]Guangxi Tourism Technology Corporation Limited, Guilin, China
[3]Guilin University of Electronic Technology, School of Continuing Education, Guilin, China
[4]Guilin University of Electronic Technology , School of Computer Science and Information Security, Guilin, China

[a] jin13781958725@163.com, [b]zhangzhaoguo@gxota.com , [c]296106092@qq.com, [d] lucky@gxota.com, [d] bobklarke@qq.com,[f]315859742@qq.com, [g]Jinlong.chen@guet.edu.cn

Abstract. Crowd counting and crowd density estimation are important tasks in the field of computer vision, involving the detection of people in images or videos and the estimation of crowd density. In this domain, deep learning algorithms such as Convolutional Neural Networks (CNNs) and You Only Look Once (YOLO) play crucial roles. CNNs are specialized deep learning models for processing image data. They extract features and learn representations from images through multiple convolutional layers, pooling layers, and fully connected layers. CNNs have achieved remarkable success in tasks such as image classification, object detection, and semantic segmentation, providing effective solutions for problems like head counting and crowd density estimation. On the other hand, YOLO is a fast and accurate object detection algorithm. Its unique feature is the ability to predict multiple bounding boxes and their corresponding class probabilities in a single forward pass. YOLO divides the image into smaller grid cells and performs bounding box and class predictions in each cell, enabling efficient object detection. In summary, crowd counting and crowd density estimation present significant challenges in computer vision. With the help of deep learning algorithms like CNNs and YOLO, researchers can address these challenges more accurately, providing powerful technical support for applications in crowd management, security surveillance, and other fields.

Keywords: Crowd Counting; People Density Detection; Convolutional Neural Networks; Training and Optimization Methods; YOLO

T. Yao et al. (eds.), *Proceedings of the 2024 3rd International Conference on Engineering Management and Information Science (EMIS 2024)*, Advances in Computer Science Research 111,
https://doi.org/10.2991/978-94-6463-447-1_46

1 Introduction

The relationship between deep learning and object detection is among the most significant advancements in the field of modern computer vision. Deep learning, particularly Convolutional Neural Networks (CNNs), has provided a robust framework for computers to recognize and locate objects within images. This progress is largely attributed to the ability of deep learning models to learn high-level feature representations from vast amounts of complex data.Before the advent of deep learning, object detection relied primarily on manual feature extraction and shallow learning models[1-3], which had limited capabilities for complex visual scene recognition. However, with the introduction of deep neural networks, especially CNNs, the performance of object detection has seen substantial improvements[4-5]. CNNs can automatically extract useful features from images, deepening the abstraction of features layer by layer, thereby achieving accurate object detection across varied scenes and conditions.Moreover, advancements in deep learning for object detection have included the introduction of Region Proposal Networks (RPNs) and anchor frameworks, innovations that allow detection models to not only identify objects within images but also precisely define their boundaries. This series of technological advancements, such as Faster R-CNN, YOLO (You Only Look Once), and SSD (Single Shot MultiBox Detector), have significantly propelled the development of object detection, making it applicable to various domains including autonomous driving, facial recognition, and surveillance systems[6-7].

Improved models Crowd-CNN[8], Multi-column CNN[9], Scale-Aware Attention Networks[10] and LSC-CNN[11] based on CNN have played an important role in crowd counting and human density detection. effect. Especially under complex conditions, crowd counting will cause great interference, and it is also a point that the visual direction attaches great importance to. Eliminating the interference caused by complex conditions so as not to affect the correct results is also a point that visual direction inspiration needs to break through.

In the field of crowd counting and human density estimation, the YOLO algorithm can be combined with other technologies to improve the accuracy and efficiency of crowd counting and density estimation. YOLO can be used to detect individual objects in images and therefore can be used for crowd counting. By detecting the location of people in an image and then counting the detected people, a YOLO-based crowd counting system can be implemented. This method can be applied to scenarios such as surveillance cameras, people counting and activity management. Another approach is to combine YOLO with density estimation techniques to estimate the density of crowds. Density estimation techniques typically use regression methods to predict the density of each pixel in an image and associate these density maps with the people detected by YOLO. This combination can provide more detailed crowd distribution information, including crowd density maps and detected number of people.

This article will briefly outline the main developments in the field of crowd counting and human density detection from three aspects: optimization and regularization algorithms, based on convolutional neural network models and YOLO; Chapter 2 mainly discusses optimization algorithms and regularization methods. . Such as

StepLR, MultiStepLR, Dropout, L1, Early Stopping, SinCLR, Batch Normalization, etc.; Chapter 3 mainly starts from the deep convolutional network based on crowd counting and human density detection. A brief overview of the architecture of different network models; Chapter 4 understands the development history of the target detection model YOLO, which is currently in hot research; Chapter 5's experimental part first briefly outlines some commonly used image classification data sets, and then mainly trains some on the data sets Classic network, and gives data for comparison; Chapter 6 briefly summarizes and discusses the current status and future development trends of crowd counting and human density detection.

2 Train and Optimize Methods

In crowd counting and human density detection, optimization and regularization techniques play a crucial role. The optimization algorithm accelerates model convergence, helps to find the optimal solution during the training process, and avoids local optimal points. Regularization technology plays a role in preventing overfitting and constraining model complexity,making the model more generalizable. When dealing with crowded scenes, the combination of these two technologies can enable the model to better adapt to different data distributions and scene changes, thereby improving the accuracy and stability of crowd counting and density estimation.

2.1 Learning Rate Scheduling

StepLR.

StepLR is used to reduce the learning rate in a stepwise manner during training. After each specified epoch or batch, the current learning rate is multiplied by a specified factor gamma, thereby reducing the learning rate. This stepped attenuation strategy helps the model better converge to the optimal solution.

MultiStepLR.

Used to adjust the learning rate in a multi-step manner during training. At prespecified milestones, the current learning rate is multiplied by a specified gamma factor, thereby reducing the learning rate. This helps the model gradually reduce the learning rate during training to achieve more stable convergence and better performance.

ExponentialLR.

It reduces the learning rate exponentially. After each epoch, it multiplies the current learning rate by a specified decay factor gamma to achieve exponential decay of the learning rate.

LinearLR.

It reduces the learning rate in a linear fashion. After each epoch or each batch, it gradually decreases the current learning rate at a predefined rate until it reaches the minimum learning rate or the end of training.

CyclicLR.

CyclicLR is used to dynamically adjust the learning rate. The core concept is to cyclically adjust the learning rate within a specified range to increase the model's adaptability to different learning rates. During each training epoch, the learning rate gradually increases from a small initial value to a large maximum value, and then gradually decreases back to the initial value. This method helps the model jump out of the local optimal solution and better explore the global optimal solution.

OneCycleLR.

OneCycleLR is designed to improve model performance by dynamically adjusting the learning rate throughout the training process. The core idea is to gradually increase the learning rate to a maximum value during the training process, and then gradually reduce it back to the initial value, forming a cycle. Such a scheduling method can help speed up the convergence of the model, improve the generalization ability of the model, and can effectively deal with under-fitting and over-fitting problems during the training process.

CosineAnnealingLR.

CosineAnnealingLR uses cosine annealing to dynamically adjust the learning rate. During the training process, the learning rate will be periodically adjusted according to the change pattern of the cosine function, first declining rapidly and then gradually stabilizing. This scheduling method can help the model search the parameter space more smoothly during the training process, help avoid falling into local minima, and improve the generalization ability of the model.

CosineAnnealingWarmRestarts.

CosineAnnealingWarmRestarts adds a periodic restart mechanism based on CosineAnnealingLR. During the training process, the learning rate is periodically adjusted in a cosine annealing manner, and the learning rate is restarted at the end of each cycle. This scheduling method can help the model more fully search the parameter space and restart training after the learning rate is restarted, which can help jump out of the local optimal solution and improve the convergence speed and performance of the model.

LambdaLR.

LambdaLR dynamically adjusts the learning rate based on user-defined lambda functions. Users can define the changing rules of the learning rate through the lambda function, such as exponential decay, linear decay, etc. LambdaLR allows users to

freely design how the learning rate changes based on training progress, number of epochs, or other custom conditions, thereby more flexibly adjusting the model training process.

ChainedScheduler.

ChainedScheduler allows users to connect multiple learning rate schedulers in chain order to apply different learning rate scheduling strategies during the training process. The design of this scheduler allows users to combine and adjust multiple learning rate scheduling strategies as needed, thereby optimizing the model training process more flexibly.

ConstantLR.

ConstantLR is used to maintain a constant learning rate during training. Unlike other schedulers, ConstantLR does not adjust the size of the learning rate, but always keeps it constant. This means that the learning rate of the model will remain at a fixed value throughout the training process and will not change as training progresses. ConstantLR is suitable for situations where the learning rate does not need to be dynamically adjusted, or when performing some specific experiments, the learning rate needs to be kept constant.

ReduceLROnPlateau.

ReduceLROnPlateau is a commonly used learning rate scheduling algorithm, mainly used to monitor the performance of the model on the verification set. The core idea is to automatically reduce the learning rate when the model performance stops improving. By reducing the learning rate, the model can tune parameters more finely, helping to overcome local minima and converge to a better global minimum. The use of ReduceLROnPlateau can improve the stability and performance of the model and accelerate the convergence process of the model.

2.2 Regularization

Dropout.

Dropout regularization is a technique commonly used in deep learning to reduce overfitting and improve the generalization ability of the model. It prevents the model from over-relying on specific neurons by randomly discarding a part of the neurons in the neural network during the training process, making the model more robust and generalizable. In each training batch, Dropout randomly sets a portion of neurons to 0 to reduce the complexity of the network. Such operations help prevent the model from overfitting on the training data and improve the model's ability to adapt to new data. During the testing phase, no dropout operation is performed, but the weights are scaled to maintain the desired output. The introduction of Dropout not only helps to improve the generalization ability of the model, but also speeds up the training process and reduces computing costs.

Early Stopping.

Early Stopping serves as a powerful regularization strategy, commonly employed to curb the overfitting of deep learning models throughout their training phase. At its heart, this method involves tracking the model's efficacy on a validation set as training progresses, and halting the training process once there's no further enhancement in the performance on the validation set. This approach effectively prevents the model from overfitting.

L1 regularization.

L1 regularization is a regularization method used for machine learning and deep learning models. Its purpose is to control the size of model parameters to reduce overfitting and improve the generalization ability of the model. This method is implemented by adding the L1 norm of the parameters (i.e., the sum of the absolute values of the parameters) as a penalty term in the loss function. Such a setting makes the model more inclined to generate sparse parameter vectors during the training process, that is, compressing some parameters to zero, thus reducing the complexity of the model.

L2 regularization.

L2 regularization is designed to mitigate overfitting and enhance the model's ability to generalize by moderating the magnitude of the model parameters. It achieves this by incorporating the L2 norm of the parameters (the sum of their squared values) into the loss function as a penalty term. This approach encourages the model to favor smaller parameter values throughout training, thereby simplifying the model's complexity and bolstering its generalization capabilities to new, unseen data.

2.3 Batch Normalization

Batch normalization[12] is a technique designed to enhance both the convergence rate and the generalization capabilities of neural networks. Its fundamental concept involves normalizing each batch of data prior to its entry into each network layer. This process adjusts the data such that its mean approaches zero and its standard deviation approaches one. Specifically, batch normalization computes the mean and variance for each batch of data, then normalizes this data by subtracting the mean and dividing by the standard deviation, resulting in a standard distribution. Subsequently, this normalized data undergoes a linear transformation through adjustable scaling and shifting parameters, maintaining the model's ability to express diverse functions. This approach helps mitigate issues related to vanishing and exploding gradients, enhances the numerical stability of the network, and boosts model convergence speed. By enabling higher learning rates, it further accelerates training. Additionally, batch normalization serves as a form of regularization, aiding in the prevention of model overfitting.

3 Deep Neural Networks

The foundation of deep learning is established by Deep Neural Networks (DNN), which leverage multiple hidden layers for training through the backpropagation algorithm. As discriminative models, they are prevalently employed in image classification tasks. This chapter delves into the domain of crowd counting by introducing several convolutional neural network (CNN) models designed for this purpose. Specifically, it explores the Crowd CNN[8], which focuses on cross-scene crowd counting, and the Multi-column CNN[9], engineered for counting individuals within a single image using a multi-column architecture. Additionally, it discusses the integration of scale-aware attention mechanisms through the Scale-aware Attention Network (SAN)[10], and the LSC-CNN[11], both of which are tailored for estimating the number of individuals in densely populated scenes. These models collectively illustrate the evolution of deep learning techniques in the realm of crowd counting.

3.1 Crowd CNN

As shown in Figure 1, This image depicts the architecture of a Crowd CNN neural network designed for image processing tasks. From left to right, the image initially passes through two convolutional layers (Conv1 and Conv2). Conv1 contains 72 filters, each with a size of 7x7, while Conv2 comprises 32 filters, each measuring 5x5. The data then flows through a pooling layer (Pool), which serves to reduce the spatial dimensions of the data, outputting 64 feature maps.This is followed by a flattening operation that transforms the two-dimensional feature maps into a one-dimensional vector. This vector is then passed to a fully connected layer (FC) with 1024 nodes, which is typically used to learn non-spatial features.Subsequently, the data passes through a Rectified Linear Unit (ReLU)[12], an activation function that introduces nonlinearity into the network. The data is then fed into another fully connected layer, this time with 128 nodes.Finally, the network employs a Softmax classifier, which converts the input into a probability distribution representing the likelihood of the image belonging to each possible category[13]. After the Softmax layer, we obtain a vector containing the predicted probabilities for each category. In the end, the sum of all predicted probabilities equals 1, denoted by Σ1.0. This architecture is commonly used for image recognition and classification tasks.

Crowd CNN[8] is a convolutional neural network tailored for crowd counting. It leverages deep learning to extract complex features from images, enabling effective estimation of crowd density across various scenes.

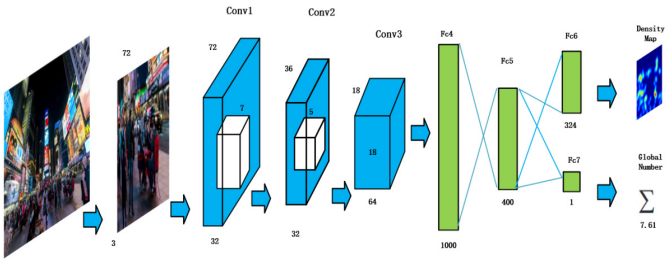

Fig. 1. The structure of the Crowd Convolutional Neural Network. At the loss layer, a density map loss and a global count loss are alternatively minimized.

3.2 Multi-column CNN

The overall structure is shown in Figure 2,This image depicts a Multi-column Convolutional Neural Network (Multi-column CNN) model used to process an input image, typically for tasks such as crowd counting. The model features a branched architecture with multiple parallel columns, each employing convolutional layers with filters of varying sizes to capture diverse features.Each column starts with a convolutional layer equipped with filters of different sizes, followed by a pooling layer to reduce spatial dimensions and widen the field of view. Within each column[14], convolutional layers with progressively smaller filter sizes extract features at various levels of detail.The features from all columns are then merged, undergoing additional convolutional layers for further feature integration. This multi-scale processing is vital for accurately interpreting scenes with varying crowd densities.Ultimately, the integrated features are processed through a regression layer to generate a density map. This map signifies the count and distribution of individuals in the input image, as indicated by the color bar on the right, where warmer colors correspond to areas of higher density.Multi-column CNN captures multi-scale features using filters of various sizes[15], effectively handling crowd counting across densities, enhancing model adaptability and accuracy.

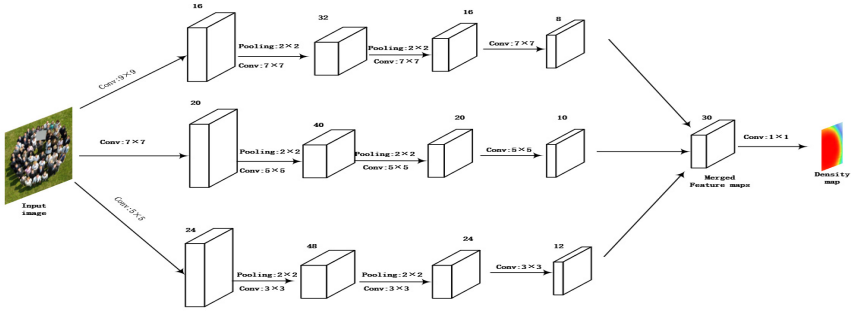

Fig. 2. The architecture of the proposed multi-column convolutional neural network for crowd density map estimation.

3.3 Scale-Aware Attention Networks

The main working principle of this model is that under complex conditions, photos will pass through three sub-network modules: multi-scale feature extractor (MFE), global scale attention (GSA) and local scale attention (LSA). The overall structure is shown in Figure 3. Since objects in images may have multiple scales, multi-scale feature extractors can effectively capture and represent features of various scales. By extracting features at different scales, the model can better understand the spatial relationships and structures between objects, thereby improving the understanding of image semantics. Global-scale attention distributes attention weights to all locations in the entire input sequence or image, rather than favoring specific local regions[16]. Doing so takes global information into account, allowing the model to better understand the overall context. In contrast, local-scale attention focuses attention weights on local areas of the input sequence or image, ignoring global information and paying more attention to local details. This image depicts a Multi-column Convolutional Neural Network (Multi-column CNN) model used to process an input image, typically for tasks such as crowd counting[17]. The model features a branched architecture with multiple parallel columns, each employing convolutional layers with filters of varying sizes to capture diverse features.Each column starts with a convolutional layer equipped with filters of different sizes, followed by a pooling layer to reduce spatial dimensions and widen the field of view. Within each column, convolutional layers with progressively smaller filter sizes extract features at various levels of detail.The features from all columns are then merged, undergoing additional convolutional layers for further feature integration. This multi-scale processing is vital for accurately interpreting scenes with varying crowd densities.Ultimately, the integrated features are processed through a regression layer to generate a density map. This map signifies the count and distribution of individuals in the input image, as indicated by the color bar on the right, where warmer colors correspond to areas of higher density[18].

3.4 LSC-CNN

LSC-CNN contains three key parts, and the overall structure is shown in Figure 4. First, the feature extractor extracts features at multiple resolutions. Almost all CNN object detectors rely on a core deep feature extraction network, whose performance is directly affected by feature quality. For crowd counting, VGG-16 based networks are widely used and provide near-state-of-the-art performance. Therefore, this model also uses the convolutional layer of VGG-16 to enhance crowd feature extraction. However[19], multi-scale feature representation faces a challenge, that is, high-resolution feature maps lack sufficient context to accurately distinguish groups of people, which may lead to misclassification. The TFM module is responsible for processing top-down features for person detection, and each scale branch is equipped with a TFM network. This top-down feature processing helps to accurately locate people in the spatial and scale pyramid. In GWTA, the input sequence is divided into blocks and assigned weights, controlled by a gating mechanism.descent with momentum. The

GWTA structure empowers the network to dynamically adjust the significance of each block to better capture long-term dependencies within the sequence. For model evaluation, the GWTA training module is substituted with a prediction fusion operation, amalgamating the loss and prediction outcomes of GWTA to estimate the head count.

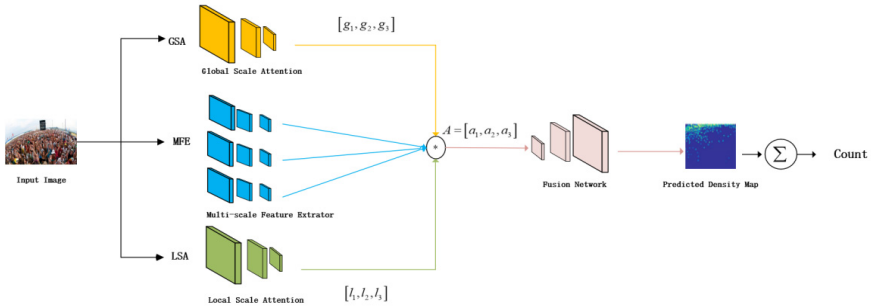

Fig. 3. Presents a methodology that employs a trio of specialized sub-networks: the Multi-Scale Feature Extractor (MFE), Global Scale Attentions (GSA), and Local Scale Attentions (LSA), all of which concurrently process an input image. The MFE's role is to produce a set of feature maps across three scales, while the GSA and LSA are designed to compute a triplet of global scores and to craft detailed local attention maps on a pixel-by-pixel basis, respectively. Post-calculation, the multi-scale feature maps are refined using the data from GSA and LSA outputs and are subsequently merged within a sophisticated fusion network. The resultant density map, created by the network, facilitates the calculation of the crowd count through the aggregation of its elements.

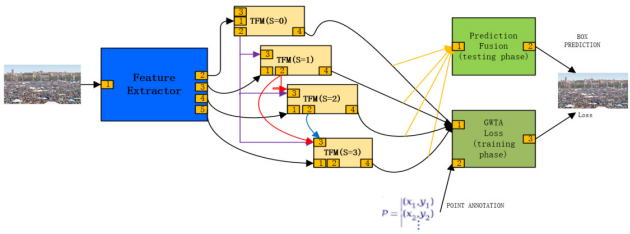

Fig. 4. Depicts the architecture of the proposed LSC-CNN. This model seamlessly incorporates multi-scale data from the feature extractor, enabling predictions at multiple resolutions. These predictions are subsequently merged to create the final detection output. The model is fine-tuned for the per-pixel classification of pseudo ground truth boxes, which are produced in the GWTA training phase, as indicated by the dotted lines.

4 YOLO

4.1 YOLOv1

YOLO (You Only Look Once) is a popular target detection algorithm proposed by Joseph Redmon, Santosh Divvala, Ross Girshick and Ali Farhadi in 2016[20]. YOLO revolutionizes the field of computer vision by providing real-time object detection capabilities.

YOLO works by dividing the input image into grids and simultaneously predicting bounding boxes and class probabilities for each grid cell. This method is different from the traditional target detection algorithm Faster-Cnn[21], which usually uses a sliding window or region proposal mechanism. It has the following key features:

1. Single pass: YOLO performs object detection through a single pass of the neural network, making it much faster than other methods.

2. Unified framework: It predicts bounding boxes and class probabilities directly from the complete image in one go, rather than through a multi-stage process.

3. Grid-based methods: YOLO divides the input image into grids and predicts bounding boxes and class probabilities for each grid cell.

4. Anchor boxes: YOLO uses anchor boxes to improve the accuracy of bounding box predictions.

5. Loss function: YOLOv1 uses a specific loss function that takes into account both positioning error and classification error.

YOLOv1 has the advantages of high real-time performance, global information processing, simplicity and intuitiveness, and end-to-end training. It can process large-size images at a faster speed, make full use of the global information of the image, and the network model is relatively simple and simplifies the training process.

4.2 YOLOv2

YOLOv2 (You Only Look Once version 2) is the second version of the YOLO series, which was improved on the basis of YOLOv1 by Joseph Redmon and Ali Farhadi [22]. YOLOv2 has achieved significant improvements in accuracy and speed, and introduced some new technologies and ideas. The following are the main features of YOLOv2:

1.Darknet-19 network structure: YOLOv2 uses a convolutional neural network named Darknet-19 as the basic network structure. Darknet-19 is a lighter weight network with 19 convolutional layers and 5 pooling layers.

2.Batch Normalization: Batch normalization technology was introduced in Darknet-19, which helps speed up the training process and improve the accuracy of the model.

3. Multi-scale training: YOLOv2 introduces multi-scale training technology, allowing the model to be trained on images of different scales, thereby improving the detection ability of objects of different sizes.

4.Anchor Boxes: YOLOv2 uses predefined anchor boxes, and the size and proportion of these anchor boxes are automatically determined through clustering technolo-

gy. The introduction of anchor boxes improves the accuracy and stability of bounding box prediction.

5. Improvements in the Darknet-19 network: Compared with YOLOv1, the Darknet-19 network structure of YOLOv2 has made some improvements, such as using a 3x3 convolution kernel instead of the 1x1 convolution kernel in YOLOv1.

6. Finer-grained feature maps: YOLOv2 divides the input image into a finer-grained grid, allowing the model to better understand the location and size of objects.

7. Prediction adjustment: YOLOv2 introduces a prediction adjustment mechanism to improve the prediction accuracy of bounding boxes, especially for the detection of small objects.

8. Full image detection: YOLOv2 adopts full image detection, using the entire image as input for target detection, thus reducing repeated calculations and redundant information.

Compared with YOLOv1, it is more accurate and has improved real-time performance. At the same time, Anchor Boxes technology is introduced to improve the prediction accuracy of bounding boxes. However, the complexity has increased, and the detection of some small objects is still poor.

4.3 YOLOv3

YOLOv3 is the third version of the YOLO series, released in 2018 by Joseph Redmon and Ali Farhadi [23]. YOLOv3 further improves on the basis of YOLOv2, improves the accuracy and speed of target detection, and introduces some new technologies. The following are the main features of YOLOv3:

1.Darknet-53 network structure: YOLOv3 uses a deeper network structure called Darknet-53 as the basis. Darknet-53 is a deep residual network with 53 convolutional layers. It is deeper than the Darknet-19 network of YOLOv2 and can extract richer features.

2. Multi-scale prediction: YOLOv3 introduces multi-scale prediction technology, which improves the model's detection ability of objects of different sizes by detecting targets at different scales. YOLOv3 simultaneously predicts bounding boxes of three different scales and predicts them on different feature maps.

3. Anchor box adjustment: YOLOv3 uses more and more refined anchor boxes (Anchor Boxes). The size and proportion of these anchor boxes are more suitable for objects of different scales, improving the prediction accuracy of bounding boxes.

4. Higher resolution: YOLOv3 uses a higher input resolution to improve detection accuracy while maintaining real-time performance. The default input resolution of YOLOv3 is 416x416 pixels, which is higher than YOLOv2.

5. Multi-scale training: Similar to YOLOv2, YOLOv3 also uses multi-scale training technology, allowing the model to better adapt to objects of different sizes and proportions.

6. Route connection and skip connection: The Darknet-53 network introduces structures such as route connection and skip connection, which help to improve the expression ability of the feature map and thereby improve the detection accuracy.

7. Use a convolutional layer with a convolution kernel size of 1x1 for prediction: YOLOv3 uses a 1x1 convolutional layer in the output layer for prediction. These convolutional layers can capture information at different scales, thereby improving the accuracy of detection.

YOLOv3 introduces more and more refined anchor boxes to the previous version, improving the prediction accuracy of boundaries. But the demand for training and parameter adjustment is high.

4.4 YOLOv4

YOLOv4 is an efficient target detection algorithm developed in 2020 by Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao[24]. It is the fourth version of the YOLO target detection series and is designed to improve detection accuracy and speed. The following are the main features and introduction of YOLOv4:

1. Excellent performance: It further improves the accuracy and robustness of detection by introducing a series of new technologies and strategies, such as CSPDarknet53, Bag of Freebies and Bag of Specials[25].

2. High-speed inference speed: YOLOv4 also focuses on the inference speed of the model. Faster inference speed is achieved by optimizing the network structure, using lightweight model components, and introducing new technologies and strategies, such as category coupling and hierarchical prediction.

3. Comprehensive functions: YOLOv4 supports a variety of functions and application scenarios, can be used to detect objects of various sizes and categories, and is suitable for many different fields.

4. Flexibility and ease of use: YOLOv4 provides open source implementation code and pre-trained models, allowing users to easily use and deploy models. In addition, the network structure of YOLOv4 is relatively simple and easy to understand and implement.

YOLOv4 has improved the detection performance and generalization ability of the previous version, and introduced multi-scale prediction, data enhancement and model fusion [26].

4.5 YOLOv5

YOLOv5 was released by Glenn Jocher in 2020 a few months after YOLOv4, developed by the Ultralytics team and released in 2020 [27]. Compared with previous versions, YOLOv5 introduces some new technologies and strategies to further improve the performance and efficiency of target detection. Here are some features and improvements of YOLOv5:

1. Lightweight model: YOLOv5 adopts a lightweight model structure, with fewer parameters and calculations, while maintaining high detection accuracy.

2. Single network structure: YOLOv5 adopts a single network structure, including a backbone network and a series of detection heads. This simplified structure makes the model easier to understand and implement, and facilitates adjustment and optimization [28].

3. AutoML technology: YOLOv5 uses AutoML (Automated Machine Learning) technology, which can automatically adjust the model structure and hyperparameters, thereby achieving better performance and efficiency. This makes YOLOv5 more versatile and adaptable[29].

4. Data enhancement: YOLOv5 introduces a series of new data enhancement technologies, such as CutMix, MixUp and GridMask, etc., to increase the diversity of training data and improve the generalization ability and robustness of the model .

5. Inference speed: While maintaining high accuracy, YOLOv5 also focuses on the inference speed of the model. By optimizing the network structure and using lightweight models, faster inference speed is achieved, which is suitable for application scenarios such as real-time target detection.

6. Open source code: YOLOv5 provides open source implementation code and pretrained models, making it easier for users to use and deploy models. In addition, YOLOv5's code structure is clear and easy to extend and customize.

YOLOv5 optimizes YOLOv4, with lightweight models, CSPDarknet53 backbone, rich data enhancement, small model parameters, TTA test enhancement, and real-time performance.

4.6 YOLOv6

YOLOv6, unveiled in an ArXiv paper in September 2022 by the Visual Artificial Intelligence Department at Meituan[30], follows in the footsteps of YOLOv4 and YOLOv5 by offering a range of model sizes tailored for industrial use[31]. Diverging from the anchor-based approaches of its predecessors, YOLOv6 embraces anchor-free detection methods[32]. Key innovations introduced in this version include[33]:

1.The implementation of a new backbone, named EfficientRep, which is built upon the RepVGG architecture, enhancing parallel processing capabilities.

2.For larger models, the inclusion of a PAN-enhanced neck to improve feature integration.

3.Drawing inspiration from YOLOX, the development of an efficient, decoupled head for more precise object detection.

4.The application of the task arrangement learning method from TOOD for more effective label assignment.

5.Introduction of novel loss functions for classification and regression, including the VariFocal loss for classification and the SIoU/GIoU loss for regression tasks.

6.The integration of a self-distillation strategy to refine both regression and classification outcomes.

7.Faster detector performance through the use of RepOptimizer and a detection quantization scheme that leverages channel-wise distillation.

4.7 YOLOv7

YOLOv7, introduced in an ArXiv paper[34] in July 2022 by the author behind YOLOv4 and YOLOR, sets new benchmarks for object detection in terms of both speed, ranging from 5 FPS to 160 FPS, and accuracy. Similar to its predecessor

YOLOv4, it trains exclusively on the MS COCO dataset, opting out of using a pre-trained backbone[35]. YOLOv7 brings forth novel architectural modifications and introduces several optimization packages aimed at enhancing accuracy without compromising inference speed—though they do extend the training duration.

The architectural innovations in YOLOv7 include:

1. The introduction of the Extended Efficient Layer Aggregation Network (E-ELAN), which enhances network learning by efficiently combining and reorganizing different groups of features. This process shuffles and merges features to improve learning efficacy without interrupting the original gradient path.

2. A fresh model scaling approach utilizing tandem models. This strategy uniformly scales the depth and width of the network blocks by the same factor, ensuring the model's structure remains optimal for performance.

4.8 YOLOv8

YOLOv8 is a new SOTA model that contains P5 640 and P6 1280 resolution target detection networks and a YOLACT-based instance segmentation model[36]. Similar to YOLOv5, in order to meet the needs of various scenarios, models of different sizes are provided. Its backbone network and Neck part refer to the design concept of YOLOv7 ELAN, adopt a richer C2f structure, and make different adjustments for different scale models. The number of channels is adjustable, and so is the performance. can be significantly improved. The head part has significantly changed from anchor-based to anchor-free by adopting a decoupled head structure. For loss calculation, see TaskAlignedAssigner adopts a positive sample dispersion strategy and introduces Distribution Focal Loss. Regarding training data augmentation, an operation was introduced to turn off Mosaic augmentation for the last 10 epochs of YOLOX to improve accuracy.

5 Experiment

5.1 Pascal VOC Dataset

Originally created by the VGG organization at the University of Oxford, UK, the Pascal VOC (Visual Object Classes) dataset [37] is one of the widely respected classic datasets in the field of computer vision and is used for various tasks, including object detection, image classification and Semantic segmentation, etc. Covers a variety of common object categories in daily life, such as people, vehicles, animals, and furniture. The data set is divided into a training set and a test set, and each category has detailed annotation information, including object bounding boxes and corresponding category labels. In addition to object detection, the Pascal VOC dataset also contains image data for semantic segmentation tasks and corresponding annotation information for specifying the semantic category of each pixel in the image. This data set provides a standard evaluation protocol for measuring the performance of the model on detection and segmentation tasks. Evaluation indicators include accuracy, recall, precision, F1 score, mAP, etc. The release of the Pascal VOC dataset promotes

research and progress in the field of computer vision and provides a standardized platform for algorithm comparison and performance evaluation.

5.2 COCO Dataset

The COCO (Common Objects in Context) data set is a large-scale image data set[38], mainly used for computer vision tasks such as target detection, semantic segmentation, instance segmentation and key point detection. Created by Microsoft Research, the dataset contains more than one million images covering 80 different categories of objects. The images are derived from a variety of real-life scenes, covering a rich variety of everyday objects and situations. Each image is equipped with detailed annotation information, including object bounding boxes, class labels, and masks for instance segmentation. The COCO dataset is widely used in the field of computer vision and has become one of the important benchmarks for evaluating algorithm performance. Its rich images and diverse scenes provide rich data resources for algorithm training and testing. Through the COCO dataset, researchers can develop and evaluate various object detection and segmentation algorithms, and promote research and development in the field of computer vision.

5.3 UCF_CC_50 Dataset

The UCF_CC_50 dataset is a dataset commonly used in crowd counting research and was created by a research team at the University of Central Florida (UCF). The dataset contains 50 video sequences, each with a resolution of 240x360 pixels, and a total of over 63,500 frames. These video sequences cover a variety of scenes and environments, including indoors, outdoors, traffic intersections, and train stations. Each sequence is annotated and includes people density maps that can be used to train and evaluate the performance of crowd counting algorithms. Due to its wide application, the UCF_CC_50 dataset has become an important benchmark dataset in the field of crowd counting. Researchers can use this data set to evaluate the performance of their proposed crowd counting algorithms in different scenarios and conduct comparisons and analyzes between algorithms.

5.4 ShanghaiTech Part Dataset

ShanghaiTech Part is a commonly used dataset for crowd counting research, created by ShanghaiTech University[39]. The data set is divided into two parts, Part A and Part B. Part A contains 482 images for training and validation. Each image is labeled with crowd density and covers different scenes and density levels. The Part A dataset is used to train and debug the crowd counting algorithm. Part B contains 716 high-resolution images for testing. Each image is also labeled with crowd density, covering different scenes and density levels. The Part B dataset is used to evaluate and compare the performance of different algorithms.

5.5 CIFAR-100 Dataset

CIFAR-100[40](Canadian Institute for Advanced Research) is one of the standard datasets for image classification tasks, created in 2009 by Alex Krizhevsky, Vinod Nair and Geoffrey Hinton from the Canadian Institute for Advanced Research. It contains 60,000 32x32 color images from 100 categories, each category contains 600 images. These categories are divided into 20 major categories, each of which contains 5 subcategories. CIFAR-100 images cover a wide range of topics, including animals, plants, vehicles, household items, and more. This dataset is designed to provide challenges to image classification algorithms because the image resolution is relatively low while the image content is complex and category-rich. The CIFAR-100 dataset is commonly used to evaluate the performance of image classification algorithms, especially when computing resources are limited. Due to its relatively small size and diverse categories, researchers can quickly test and compare different classification models. Common image classification algorithms such as CNN (Convolutional Neural Network), ResNet, VGG, etc. are often trained and evaluated on the CIFAR-100 data set to obtain better generalization performance on larger and more complex data sets.

5.6 Experimental Results

Table 1. About the experimental results of various versions of YOLO on the Pascal VOC data set and COCO data set. The metrics reported by YOLO and YOLOv2 are based on the VOC2007 dataset, while the metrics for the remaining architectures are based on the COCO2017 dataset.

Version	*Date*	*Anchor*	*Frame-work*	*Backbone*	*AP(%)*
YOLO[44]	2015	No	Darknet	Darknet24	63.4
YOLOv2[44]	2016	Yes	Darknet	Darknet24	63.4
YOLOv3[44]	2018	Yes	Darknet	Darknet53	36.2
YOLOv4[44]	2020	Yes	Darknet	CSPDarknet53	43.5
YOLOv5[44]	2020	Yes	Pytorch	Modified CSPv7	55.8
YOLOv6[44]	2022	No	Pytorch	EfficientRep	52.5
YOLOv7[44]	2022	No	Pytorch	RepConvN	56.8
YOLOv8[44]	2023	No	Pytorch	YOLOv8	53.9

Table 2. Different models are compared in the ShanghUCF CC 50 data set and the ShanghaiTech Part data set.

	ST Part_A		**ST Part_B**		**UCF_CC_50**	
Models	MAE	MSE	MAE	MSE	MAE	MSE
Zhang et al. [9]	181.8	277.7	32.0	32.0	467.0	498.5
SCNN [41]	90.4	135.0	21.6	33.4	318.1	439.2
CP-CNN [42]	73.6	106.4	20.1	30.1	295.8	320.9
IG-CNN [43]	72.5	118.2	13.6	21.1	291.4	349.4
Crowd CNN[8]	-	-	-	-	467.0	498.5

MCNN[9]	110.2	173.2	26.4	41.3	377.6	509.1
SAN[10]	-	-	16.86	28.41	271.6	391.00
LSC-CNN[11]	66.4	117.0	8.1	12.7	225.6	302.7

6 Conclusions

In crowd counting and human density detection tasks, convolutional neural networks and YOLO algorithms combine optimization and regularization techniques to provide effective methods to solve these challenges. First, the optimization algorithm plays a key role in training CNN and YOLO models, optimizing model parameters by minimizing the loss function to improve the model's performance on crowd counting and density detection tasks. This helps accelerate model convergence, allowing for faster model training and optimization processes. Secondly, regularization technology plays a role in regularizing model parameters in CNN and YOLO, which helps prevent the model from overfitting the training data and improves the generalization ability of the model. By constraining the complexity of the model, regularization technology makes the model more robust and better able to adapt to different crowd density and counting scenarios. In summary, the convolutional neural network and YOLO algorithm based on crowd counting and human density detection combine optimization and regularization techniques to provide powerful tools for solving this type of problem. The application of these technologies makes the model more stable and accurate in complex scenes, providing important support and promotion for research and applications in the fields of crowd counting and density detection.

Acknowledgment

This work was funded by the Guangxi Science and Technology Development Project (AB23026135； AB21220011), the Guilin Science and Technology Plan Project (20210220).

References

1. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), 770-778.
2. Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. In International Conference on Learning Representations (ICLR).
3. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going Deeper with Convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), 1-9.
4. Dollár, P., Tu, Z., Perona, P., & Belongie, S. (2009). Integral Channel Features. In British Machine Vision Conference (BMVC).

5. Girshick, R. (2015). Fast R-CNN. In Proceedings of the IEEE international conference on computer vision (ICCV), 1440-1448.
6. Dalal, N., & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 1, pp. 886-893.
7. Uijlings, J. R. R., van de Sande, K. E. A., Gevers, T., & Smeulders, A. W. M. (2013). Selective Search for Object Recognition. International Journal of Computer Vision, 104(2), 154-171.
8. Zhang, Y., Zhou, D., Chen, S., Gao, S., & Ma, Y. (2015). Cross-scene Crowd Counting via Deep Convolutional Neural Networks. IEEE Transactions on Neural Networks and Learning Systems, 26(7), 1455-1467.
9. Zhang, C., Li, H., Wang, X., & Yang, X. (2016). Single-Image Crowd Counting via Multi-Column Convolutional Neural Network. IEEE Transactions on Circuits and Systems for Video Technology, 26(6), 1025-1036.
10. Liu, X., van de Weijer, J., Bagdanov, A. D., & Zhang, L. (2019). Crowd Counting Using Scale-Aware Attention Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 41(12), 2858-2872.
11. Boominathan, L., Kruthiventi, S. S. S., & Babu, R. V. (2016). Locate, Size and Count: Accurately Resolving People in Dense Crowds via Detection. IEEE Transactions on Circuits and Systems for Video Technology, 26(7), 1255-1268.
12. Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Proceedings of the 32nd International Conference on Machine Learning (ICML-15).
13. Zhang, C., Li, H., Wang, X., & Yang, X. (2016). Single-Image Crowd Counting via Multi-Column Convolutional Neural Network. IEEE Transactions on Circuits and Systems for Video Technology, 26(6), 1025-1036.
14. J. Long, E. Shelhamer, and T. Darrell. Fully convolution- al networks for semantic segmentation. arXiv preprint arX- iv:1411.4038, 2014.
15. M. D. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang, Q. V. Le, P. Nguyen, A. Senior, V. Vanhoucke, and J. Dean. On rectified linear units for speech processing. In ICASSP, pages 3517–3521. IEEE, 2013.
16. D. B. Sam, S. Surya, and R. V. Babu. Switching convolu- tional neural networks for crowd counting. In IEEE Confer- ence on Computer Vision and Pattern Recognition, 2017.
17. B. Sheng, C. Shen, G. Lin, J. Li, W. Yang, and C. Sun. Crowd counting via weighted vlad on dense attribute feature maps. IEEE Transactions on Circuits and Systems for Video Tech- nology, 2016.
18. V. A. Sindagi and V. M. Patel. CNN-based cascaded multi- task learning of high-level prior and density estimation for crowd counting. In IEEE International Conference on Advanced Video and Signal Based Surveillance, 2017.
19. V. A. Sindagi and V. M. Patel. Generating high-quality crowd density maps using contextual pyramid CNNs. In IEEE International Conference on Computer Vision, 2017.
20. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779–788, 2016.
21. R. Girshick, “Faster-cnn,” in Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448, 2015.
22. J. Redmon and A. Farhadi, “Yolo9000: better, faster, stronger,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7263–7271, 2017.

23. J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.
24. A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," arXiv preprint arXiv:2004.10934, 2020.
25. S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1492–1500, 2017.
26. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," The Journal of Machine Learning Research, vol. 15, no. 1, pp. 1929–1958, 2014.
27. G.Jocher,"YOLOv5 by Ultralytics." https://github.com/ultralytics/yolov5, 2020. Accessed: February 30, 2023.
28. Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," arXiv preprint arXiv:2107.08430, 2021.
29. Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," arXiv preprint arXiv:2107.08430, 2021.
30. C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie, et al., "Yolov6: A single-stage object detection framework for industrial applications," arXiv preprint arXiv:2209.02976, 2022.
31. C. Feng, Y. Zhong, Y. Gao, M. R. Scott, and W. Huang, "Tood: Task-aligned one-stage object detection," in 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 3490–3499, IEEE Computer Society, 2021.
32. X. Ding, H. Chen, X. Zhang, K. Huang, J. Han, and G. Ding, "Re-parameterizing your optimizers rather than architectures," arXiv preprint arXiv:2205.15242, 2022.
33. C. Shu, Y. Liu, J. Gao, Z. Yan, and C. Shen, "Channel-wise knowledge distillation for dense prediction," in Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 5311–5320, 2021.
34. C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," arXiv preprint arXiv:2207.02696, 2022.
35. C.-Y. Wang, H.-Y. M. Liao, and I.-H. Yeh, "Designing network design strategies through gradient path analysis," arXiv preprint arXiv:2211.04800, 2022.
36. G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics." https://github.com/ultralytics/ultralytics, 2023. Accessed: February 30, 2023.
37. Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. (2010). The Pascal Visual Object Classes (VOC) Challenge. International Journal of Computer Vision, 88(2), 303-338.
38. Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In European conference on computer vision (pp. 740-755). Springer, Cham.
39. Zhang, Y., Zhou, D., Chen, S., Gao, S., & Ma, Y. (2016). Single-image crowd counting via multi-column convolutional neural network. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 589-597).
40. Krizhevsky, A., & Hinton, G. E. (2009). Learning multiple layers of features from tiny images. Technical report, University of Toronto.
41. D. Babu Sam, S. Surya, and R. V. Babu, "Switching convolutional neural network for crowd counting," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

42. V. A. Sindagi and V. M. Patel, “Generating high-quality crowd density maps using contextual pyramid CNNs,” in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017.
43. D. Babu Sam, N. N. Sajjan, R. V. Babu, and M. Srinivasan, “Divide and grow: Capturing huge diversity in crowd images with incrementally growing CNN,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
44. Terven J, Cordova-Esparza D. A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond. arXiv 2023[J]. arXiv preprint arXiv:2304.00501.