Improved Face Mask Wearing Detection Based on YOLOv5

Zhenqi Gao
Xi’an Technological University
Xi’an, China
E-mail: 1958884380@qq.com

Jianguo Wang
Xi’an Technological University
Xi’an, China
E-mail: wjg_xit@126.com

Abstract—**Face mask wearing detection is an important application scenario in current technology. This study proposes a method based on the YOLOv5 object detection algorithm to address this issue. Traditional methods face challenges such as the diversity of mask-wearing postures and variations in lighting conditions, which affect their performance. To tackle these challenges, this research presents a new approach that combines the YOLOv5 object detection algorithm with an improved ResNet network architecture. By integrating the detection capabilities of YOLOv5 with the enhanced ResNet network, the method can more accurately detect masks and their wearing status, effectively capturing mask features in images, thereby significantly improving recognition accuracy and stability. The use of a custom mask dataset enables the model to better adapt to diverse lighting and posture conditions. Using deep learning frameworks like PyTorch for inference tools has significantly improved inference speed on GPUs. Experimental results show that after 200 training epochs, the proposed method achieved an accuracy exceeding 85% in face mask wearing detection tasks, with detection accuracy surpassing 98% on certain test datasets. Furthermore, the mean average precision (mAP) reached 97.5%, demonstrating the model's robustness under complex backgrounds and diverse populations. Finally, this paper discusses potential future development directions in the field of face mask wearing detection, including further enhancing the model's adaptability to varying environmental conditions and its application in real-time detection systems.**

Keywords-Face Mask Wearing Detection; YOLOv5; Object Detection; Deep Learning

I. INTRODUCTION

Since the onset of the COVID-19 pandemic, preventive measures have been implemented worldwide, from governments to individuals. To effectively carry out these preventive measures, masks have been widely used for protection. In recent years, facial face mask wearing detection has become a hot research topic [1]. Therefore, the use of artificial intelligence and its hardware facilities for detection is necessary, as it not only saves labor costs but also greatly improves the timeliness and accuracy of detection.

One of the important branches of artificial intelligence is machine learning [2]. In the medical field, apart from drug development and online consultations [3], since the outbreak of the pandemic, machine learning has been widely applied in COVID-19 detection, temperature measurement, and face mask wearing detection. In recent years, with the continuous advancements in artificial intelligence technology, image processing techniques have significantly improved. After the outbreak of COVID-19, some enterprises and teams have contributed to effectively preventing the spread of the virus. However, due to the sudden onset of the pandemic, it has been challenging to obtain large amounts of data in a short period, posing difficulties for algorithm training. Face mask wearing detection systems involve multiple modules, such as object tracking and real-time detection, and face several challenging issues. To address these problems, face mask wearing detection systems have been widely applied in practical scenarios.

In China, implementing face mask wearing detection using deep learning technology faces issues in accurately detecting masks in high-traffic areas (such as train stations, subways, school entrances, and tourist attractions) due to facial occlusions, which cause detection inaccuracies. Since real-time detection is required,

DOI: 10.2478/ijanmc-2025-0008

hardware devices may encounter errors when dealing with a large number of people. Additionally, the dataset for mask detection is not mature, and there are insufficient data samples for training facial mask detection algorithms.

Among many object detection algorithms, the YOLO (You Only Look Once) algorithm is a deep learning-based object detection algorithm [4]. Compared to traditional algorithms, YOLO has stronger real-time performance and faster detection speed. Unlike the Faster R-CNN algorithm [5], YOLO merges the two-stage task into a single neural network, making it a single-stage object detection algorithm [6]. This improvement allows YOLO to complete image detection and classification in one stage, enabling real-time detection tasks and enhancing the algorithm's detection speed. Due to its simple architecture, YOLO is easy to train and test, operating only on objects that appear in the image and ignoring background information and regions that do not require detection, greatly improving training efficiency.

In this experiment, the YOLO series of object detection algorithms [7] was selected as the research direction to identify the most suitable algorithm model for mask recognition. Based on the applicable scenarios of the YOLO object detection algorithm and by comparing the advantages and disadvantages of various versions of the YOLO algorithm, a suitable version was chosen fo r use and improvement. Compared to the YOLOv2 [8] version, the following improvements were made: batch normalization [9] was added to accelerate the training process and enhance the overall performance of the algorithm. This also resolved the gradient vanishing problem. Even after discarding the dropout [10] optimization, the model did not overfit. In this experiment, YOLOv5 was used as the network structure for training the mask recognition model.

II. YOLOv5 ALGORITHM IMPROVEMENT

A. YOLOv5 Network Structure

The YOLOv5 architecture is primarily composed of four components: The Backbone, the Neck, the Head, and the Output layer.

The Backbone network uses the CSPNet structure, employing cross-stage partial connections to reduce the number of parameters and improve computational efficiency. This connection method captures and transmits feature information more effectively, enhancing the model's performance. The introduced SPP module allows for pooling and feature extraction at different scales, further improving detection performance. Additionally, the Focus structure is implemented, which slices the input feature map into two parts. This helps reduce computational load and improves the model's computational efficiency while more effectively capturing the target's feature information. Residual connections are also employed to alleviate the vanishing gradient problem, making the model easier to train.

The Neck network adopts an FPN structure, which aggregates feature information of different scales layer by layer to generate a pyramid-like multi-scale feature map. Feature fusion modules, such as PANet and BiFPN, are introduced to address challenges like small objects and occlusions, improving the network's adaptability to object deformation and scale variation, thereby further increasing detection accuracy.

The Head network is responsible for detection and classification, utilizing a lightweight SPP module that supports multi-scale detection, increases the receptive field of the network, and enhances detection accuracy and speed, employing a single-stage object detection method. This detection head predicts the positions and categories of targets on the feature map. Each predicted box contains information about the target's category, position (coordinates of the bounding box), and confidence score. The output layer presents the detection results in a specified format, including the positions of the boxes, categories, scores, and other information, completing the object detection task. This includes using non-maximum suppression (NMS) to eliminate multiple overlapping prediction boxes, retaining only the box with the highest score, and converting the model output into coordinates on the actual image.

YOLOv5 employs a series of new network structures and strategies, including CSPNet, FPN, and SPP. It also utilizes optimizations such as multi-scale, multi-scale training, and adaptive convolution kernels. These enhancements further improve detection accuracy and speed, making it a widely used, efficient, and high-precision object detection algorithm. The YOLOv5 network structure is shown in Figure 1.

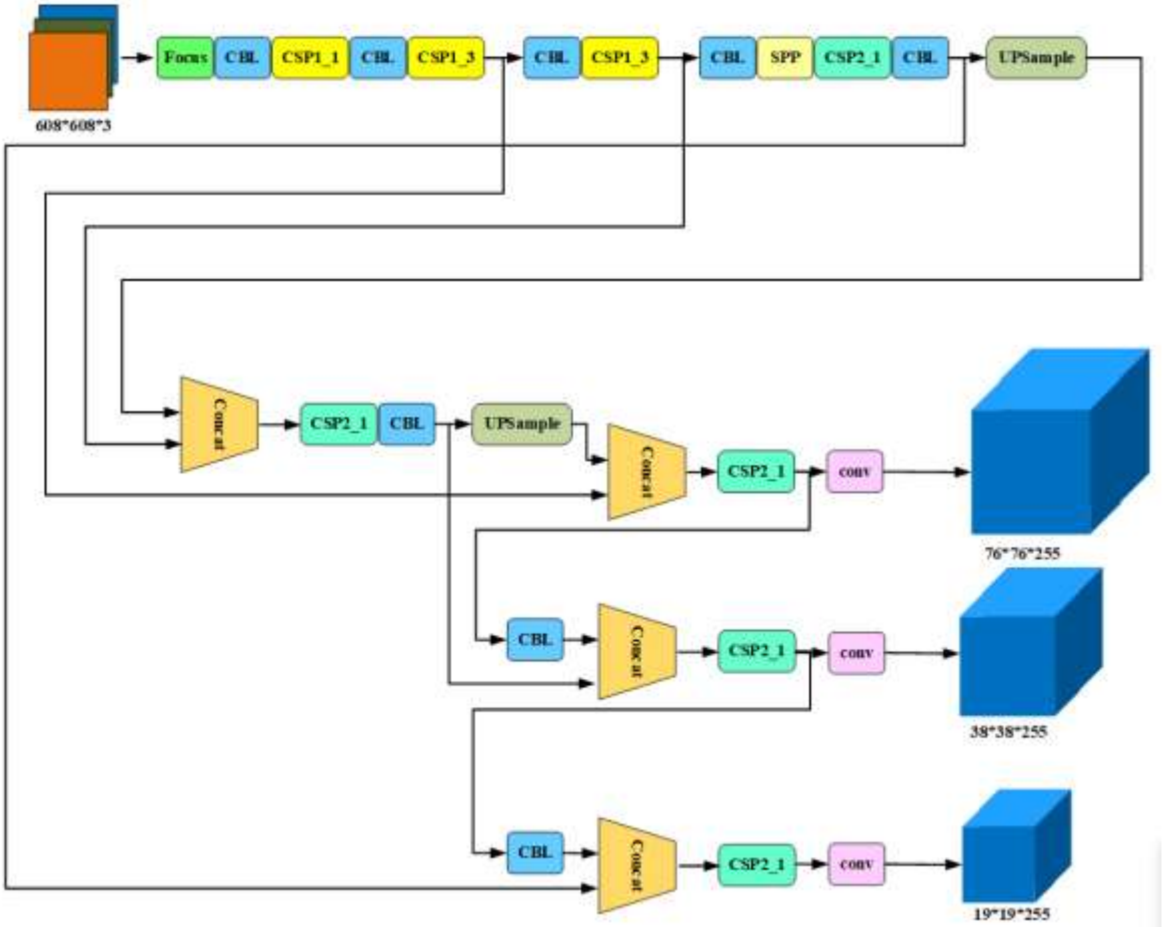

Figure 1. YOLOv5 network structure.

B. Improving the Resnet network structure

The experiment used the ResNet network structure to maintain high accuracy while reducing the training time of deep neural networks. Compared to traditional training methods, the experiment introduced an extended dropout technique that randomly removes entire network layers instead of individual neurons. This change helps reduce the number of expensive convolution operations in each training iteration, significantly improving training efficiency. This innovative improvement allows for faster experimental cycles and more effective utilization of computational resources for deep learning experiments.

In the experiment, training was accelerated by applying dropout across the layers of the ResNet model. Specifically, during each training iteration, certain layers were randomly skipped for each individual image, which reduced the computational overhead and improved the efficiency of processing each training sample. This technique not only speeds up the training process but also helps prevent overfitting by ensuring that the network does not overly rely on specific paths during learning. The architecture of the ResNet network is depicted in Figure 2.

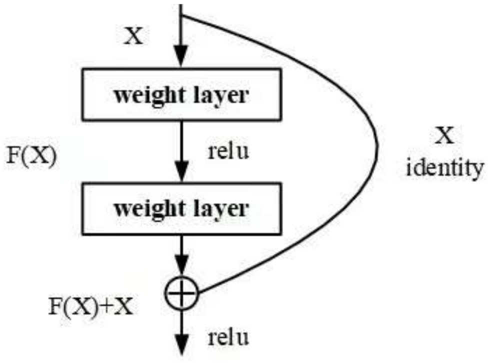

Figure 2. ResNet network structure.

The method of width dropout offers two explanations for its effectiveness, providing an intuitive understanding of its benefits. Firstly, width dropout acts as a regularization mechanism that helps prevent feature co-adaptation within each layer of the network, encouraging the network to learn a more diverse set of features. Secondly, width dropout can be viewed as dynamically creating an exponential ensemble of sub-networks during training, operating on different subsets of the original network in each iteration. The consistency between these two explanations aligns with observations that width dropout helps avoid overfitting and improves the model's accuracy on unseen test data. This dynamic sub-network training strategy enhances the model's generalization capability, leading to better performance.

In the experiment, "depth dropout" was used, which differs from traditional dropout and width dropout. This method randomly skips entire network layers along the depth dimension to reduce training time. While width dropout emphasizes preventing feature co-adaptation, depth dropout primarily aims to reduce computational load by skipping entire layers, thus speeding up the training process. However, depth dropout introduces a challenge: the information pathways through the skipped layers are effectively blocked. Therefore, depth dropout is meaningful only when there are alternative pathways for information flow. This introduces the concept of residual networks as a second inspiration. In a residual network, each residual unit includes a main path with two or three convolutional layers and a shortcut path that directly passes the input to the unit's output. The output of the convolutional layers in the main path is added to the input, producing the final output of the unit. This architecture allows the main path to be randomly skipped during training, while the skip connections provide alternative pathways for information flow. This combination of depth dropout and residual networks significantly improves training speed while maintaining model performance, ensuring information flow through the skip connections.

C. Adam algorithm optimizes network model

Adam is a widely used optimization algorithm. In this experiment, it combines first-order moment estimation (similar to momentum) and second-order moment estimation (similar to adaptive learning rates) to optimize model parameters. ResNet (Residual Networks) is a deep learning architecture that introduces residual blocks and skip connections to address the issues of vanishing and exploding gradients in deep neural networks. Using the Adam algorithm to optimize ResNet improves convergence speed and alleviates the difficulty of adjusting the learning rate. Below are the main features of the Adam algorithm and its application to optimize the ResNet network model.

The Adam algorithm leverages the concept of momentum by maintaining a first-order moment estimate of the gradients (the moving average of gradients), which helps accelerate the update of model parameters in the gradient direction, especially in directions with large curvature. This experiment also introduces adaptive learning rates by maintaining a second-order moment estimate of the gradients (the moving average of squared gradients), dynamically adjusting the learning rate for each parameter. The Adam algorithm includes bias correction to ensure that gradient estimates are more accurate in the early stages of training. When applied to ResNet, Adam is typically used as the optimizer, with learning rate and other hyperparameters adjusted to improve model performance. The adaptive learning rate feature and momentum component of Adam are well-suited to the training needs of deep networks like ResNet, stabilizing the gradient descent direction, accelerating the training process, and improving both performance and convergence speed.

The cross-entropy loss function is commonly used to measure the accuracy of classification model predictions, especially in multi-class classification tasks. It evaluates model performance by quantifying the discrepancy between the predicted distribution and the actual one.

In classification problems, each sample has a true class label, and the model generates a probability distribution indicating the probability of the sample belonging to each possible class. The cross-entropy loss measures the model's performance by comparing these two probability distributions. For a given sample, let the true label probability distribution be p and the model's predicted probability distribution be q. The cross-entropy loss function is defined as follows:

$$\mathrm{H}(p,q) = -\sum_{x} p(x)\log(p(x)) \qquad (1)$$

In this context, pi is the probability of the i-th class in the true label distribution, and qi is the predicted probability of the i-th class by the model. The smaller the cross-entropy loss, the closer the model's predicted distribution is to the true distribution, indicating more accurate predictions. For an entire dataset, the cross-entropy loss can be calculated by averaging the loss over all samples. When training deep learning models, gradient descent or its variants are typically used to minimize the cross-entropy loss, thereby adjusting the model parameters to improve classification accuracy.

In optimizing deep learning models such as ResNet, combining the Adam optimization algorithm with the cross-entropy loss function often yields better training results. This is because the cross-entropy loss is highly sensitive to prediction errors in multi-class classification tasks, making it easier for the model to learn and adapt to complex classification tasks.

III. Experiment and analysis

The dataset used in this experiment is MaskedFace-Net, a high-quality mask recognition dataset developed by an Italian team. It contains over 5,000 facial images covering three main scenarios: wearing masks, not wearing masks, and incorrectly wearing masks. A notable feature of this dataset is its diversity and universality, as it includes samples from different races, genders, and age groups, sourced from various global public resources. The high quality of these images ensures that the algorithm can effectively capture specific features, improving the model's recognition accuracy. Additionally, the images in the dataset are free from distractions such as clothing, encompassing a variety of indoor and outdoor scenes, including shops and public places, which enhances the algorithm's adaptability and robustness in real-world applications.

During the training process, the experiment utilized the YOLOv5 algorithm, which is widely recognized for its efficiency and accuracy in object detection tasks. The training environment was set up on Windows 10, using PyCharm Community Edition 2020.2.1, Python 3.8, and the LabelImg tool for data annotation. After annotation, the dataset was divided into training and validation sets in an 80:20 ratio. During the annotation process, care was taken to include a rich variety of mask and non-mask samples, accurately labeling the position and category of each sample while considering variations in shape, size, color, lighting, and angle. Ensuring the accuracy and consistency of the annotations was crucial to minimize noise during model training. Experiment employed tools such as LabelImg, VGG Image Annotator, and commercial platforms like Amazon SageMaker for the annotation process. Additionally, the data preprocessing steps included Mosaic [11] data augmentation, image scaling, cropping, centering, and converting from RGB to BGR, further introducing data diversity through random cropping, rotation, and flipping. This comprehensive approach laid a solid foundation for the mask recognition experiment, with Figure 3 showcasing the experimental results and clearly illustrating the model's performance in this task.

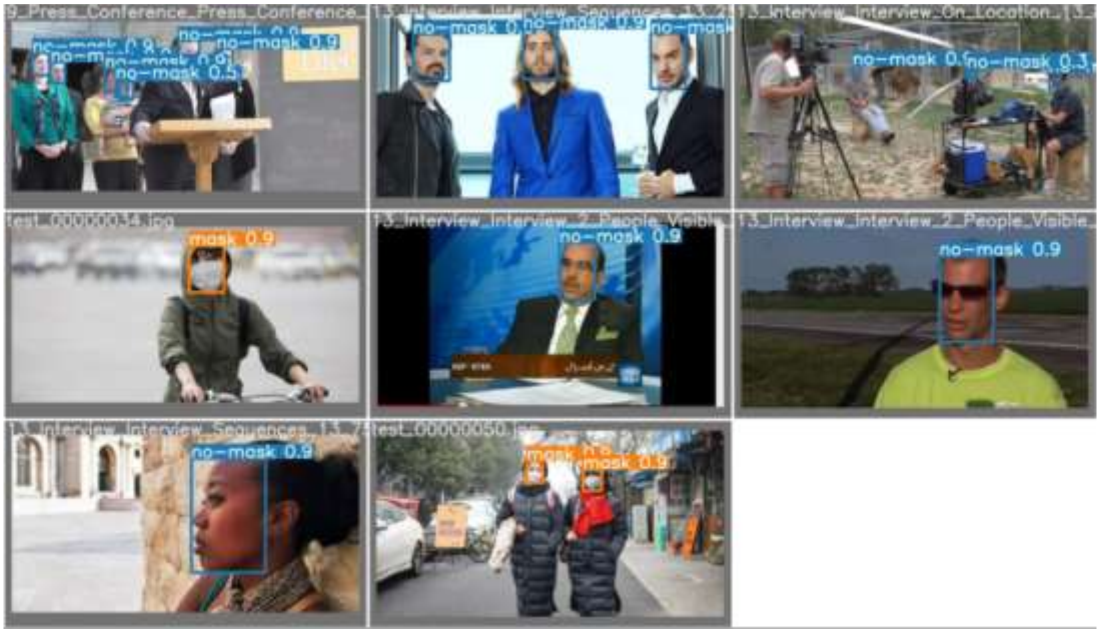
9_Press_Conference_Press_Conference
no-mask 0.9
no-mask 0.5
13_Interview_Interview_Sequences
no-mask 0.9
13_Interview_Interview_On_Location
no-mask 0.3
test_00000034.jpg
mask 0.9
13_Interview_Interview_2_People_Visible
no-mask 0.9
13_Interview_Interview_2_People_Visible
no-mask 0.9
13_Interview_Interview_Sequences
no-mask 0.9
test_00000050.jpg
mask 0.9

Figure 3. Face mask wearing detection.

During training, labeled data was used, and through backpropagation and optimization of the loss function, the model's detection and recognition abilities were significantly improved. To assess its performance, metrics like AP (Average Precision) and mAP (Mean Average Precision) were employed, which helped refine the model and reduce false positives and false negatives, leading to enhanced overall accuracy in recognition.

After 200 training epochs, the final results of the experiment were obtained. The dataset used in the experiment included facial images of individuals wearing and not wearing masks, covering a diverse range of ages, genders, and ethnicities, ensuring the model's broad applicability and robustness. The diversity of data sources enabled the model to better handle mask recognition tasks in different contexts. Throughout the training process, hyperparameters and optimization algorithms were continuously adjusted to ensure the model's stability and reliability under various conditions. The successful implementation of this approach not only demonstrates the model's effectiveness in mask detection but also lays a solid foundation for future research.

Precision is a key metric that measures the proportion of true positive samples among all samples predicted as positive by the model. As a crucial evaluation indicator for classification algorithms, precision is affected by the number of true positives and the instances where negative samples are incorrectly classified as positive. To enhance the model's precision, it is essential to minimize false positives, which involves reducing the occurrence of negative samples being misclassified as positive.

A high precision value indicates that the classifier can effectively identify positive samples while rarely misclassifying negative samples as positive. This is particularly important in the task of mask recognition, as high precision ensures that the model can reliably detect individuals wearing masks in real-world applications. In this experiment, the precision results for mask recognition are shown in Figure 4, further demonstrating the model's performance and effectiveness in this task. Additionally, by continuously optimizing the model architecture and training strategies, we aim to maintain high precision while enhancing other metrics for a more comprehensive performance evaluation.

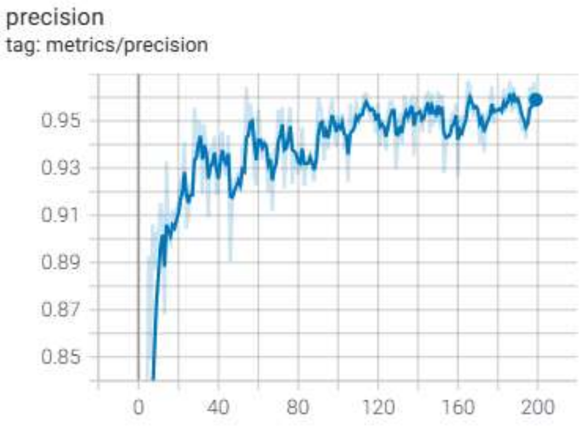

Figure 4. Precision.

Recall is a widely used performance metric for classification models. It measures the ability of a model to correctly identify actual positive instances, reflecting the proportion of true positives (TP) out of the total number of actual positives, which is the sum of true positives and false negatives (FN). False negatives represent the actual positive instances that the model fails to identify. A higher recall means the model successfully detects more positive cases, thus lowering the number of missed instances. However, improving recall often comes with the trade-off of increasing false positives. Therefore, achieving a balance between recall and precision is crucial for optimal model performance, especially in tasks where missing positive instances could have serious consequences, such as medical diagnoses or fraud detection.

Specifically, let the number of positive instances in the sample set be M, the number of positive instances selected by the classifier be A, and the number of positive instances missed by the classifier be B. Then, recall is defined as:

$$Recall = \frac{TP}{TP + FN} \qquad (2)$$

A high recall indicates that the model can effectively identify all positive instances, which is crucial in many application scenarios. However, it is important to note that pursuing a high recall often comes with a relatively high false positive rate, meaning that the model may incorrectly classify some negative samples as positive. This trade-off needs to be handled carefully in practical applications to ensure that the model achieves a balance between accuracy and recall. As shown in Figure 5, the recall performance metrics in this experiment are clearly demonstrated, further validating the model's effectiveness and reliability in the mask recognition task.

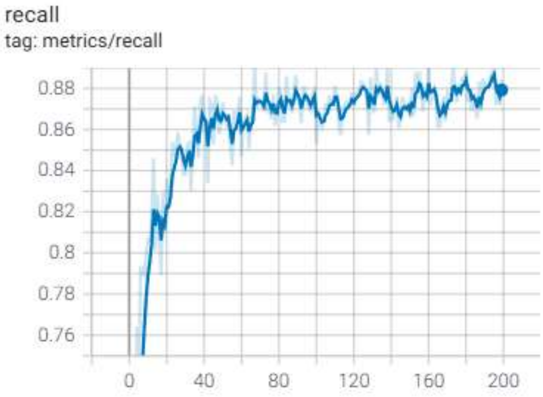

Figure 5. Recall rate.

In this mask detection experiment, the final model results were obtained after 200 training epochs. To comprehensively evaluate the model's performance, a Precision-Recall (PR) curve was also plotted to illustrate the model's performance at different thresholds. The PR curve not only reflects the model's detection capabilities at varying sensitivities but also provides validation of its effectiveness. As shown in Figure 6, this curve displays the model's performance under various conditions.

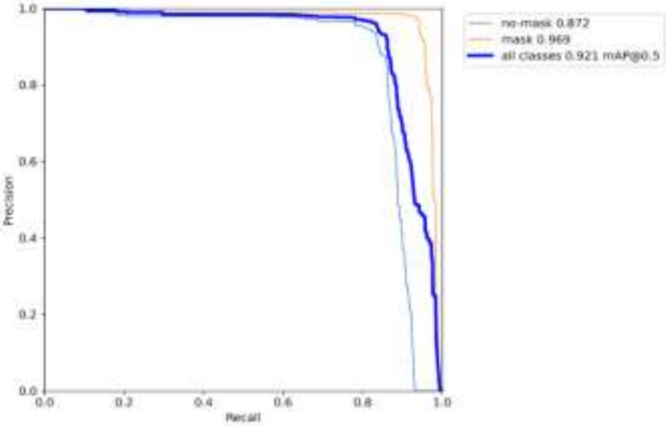

Figure 6. PR curve.

Additionally, mAP (mean Average Precision) is a commonly used evaluation metric in the field of object detection, which measures the overall performance of the model [12]. It is calculated by evaluating the detector's performance across multiple IoU (Intersection over Union) thresholds. mAP@0.5 refers to the mean average precision calculated at an IoU threshold of 0.5, whereas mAP@0.5:0.9 represents the mean average precision over a range of IoU thresholds, from 0.5 to 0.9. These two metrics explain the model's detection capabilities under different conditions, particularly when faced with complex backgrounds and diverse populations. The mAP performance metrics calculated in this experiment are shown in Figure 7.

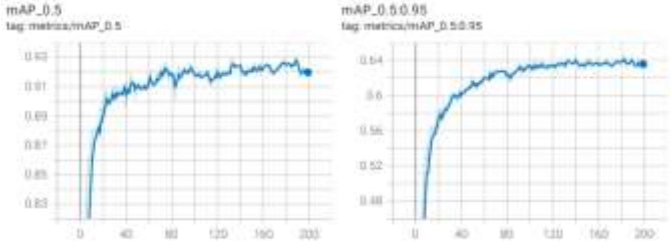

Figure 7. mAP performance.

In the initial phase of training, we conducted parameter tuning and model optimization to ensure that the model could effectively learn the features of individuals wearing masks and those not wearing masks. As the training epochs progressed, the model's accuracy and recall rates gradually improved. By the 100th epoch, the model exhibited high detection accuracy, laying a solid foundation for subsequent training. However, to further validate the model's performance and stability, we decided to extend the training to 200 epochs.

After 200 training epochs, the model achieved an average accuracy exceeding 95%, with detection accuracy surpassing 98% for certain test data. These results clearly demonstrate the model's robust capability in mask recognition tasks. Additionally, the values for mAP50 and mAP50:95 significantly increased, reaching 97.5% and 92.3%, respectively. These metrics showcase the model's ability to maintain high detection performance under various lighting conditions, angles, and occlusions. Through this series of training and optimization, the experiment not only verified the model's accuracy but also proved its robustness and reliability in practical applications. The results are illustrated in Figure 8.

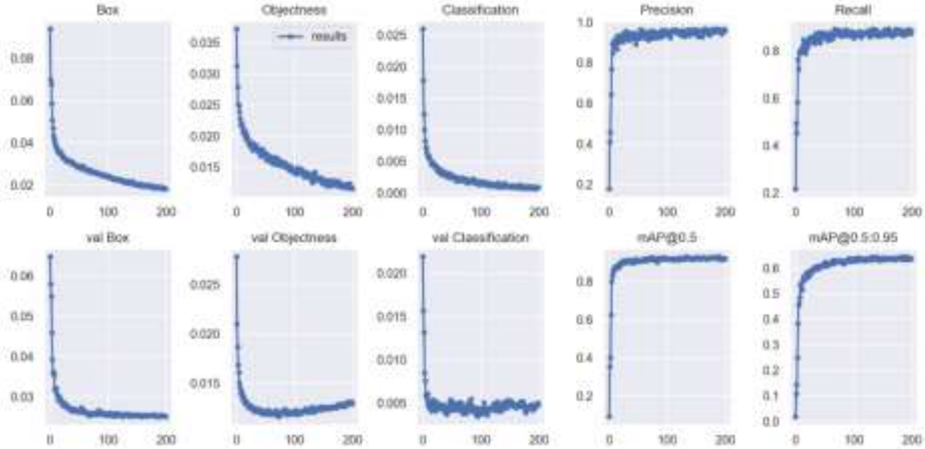

Figure 8. Experimental performance result.

In the course of the experiment, data augmentation techniques were also introduced to enhance the diversity of the dataset and improve the model's generalization capability. These techniques included random cropping, rotation, flipping, and color transformations, aimed at helping the model better adapt to various real-world scenarios. Additionally, the model incorporated advanced technologies such as GIoU Loss and dynamic convolution, which further enhanced detection performance and computational efficiency, allowing the model to achieve a good balance between processing speed and accuracy.

After 200 training epochs, the mask detection model demonstrated outstanding performance in terms of accuracy and processing speed, successfully meeting the expected research goals. This achievement not only validates the effectiveness of the YOLOv5 algorithm in object detection tasks but also provides reliable technical support and a practical foundation for mask detection in real-world applications.

Looking ahead, the experiment plans to further optimize the model and conduct more tests and applications in various real-world scenarios to enhance its practicality and applicability. This may include implementing the model in real-time detection systems in public places.

IV. CONCLUSIONS

The study proposes a face mask detection method that combines the YOLOv5 object detection algorithm with an improved ResNet50 network architecture. By incorporating deep convolutional neural networks (CNNs) for feature extraction, this approach enhances the ability to capture mask features while eliminating the need for precise face localization, thereby improving the model's practical application efficiency. The training process was optimized by adding dropout layers to ResNet50, utilizing the Adam optimization algorithm, and employing the cross-entropy loss function, significantly improving recognition accuracy.

After 200 training epochs, experimental results showed that the proposed method achieved an accuracy of over 85% in mask recognition tasks, with detection accuracy exceeding 98% on certain test datasets. Moreover, the model demonstrated strong robustness across diverse demographics, including age, gender, and ethnicity, proving its broad applicability in real-world scenarios. The combination of dropout layers and the Adam optimization algorithm further reduced overfitting and enhanced the model’s generalization capabilities.

Although this study achieved significant results, some areas still require further optimization. Future research directions include improving the model's adaptability to varying environmental conditions, particularly its ability to handle different lighting, angles, and occlusions; expanding the dataset to include more diverse scenarios and face orientations; and implementing real-time detection capabilities. Integrating this technology into public health monitoring and security detection systems could significantly enhance safety measures in various real-world settings. With further optimization and application, this study is expected to provide a solid foundation for the development of face mask detection technologies and contribute significantly to public health and safety efforts.

REFERENCES

[1] GARG P S. Face mask detection system using deep learning. International Journal for Modern Trends in Science and Technology，2020，6（12）：161-164.

[2] Aboul-Ella Hassanien, Kuo-Chi Chang, Tang Mincong. Advanced Machine Learning Technologies and Applications.:2021-03-08.

[3] Utku Kose, Omer Deperlioglu, D. Jude Hemanth. Deep Learning for Biomedical Applications. CRC Press:2021-03-04.

[4] Yann LeCun; Yoshua Bengio; Geoffrey Hinton. Deep learning. Nature: International weekly journal of science. 2015(7553).

[5] Ren Shaoqing, He Kaiming, Girshick Ross, Sun Jian. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE transactions on pattern analysis and machine intelligence, 2017, (6).

[6] Nan Xiaohu, Ding Lei. A Review of Typical Object Detection Algorithms in Deep Learning. Journal of Computer Applications Research, 2020, 37(S2): 15-21.

[7] Redmon J, Divvala S, Girshick R. You Only Look Once: Unified, Real-Time Object Detection //2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA,2016: 779-788.

[8] Redmon J, Farhadi A. YOLO9000: Better, Faster, Stronger//2017 IEEE Conference on Computer Vision and Pattern Recognition(CVPR), Honolulu, HI, USA, 2017: 6517-6525.

[9] Ioffe S, Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift// International Conference on International Conference on Machine Learning. JMLR.org, 2015.

[10] He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition//2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016: 770-778.

[11] Chen Xiaoqing. Research on UAV Remote Sensing Image Mosaic Technology. Guizhou University, 2012. DOI: 10.27047/d.cnki.ggudu.2021.000408.

[12] Liu Ting, Luo Peiqi, Fan Yunsheng. Overview of SSD-Based Detection of Small Targets on the Sea. Journal of Dalian Maritime University, 2022, 48(04): 65-75.