Research on MTCNN Face Recognition System in Low Computing Power Scenarios

YingGang Xie[1,2], Hui Wang[1], ShaoHua Guo[1]

[1] School of Information & Communication Engineering, Beijing Information Science and Technology University, China
[2] Key Laboratory of the Ministry of Education for Optoelectronic Measurement, Technology and Instrument, Beijing Information Science & Technology University, China
xieyinggang@bistu.edu.cn, 1374207430@qq.com, guoshaohua0930@163.com

Abstract

This paper analyzes the time-consuming analysis of each cascading network module (PNet module, RNet module and ONet module) in MTCNN, and finds that the time-consuming of PNet module is the highest (about 70%). According to the results of time-consuming analysis, two improved methods are proposed, one is to reduce the number of candidate face frames in input PNet network and the other is to reduce the number of output face frames in PNet network. Then, aiming at the problem that MTCNN algorithm has low detection speed in high-resolution video and cannot meet real-time requirements, a series of optimization such as adjusting the minnisize parameter and PNet threshold in combination with the low computing power application scenarios of the channel bayonet. It is verified in the FDDB face test set and practical application, the detection speed has increased by 70.1% when the detection rate has dropped by only 3.5%, and the improved scheme has achieved good results. Compared with the performance of OpenCV-VJ and SURF face detection algorithms on FDDB, the optimized MTCNN algorithm has better performance. Through the analysis of the detection results of the specific FDDB data set pictures, it is found that the undetected face conditions do not meet the actual application scenarios in this article, which proves that the optimized algorithm has excellent performance in actual applications. The test results reflect that the reproduced and optimized MTCNN face detection algorithm has good robustness to face pose changes, and fully meets the requirements of face recognition systems in low computing power scenarios such as channel bayonet.

Keywords: MTCNN, Low computing power scenarios, Face detection, PNet time-consuming optimization

1 Introduction

A mature face recognition system usually consists of image acquisition, image preprocessing, face detection, face tracking, face alignment, feature extraction and comparison. Among the more critical steps are face detection, tracking and face feature extraction. In recent years, face recognition systems have been widely used in channel bayonet systems such as smart access control and identity verification in high-speed railway stations. These channel bayonet face recognition systems have all or most of the face image collection, face detection, face alignment, face quality detection, face feature extraction, face tracking and other steps. However, some of these systems require a high degree of cooperation from people, some are complex to implement, and some have high requirements for hardware such as computing devices. On the one hand, the computing power of embedded systems is not enough to support face detection, tracking and face feature pairing based on deep learning. Real-time requirements, some channel bayonet face recognition systems require people to deliberately approach the camera to cooperate with the system for verification, discarding the natural and convenient advantages of face recognition. The specific target scenario studied in this paper is a single-channel bayonet (single face close range), and the goal is to be able to quickly compare and recognize faces within 1-4 meters. The goal of the research is to apply a faster and better performance algorithm to the channel bayonet face recognition system with low computing power, and to improve the operating speed of the face recognition system through the improved face detection algorithm. It can be mounted on low-end devices with poor computing performance while maintaining certain detection and recognition performance.

Aiming at the above problems, this paper combines the channel bayonet application scenarios, through research and improve face detection algorithms, face tracking auxiliary algorithms, and introduce face selection algorithms in the face recognition module to satisfy the real-time and convenience requirements of the channel bayonet face recognition system.

*Corresponding Author: YingGang Xie; E-mail: xieyinggang@bistu.edu.cn
DOI: 10.3966/160792642020092105020

After investigating and studying the existing face recognition system and analyzing its advantages and disadvantages, this paper improves the key technologies of the face recognition system. Firstly, using Python language to reproduce the MTCNN face detection algorithm on the TensorFlow deep learning framework, and it is optimized and improved by combining with the actual application scene of channel bayonet in this paper. The fusion of the improved speed-up MTCNN algorithm and the supervised Kalman filter human tracking algorithm proposed in this paper accelerates the system operation speed. According to the face selection algorithm proposed in this paper, a face selection module with multiple purposes is designed. Then according to the comparison and analysis of various face recognition algorithms, the FaceNet face recognition algorithm is selected, and the face recognition module is designed in combination with the face selection algorithm proposed in this paper. Finally, the above research is applied to the channel bayonet face recognition system, and the system design and implementation are completed. The chapter arrangement and corresponding main work of this paper are as follows.

The first part is the introduction and the second part is overseas and domestic research status, which explains the research background and significance of face recognition system; This paper analyzes the development status of face detection in the system at home and abroad, and compares and analyzes various algorithms, paving the way and explaining the reasons for the selection of various algorithms in the following research work. The third part introduces the common data set of face detection. The fourth part and the fifth part choose MTCNN algorithm as the face detection algorithm in the system according to the investigation and experimental comparison of face detection algorithm. Since there is no engineering implementation of MTCNN algorithm based on Python language and TensorFlow framework, it is reproduced and trained after an in-depth understanding of its convolutional neural network structure and detailed training methods. Afterwards, the time-consuming analysis of all levels of the network in MTCNN was carried out, and it was found that the PNet module took the highest proportion of time, and the corresponding speed-up suggestions were put forward. Then, in view of the low detection speed of the MTCNN algorithm in high-resolution videos, it cannot meet the real-time problem, combined with the specific application scenarios of the channel bayonet face recognition in this paper, a series of optimizations such as minnisize adjustment and PNet threshold adjustment are performed. Finally, according to the detection rate on the FDDB face test set and the actual detection speed in 720p high-resolution video, it can be seen that the detection speed increases by 70.1% when the detection rate drops by 3.5%. By analyzing the detection results of specific FDDB data set images, it is found that the undetected face conditions do not meet the practical application scenarios in this paper, which proves the excellent performance of the optimized algorithm in practical application. Finally, the face detection module was designed and implemented, and actual engineering tests were carried out; the test results reflect the specific data that the reproduced and tuned MTCNN face detection algorithm has good robustness to face pose changes and fully meet the requirements of the channel bayonet face recognition system. The sixth part is a summary and outlook. It summarizes the main research content of this paper, analyzes the affirmative part of the research work and the problems that need to be studied, and put forward the research direction for future work.

2 Overseas and Domestic Research Status

Face detection algorithms are divided into knowledge-based, feature-based, statistics-based, and deep learning-based algorithms. Face detection algorithms based on deep learning have achieved remarkable success in recent years [1]. This article only discusses the part of Convolutional Neural Networks (Convolutional Neural Networks, this chapter is referred to as “CNN”) in deep learning. In the task of face detection, CNN has been successfully introduced as a face feature extractor. The face detection algorithm based on CNN has strong robustness to influencing factors such as face pose changes, illumination changes, blur and so on [2]. CNN-based face detection algorithms can be roughly divided into cascade-based, classification-based and logistic regression-based algorithms.

The proposed face detection algorithm based on cascaded CNN is inspired by the VJ framework. The face detection algorithm based on the VJ framework is fast, and the face detection algorithm based on CNN has high accuracy. A natural idea is to combine the advantages of these two algorithms. The face detection algorithm based on cascaded CNN puts CNN into the cascaded structure to improve the detection accuracy and speed [3]. A feedback Radial Basis Function neural network (FRBF) is proposed to estimate the missing attribute values for incomplete data. The error between the actual output value of RBF neural network and the expected value is fed back to the input layer [4], then a feedback RBF neural network is constructed. In addition to using multiple CNN networks to cascade, different network layers in the same network are also used to form a cascade structure [5]; The first few layers detect faces that are easier to detect, and the latter layers detect faces that are difficult to detect. The most famous MTCNN [6] (Multi-task convolutional neural network, multi-task convolutional neural network) face detection algorithm integrates three CNNs into one CNN model, and integrates face

detection and face alignment tasks into one framework for implementation [7]. Combining different methods of MTCNN has different detection advantages in different situations [8]. For example, Zhang [9] et al. used the multi-task convolutional neural network (MTCNN) under the CaffeOnACL framework for face detection, and adopted local binary mode (LBP) As a face recognition algorithm, it is fast and accurate. Sabbir Ejaz [10], guo [11] and wang [12] et al. proposed a feasible method, which includes first detecting the facial area and using multi-task cascaded convolutional neural network (MTCNN) to solve the face occlusion Then use the Google FaceNet embedding model to perform facial feature extraction. Finally, the classification task has been performed by a support vector machine (SVM), which has excellent performance in masked face recognition. Lu [13] et al. proposed that a multi-task cascaded convolutional network (MTCNN) was used to detect all faces in the image, and then, by using a deep convolutional neural network and performing transfer learning from a pre-trained VGGFace model, the effect was good. Its overall complexity is well controlled and can be applied in some industrial scenarios [14].For example, Yi [15]et al. proposed the MTCNN-based facial occlusion recognition research in railway face-scrolling scenes, and Antony [16]and deng [17]et al. proposed the MTCNN-based driver fatigue detection.

A series of CNN face detection algorithms based on the classification of candidate regions is one of the most important branches in the current face detection technology field. This type of algorithm first generates many candidate face frames, and then uses the CNN network to determine whether there are faces in the candidate frames. The most famous algorithms in this series are R-CNN [18], SPP-NET [19], Fast R-CNN [20] and Faster R-CNN [21], etc. The detection process becomes more and more simple, and the detection speed has also been steadily improved. HyperFace [22] face detection algorithm uses a selective search algorithm to generate candidate frames, and the subsequent Faster-CNN algorithm uses a region proposal network to generate candidate frames. Face-RCNN [23] is a face detection algorithm designed based on the Faster-RCNN network and the central loss function. CMS-RCNN [24] face detection network introduces the contextual reasoning of the face into the Faster-RCNN face detection algorithm, thereby reducing the detection error rate. J. J. Li [25] achieves state-of-the-art results over prior arts on both the WIDER FACE dataset and the Face Detection Dataset and Benchmark. SSH [26] face detection algorithm introduces the candidate region suggestion network in the Faster-RCNN network into the VGG network structure, and at the same time, it also achieves good detection effect by removing its full connection layer.

The CNN target detection algorithm based on logistic regression is represented by YOLO [27] and SSD [28]. After training on the face set, it can complete the classification of the face and the regression of the face bounding box in the face detection at one time. The subsequent FaceBoxes [29] were inspired by the face region extraction network in Fast R-CNN and the multi-scale mechanism in SSD, and proposed a neural network that only contains fully convolution and can be trained end-to-end. Y. T. Chang [30] compared several algorithms of defect detections using a data set, which comprises 20 categories of objects and 50 images in each category. Cai et al. proposed the MS-CNN [31] face detection network. In order to find faces of different sizes, face detection is performed on multiple levels of the network. Wang et al. proposed the FAN [32] face detection algorithm based on the RetinaNetnetwork structure, J. S. Li [33] using the attention mechanism to enhance the network to extract facial features. The SRN [34] face detection network is also improved based on the RetinaNet network structure, adding binary classification and regression tasks, and fine-tuning the position of the anchor node on the high-level feature map. Pyramid [35] face detection network uses background information to improve the performance of face detection.

In summary, on the FDDB data set, the three types of algorithms based on deep face detection are shown in Table 1.

Table 1. Comparison of speed and accuracy of three types of methods

	Speed	Precision	Representative algorithm
Based on cascade	Fastest	Lower	Cascade CNN [4] MTCNN [6]
Based on classification	Slower	Higher	ICS [7]
Based on logistic regression	Faster	Higher	Faster R-CNN [21] Face R-CNN [23] Face R-FCN [25]

3 Common Data Sets for Face Detection

3.1 FDDB Data Set

FDDB data set has a total of 2845 images. These images contain 5,171 faces. They are one of the most authoritative face-detection evaluation datasets in the world. FDDB face test data set contains both black and white and color images, and faces contain different pose, occlusion, resolution and other factors that affect the detection rate, as shown in Figure 1. In addition, this data set is large, so it is more challenging to evaluate the face detection algorithm on this data set. Moreover, the author provides a prescribed procedure to evaluate the results, so it is fair to evaluate the detection algorithm on this data.

Figure 1. FDDB data set

The image resolution of the FDDB data set is small. The resolution of all images is less than 450×450, and the smallest marked face size is 20×20. The relevant training methods include ten fold cross validation based on FFDB data set and unlimited training based on isolated FDDB data set. However, due to the small number of FDDB data sets, most of them use unlimited training. The detection result has two methods: discrete ROC and continuous ROC. Discrete ROC focuses on whether the intersection ratio between the detection frame and the labeled frame is greater than 0.5, while continuous ROC focuses on whether the intersection ratio between the detection frame and the labeled frame is close to 1. Since most of the methods of unlimited training are used, the detector will be affected by the training set, so discrete ROC is more reliable.

3.2 WIDER FACE Data Set

The WIDER FACE data set contains 32,203 images, including 393,703 faces marked on them. Faces contain various scale changes, posture changes and other factors affecting the detection, which can be used as a training set or a test set. As shown in Figure 2.

Figure 2. WIDER FACE data set

The resolution of the images in the data set is generally high, and the width of all images are scaled to a size of 1024 in length and width; the data set is all color images, and the minimum labeled face size is 10×10. The whole data set is divided into training set, validation set and test set according to the ratio of 4:1:5, but the true label value of the test set is not public, and the test results need to be submitted to the official for comparison, which is more fair.

4 Comparison and Analysis of Related Algorithms

Traditional face detection algorithms use classifiers for classification after extracting artificially designed features by human. Classical algorithms include Adaboost algorithm based on Harr features, SVM algorithm based on HOG features, and so on. These classical algorithms tend to lose one way or the other in detecting speed and accuracy. In recent years, with the development of deep learning, the use of convolutional neural networks in face detection technology improves accuracy while also taking into account the detection speed.

In the two-stage detection network, Faster R-CNN completes the detection task through classification and regression and has a high mAP (mean Average Precision), but its own RPN (Region Proposal Network) network generates too much RoI (Region of Interest), resulting in a large amount of calculation and slow detection speed. Face R-CNN is a representative algorithm of two-step Face detection. It is an improved face detection algorithm based on the Faster R-CNN framework. Its characteristics are high accuracy but poor speed. In a one-stage detection network, the more effective solutions include YOLO and SSD. They both complete the target detection through regression. YOLO and SSD are adjusted after the target box is manually defined by the anchor node, which reduces the number of RoI and accelerates the speed of the algorithm; among them, YOLO can fully meet the industrial real-time requirements, but when transplanted into the face detection algorithm, the detection effect of small targets such as faces is poor. In practical applications, most faces occupy a small proportion in the image, so they cannot be detected. Although the accuracy of the SSD algorithm is good, too many model parameters require the support of high-performance graphics card, which cannot meet the real-time requirements of products on devices with only CPU or embedded devices.

Among the cascaded CNN detection algorithms, MTCNN has good robustness to face posture changes and occlusion, and it is one of the few cascaded CNN face detectors that can be applied in industrial scenes. In practical application, although its own cascade structure limits the detection speed in the case of multiple faces, it can still meet the actual requirements of product detection performance and real-time performance in the common channel bayonet scene (1 to 3 people) after industrial level optimization. Considering that the MTCNN algorithm is a multi-task

network structure, it can not only complete the face detection function, but also do the task of marking the key points of the face. A network completes two tasks and provides key points for the face alignment task. The integration of the recognition system will be higher and the running speed will be faster, so the MTCNN algorithm is selected as the face detection algorithm in this system.

4.1 MTCNN Algorithm

The idea of the MTCNN algorithm is to scale the image to different sizes as the input of each network layer, and then using the idea of rough to fine to sort three independent network which detection accuracy from poor to good. Then a cascade structure consisting of three convolutional neural networks is constructed to accomplish the multi-task detection target. MTCNN algorithm can complete three tasks at the same time: face detection, face border regression and face feature point positioning. Because the three tasks require different training labels, different loss functions are required.

The face detection task uses the two-class cross-entropy loss function:

$$L_i^{\det} = -\left(y_i^{\det}\log(p_i) + \left(1-y_i^{\det}\right)\left(1-\log(p_i)\right)\right). \quad (1)$$

Among them, p_i represents the possibility that the sample x_i is a human face, and $y_i^{\det}$ is 0 or 1.

The bounding box regression and key point tasks use the L2 loss function:

$$L_i^{\det} = \| \hat{y}_i^{\det} - y_i^{\det} \|_2^2 . \quad (2)$$

Where $\hat{y}_i^{\det}$ is the regression box of network output and $y_i^{\det}$ is the true value.

$$L_i^{landmark} = \| \hat{y}_i^{landmark} - y_i^{landmark} \|_2^2 . \quad (3)$$

Among them, $\hat{y}_i^{landmark}$ is the key point coordinates of the network output, and $L_i^{landmark}$ is the true value.

The total loss function is:

$$\text{loss} = \sum_{i=1}^{N}\sum_{j\in\{\det, bbox, landmark\}} \alpha_j \beta_i^j L_i^j . \quad (4)$$

Where α_j represents the importance of different tasks, β_i^j is an indicator of sample type, and L_i^j has different loss functions in different training samples.

MTCNN puts forward the idea of online hard sample training. In the training process, the hard samples that are difficult to train are selected online to accelerate the convergence of the network. The processing method is to sort the batches of the current batch with forward propagation loss in each minibatch, and find out the top 70% difficult samples. Only difficult samples are used for training in back propagation. The MTCNN algorithm process is roughly that the image completes related tasks through three cascaded networks.

The first is the data preprocessing stage. The MTCNN algorithm responds to face size changes by building an image pyramid, scaling the original image to different sizes through a certain scale factor, and building an image pyramid as the input data of the network cascade architecture.

In the first stage, all images in the image pyramid are obtained through a shallow full convolutional neural network PNet to obtain the candidate face frame and the face frame regression (the face frame regression is used to correct the position of the candidate face frame), so as to realize the role of rapidly generating the candidate face frame. Then use the NMS (Non-Maximum Suppression, non-maximum suppression) algorithm to merge the candidate face frames with a high overlap rate.

In the second stage, the candidate face frame generated in the first stage is used as the input of RNet; RNet is more complex than PNet in network structure, which can remove most of the wrong candidate face frame and then use the face frame regression vector to fine-tune the position of the candidate face frame. Then use the NMS algorithm to reduce the face frame.

The process of the third stage and the second stage are similar, and both use the output of the previous stage as the input of this stage. Adjust the position of the face frame while removing the wrong candidate frame, and output the position coordinate information of the five face feature points. It's just that the network structure of ONet is more complex than RNet, and the output results are more accurate.

The network structure of MTCNN is shown in Figure 3.

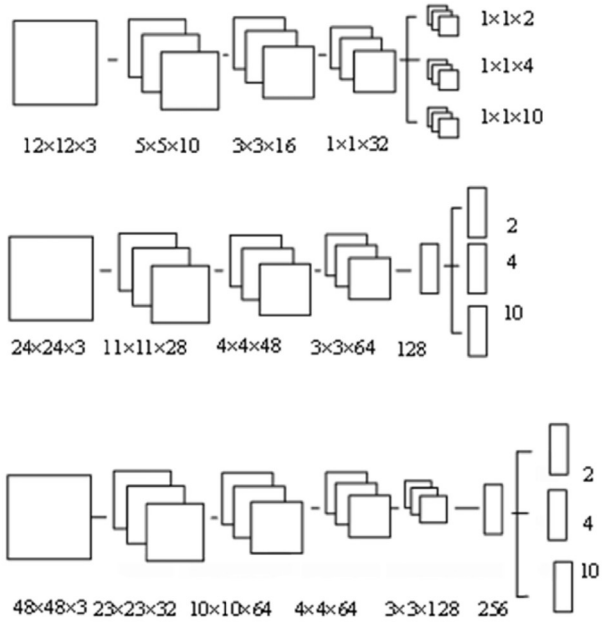

Figure 3. Network structure of MTCNN

4.2 Network Time-consuming Analysis

The face detection algorithm is proposed to solve two major problems, one is to detect whether the image contains a face, and the other is how to extract the position information of the face. The process of most face detection algorithms can be divided into two parts. The first step is to find all candidate areas that may contain faces in the image, and the second step is to select the candidate regions containing the highest probability of faces from these candidate regions. The MTCNN algorithm uses PNet to find the candidate area of the face, and uses RNet and Onet to further select the face candidate area with the highest probability. Generally, this kind of algorithm will be outstanding in the detection rate, but the speed will not be too ideal. In the following, the time-consuming of MTCNN network at all levels will be analyzed one by one to determine the improved method.

The average time consumption of PNet module, RNet module and ONet module was respectively calculated by sending 200 single-person face video frame images in each group into MTCNN face detector, as shown in Table 2. The PNet module, RNet module and ONet module here respectively include the preprocessing of input data, network operation and NMS algorithm, not just the process of model calculation [36].

Table 2. Time-consuming statistics of MTCNN after modifying parameters

parameter settings	PNet module/ms	RNet module/ms	ONet module/ms	Total time/ms
Control group	120.2	39.0	18.4	177.6
minisize40	42.1	12.7	4.7	59.5
minisize60	27.1	12.7	4.2	44.0
minisize80	19.8	11.1	4.5	35.4
PNet threshold 0.5	47.3	17.4	9.1	73.8
PNet threshold 0.7	37.7	11.0	4.5	53.2
PNet threshold 0.8	33.7	10.1	4.2	48.1
scale_factor0.6	24.1	11.0	4.4	39.4
scale_factor0.5	19.1	13.1	4.9	37.1

Among them, the control group parameters: the PNet threshold is 0.6, the minimum face size is 20, the pyramid scaling factor is 0.709, and the image resolution is 720p. In addition, the minisize parameter set for the PNet threshold change group and the scale_factor change group is 40, and the other parameters are the same as the control group. Timing method: use the cv2.getTickCount() function to time each module before and after running. The cv2.getTickCount() function here returns the clock cycle from the start of the program to the current code execution; for example, set the timing function before and after the PNet module t_1 = cv2.getTickCount() and t_2 = cv2.getTickCount(), the module running time can be calculated by t=($t_2 - t_1$)/cv2.getTickFrequency(), the function cv2.getTickFrequency() returns the number of clock cycles of the CPU in 1 second, so it can be timed in seconds.

According to the data analysis in Table 1, it is concluded that the PNet module stage consumes the most time. However, when we analyze the three cascaded network structures in MTCNN, we can see that the complexity of the network models of PNet, RNet, and Onet increase sequentially, which means that their parameter quantities are also increased sequentially, and their running time should also increase sequentially. This is inconsistent with our experimental results. After careful analysis of the algorithm flow of MTCNN, it is found that in the data processing stage, the MTCNN algorithm adopted the image pyramid method to scale the original image with a coefficient of 0.709 in order to solve the problem of face scale change, and made the face in the image close to the image size (12×12) required by PNet training. The implementation method is to first reduce the original image by 12/minisize (minisize is the hyperparameter, that is, the smallest face size in the image, set to 20 in MTCNN), and then reduce it to a size close to 12×12 by the scale factor. As mentioned above, when the zoom factor is smaller and the minisize is larger, the fewer pyramid images are generated, and the image input into PNet will be relatively reduced. If the resolution of the original image is high, the data processing stage will be very time-consuming, and PNet will accept a lot of input images. In combination with actual scenes, the face size that needs to be recognized in the channel bayonet is often larger than 20×20. The minisize could be scaled up for high resolution images or the scaling factor could be lowered to reduce the amount of pyramid images and speed up the testing.

The number of face frames output by the previous subconvolutional network will be reduced after being eliminated by the NMS; therefore, the number of face frames that the PNet module needs to process is the largest, followed by the RNet module, and the ONet module least. This is also the cause of the most time-consuming part of the PNet module. Therefore, adjust the default settings of the network threshold [0.6, 0.7, 0.7] to [0.7, 0.7, 0.7] and [0.8, 0.7, 0.7] and re-test. Combining the data in the table and the algorithm flow of each module, it can be analyzed that as the PNet threshold increases, the amount of data that the NMS algorithm needs to process is reduced, and the time consumption of the PNet module is reduced; and the number of pictures sent to the RNet module is reduced. The RNet module consumes less time. Therefore, in the subsequent testing stage, the network threshold can be appropriately increased to speed up the network operation [37].

In summary, the following conclusions can be drawn: (1) PNet module part consumes the most time, so in terms of speed improvement, it gives priority to

improve PNet module. (2) In the actual scene, the minisize (recognized minimum face size) can be adjusted to adapt to factors such as image quality and distance, which can speed up the operation of the face detector at the same time [38]. (3) The scaling factor and network threshold can be adjusted appropriately to speed up the running speed.

Since the specific scene of the application of MTCNN face detection algorithm trained in this paper is the channel bayonet, such as the single channel bayonet of the railway station entrance examination bayonet, intelligent face recognition access control, etc., the detection personnel need to be close to the image collection equipment. The size of the face in the channel bayonet video frame is much higher than the 20x20 resolution. In MTCNN, the relevant coefficient minisize can be adjusted according to the minimum face size in the image to be detected. According to the experimental data obtained from the time-consuming analysis part of the MTCNN algorithm in this article, the minisize could be adjusted from the original default parameter 20 to 40 under a single person face, and the number of detection frames increases, and the speed is increased significantly. Therefore, the solution adjusts the minimum face size to be detected to 40×40 according to actual application scenarios. And on this basis, it analyzes the influence of further adjusting the network threshold on the detection rate and detection speed of the MTCNN algorithm.

5 Experiment and Analysis

5.1 Network Training

Since the MTCNN algorithm does not have the corresponding official code, and most of the reproduction projects are built on the caffe deep learning platform, the deep learning platform chosen in this article is tensoflow. In this way, MTCNN needs to be duplicated first, and then the model can be improved based on it. Finally, the algorithm performance before and after the improvement can be compared on the same platform TensorFlow.

In terms of project implementation, the software used and the environment built are: Anaconda, pycharm, py3.6, tensorflow1.8 and their dependent packages. The hardware used is: CPU i5-7300HQ 2.5GHz, memory 16GB, GPU is GTX1050.

When building network models of PNet, RNet and ONet, this paper selects the TF-Slim library in TensorFlow to implement. TF-slim is a lightweight deep learning library built into TensorFlow, making the process of testing, training and building models very simple. The convolutional layer, pooling layer, full connection layer and deep separable convolutional layer to be used in this paper can all be implemented by using the TF-Slim library. For example, the full connection layer can be realized using the slim.fully_ connected() library function, with the number of network input and output neurons indicated in parentheses.

Then the corresponding function modules are built according to the algorithm flow and training process of MTCNN. The algorithm flow has been described in detail in Section 3.4, and it will not be repeated in this section. MTCNN is a three-stage cascade neural network, and the training process is divided into three steps to conduct separate training for PNet, RNet and ONet. Each network requires its own training set, and the output of the previous network is the training input of the current network. The training set of PNet uses 12×12 pictures, which are obtained after fine-tuning up and down based on the actual label information of the face on the prepared data set. The image interception was divided into positive sample, negative sample, partial face sample and key point sample according to the numerical size of IoU(Intersection over Union, handover ratio), with a ratio of 1:1:3:1. The positive sample, negative sample and part of the face samples are randomly clipped. The maximum IoU value of cropped image and face frame is greater than 0.65 as positive samples, those with greater than 0.4 and less than 0.65 are partial face images, and those with less than 0.3 are negative samples, and the image with key points is taken from the key point sample. The training sets of RNet and ONet need to be intercepted and normalized from the original image by the regression box information output by the network of the previous layer, and only 70% of the training data before the classification loss is taken for the difficult case mining training. The training process needs to be divided into three parts according to the MTCNN algorithm flow and trained in sequence.

The training set of the face detection part is WIDER_Face_train data set. The training sets of face key point detection are LFW_5590 and NET_7876 data sets. According to the original paper, the thresholds of the three networks were set to 0.6, 0.7, 0.7 (set according to the MTCNN paper), the initial learning rate was set to 0.001, and the minisize was set to 20 (the minimum face size marked in FDDB is 20×20). The epoch of each network (1 epoch means all samples in the training set for 1 pass) were set to 30, 22, 22, and batch_size was set to 384.

5.2 Performance Analysis

In this paper, FDDB is used as the test set to compare the performance of the duplicated MTCNN and the two improved schemes. The FDDB data set provides prescribed procedures to evaluate face detection algorithms. But the program it provides is written in C/C++ language, needs to be compiled by make, and the corresponding C/C++ version of opencv needs to be configured in Visual Studio. The environment building process is complicated, and the

C/C++ language environment of Visual Studio that needs to be built conflicts with the Python language environment of PyCharm that has been built in this article. For example, the py-opencv library in the built py3.6 environment conflicts with the C/C++ version of opencv and cannot be in the same system. Based on the above reasons, this article packages the evaluation program provided in FDDB into an exe executable program, and places the required opencv-related files opencv_world310d.dll and opencv_world310.dll in the same directory, completing the migration of the evaluation program. The data required by the evaluation program is the detection result of the face detector on the FDDB data set, so this article first designs the program fddbout.py to record the position and size information of the face frame obtained after all the images in the FDDB data set through the face detection algorithm.

Perform performance analysis below. According to the time-consuming analysis results of the control group and the minisize parameter variation group in Table 2, the maximum speed improvement was achieved when the value of minisize was elevated from 20 to 40, which changed from 177.6ms/ frame to 59.5ms/ frame, reducing the time consumption by 66.5%. The performance of a variety of replicating MTCNN algorithms over FDDB was tested. The test results are shown in Figure 4. The thresholds of the three cascaded networks were set as 0.6, 0.7 and 0.7 respectively.

As can be broadly seen in Figure 7, the performance declined as minisize increased. The specific data are shown in Table 3.

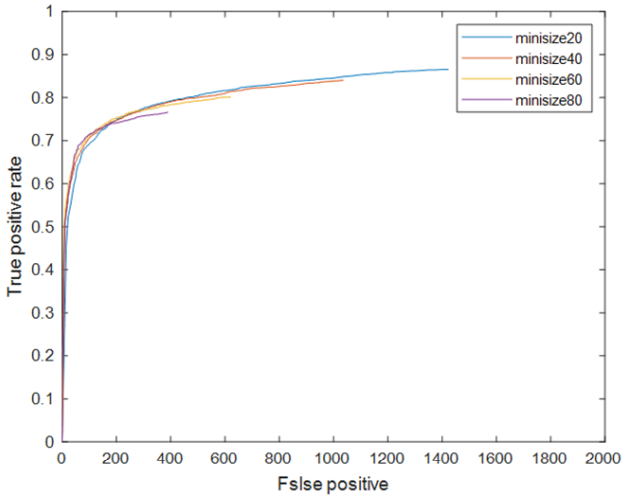

Figure 4. Algorithm performance comparison under different minisizes

Table 3. Comparison of algorithm performance under different minimum face parameters

minisize	Maximum accuracy	FP=400 accuracy	Number of FP	Average Detection time/ms
20	0.866	0.792	1423	177.6
40	0.840	0.790	1036	59.5
60	0.801	0.784	620	44.0
80	0.766	0.766 (FP=389)	389	35.4

As can be seen from Table 2, the detection rate has dropped by 2.6 percentage points when the minisize changes from 20 to 40, and the detection rate has dropped by 3.0%; The average detection time decreased from 177.6ms to 59.5ms, a decrease of 118.1ms and a relative decrease of 66.5%. The time consumption was significantly reduced compared to minisize60 and minisize80. The experimental data prove that the scheme is completely feasible.

On the basis of changing the minisize from 20 to 40, we will explore whether changing the PNet network threshold can further greatly reduce the time-consuming detection. Therefore, the PNet threshold is set to 0.5, 0.6, 0.7, 0.8 and tested on the FDDB data set. The test results are shown in Figure 5. The specific values are shown in Table 4.

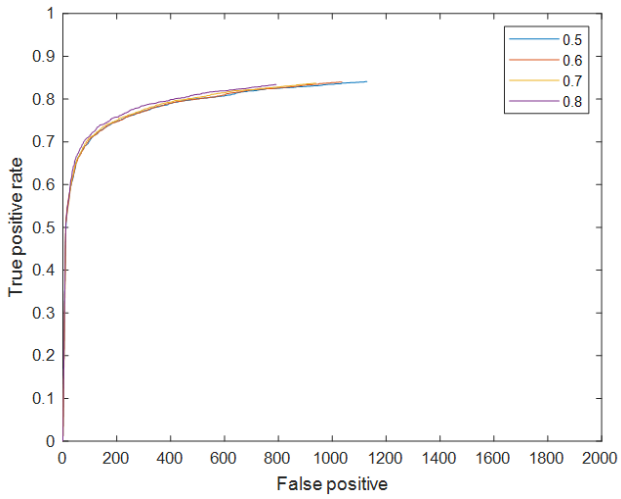

Figure 5. Comparison of algorithm performance under different thresholds

Table 4. Comparison of algorithm performance under different thresholds

PNet threshold	Maximum accuracy	FP=800 accuracy	Number of FP	Average detection time/ms
0.5	0.841	0.825	1129	73.8
0.6	0.840	0.826	1036	59.5
0.7	0.837	0.828	940	53.2
0.8	0.834	0.834 (FP=792)	792	48.1

After analyzing the data in Table 4, the detection rate and average detection time decreased slightly and changed little after increasing the PNet threshold value in the original setting.However, increasing the PNet threshold reduces the number of FP, which will reduce the number of false detections in practical application. Therefore, the PNet threshold is selected here as 0.7.

The eventual improvement was to adjust minisize to 40 and the PNet network threshold to 0.7, achieving an acceptable range of only 2.9% drop in detection rate (relative drop of 3.5%), and its time-consuming from 177.6ms/frame The reduction is 53.2ms/frame (the average time consumption is relatively reduced by 70.1%).

Finally, the repeated MTCNN algorithm, the adjusted MTCNN algorithm (minisize=40, PNet threshold of 0.7) and the built-in VJ algorithm based on Adaboost and the surf-based face detection algorithm were compared on the FDDB data set. The test results are shown in Figure 6, and the specific data are shown in Table 5. The results show that the algorithm in this paper is significantly better than the traditional VJ face detection algorithm and the face detection algorithm based on SURF features.

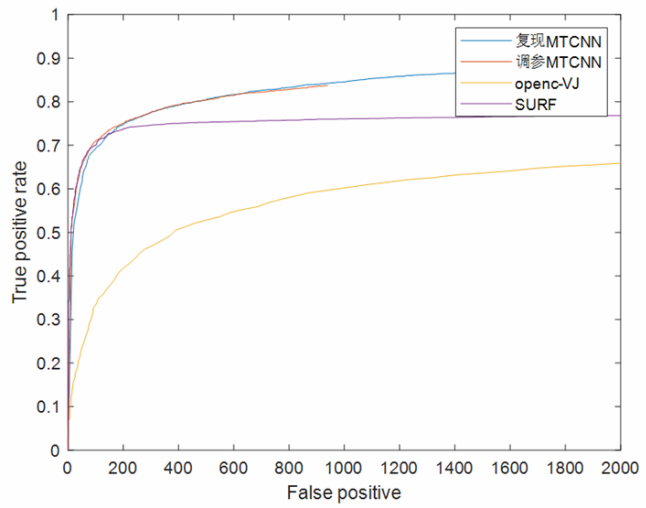

Figure 6. Performance comparison of various face algorithms

Table 5. Comparison of detection rates of various algorithms under different FP

algorithm	FP=200	FP=400	FP=600	FP=800
Repetition MTCNN	0.747	0.792	0.816	0.832
Improved MTCNN	0.749	0.793	0.814	0.828
opencv-VJ	0.411	0.513	0.552	0.583
SURF	0.737	0.750	0.755	0.757

5.3 Test Analysis

Figure 7 shows the difficult detection pictures of occlusion and pose changes on the FDDB data set. The ellipse box is the real-value face box given by the data set, and the box is the face box given by the MTCNN

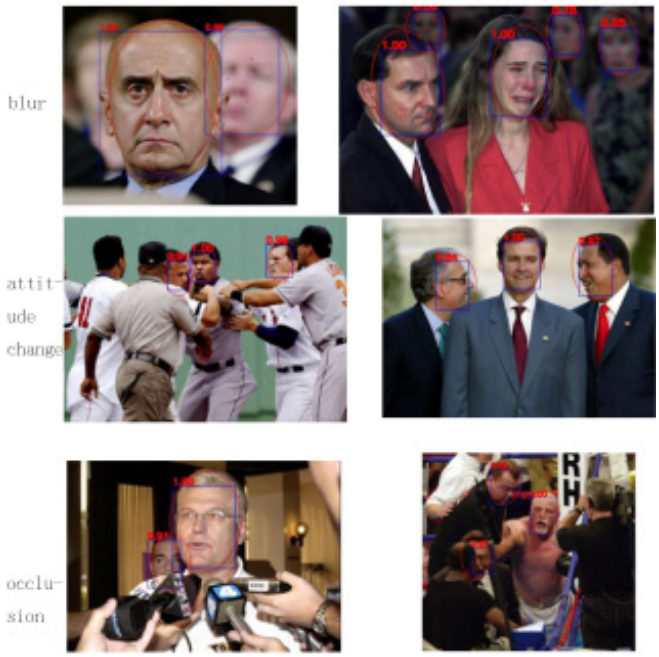

Figure 7. Detection results of blur, occlusion, and attitude change in FDDB

detection in this paper. It can be seen from the detection results that the MTCNN algorithm reproduced in this paper has good robustness to pose and occlusion.

As shown in Figure 8, there are some difficult images not detected on the FDDB data set by the reproduced MTCNN algorithm in this paper, and there are three typical cases with large attitude change, fuzzy and small face size, and excessive occlusion. The recognition rate of these faces into the face recognition module is very low, which will result in the waste of computing resources. Moreover, these situations have no practical significance in the face recognition of channel bayonet in this paper. Even if they are detected, they will be determined by the face selection module designed in Chapter 4 that does not meet the frontal face conditions and cannot achieve a certain degree of clarity and human face images whose face confidence is not up to the standard are eliminated.

In summary, the improved and optimized MTCNN algorithm in this paper not only has a good detection rate on the FDDB data set, but also shows that this algorithm will perform better in real applications through the analysis of specific hard cases.

The main problem of the channel bayonet face recognition system in this paper is the decline in the accuracy of face detection and recognition caused by face pose changes, so pose changes are the main test point in the module test.

The face detection module is the most important module of the face recognition system designed in this paper, which determines the performance of the face recognition system to a certain extent. The face detection module outputs the border of the face in the image and five key points of the face. The algorithm

Figure 8. Difficult example diagram of FDDB with missed detection

adopted by the detection module is the MTCNN face detection algorithm after modifying the minisize parameter and PNet threshold according to the actual application scenario. The main test method is to simulate the pose changes of the face in the real scene, and test whether it can detect the face and its key point coordinates under a variety of pose conditions. Figure 9 shows the test results.

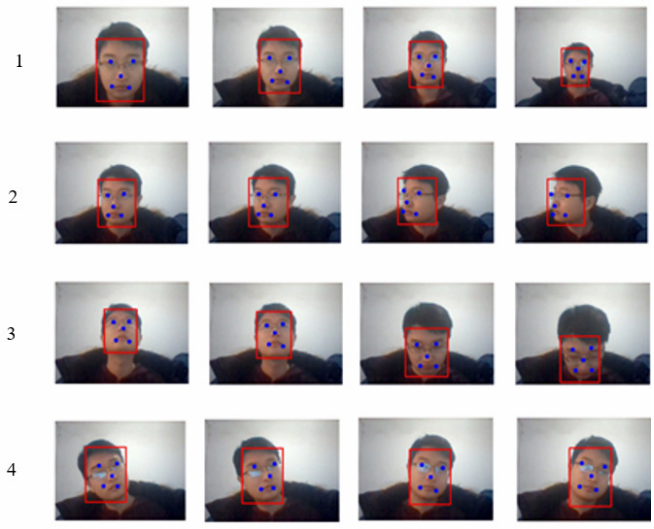

Figure 9. Test result of face detection module

The first group of test samples simulates the face detection effect in the case of frontal faces. From the first group of samples, it can be seen that in the case of frontal detection, no matter whether the size of the face changes or not, the face can be accurately detected and five faces are labeled key point. Then the second, third, and fourth groups of samples simulate various situations of face pose changes to test the detection of non-standard pose faces by the face detection module. It can be seen that the detection effect of the face detection module is not affected when the person is in the posture of head up, head down, side face, or tilted head. On the basis of the above video frames, a detection statistics table is made for the three pose changes of the face, and the detection results of the recurring MTCNN algorithm for the pose changes are counted. The statistical results are shown in Table 6. From the data in the table, it can be seen that it has good robustness to attitude changes, which fully meets the this system requires.

Table 6. Detection range in different postures

	Profile	Pitch	Tilt head
examination range (The front face is 0°)	Left face 75° Right face 71°	Head up 37° Head down 45°	Tilt your head left 41° Tilt your head right 35°

The face detection module designed in this paper plays a very important role in the system. Only when the face detection module is running normally and completing the task can the subsequent modules provide important information such as face frame position and key point coordinates. This module can also be used alone, with strong reusability, and can be used in other face recognition systems or various research work that requires the use of face detection modules.

5.4 The Experimental Summary

This chapter first introduces the factors and performance indicators that affect the detection rate of face detection algorithms, and then introduces commonly used face data sets. By analyzing some existing detection algorithms, the MTCNN algorithm is selected as the improvement object [39]. Then the MTCNN algorithm was reproduced on tensorflow and time-consuming analysis was made. On this basis, an improved scheme was proposed (the minisize parameter was adjusted from 20 to 40 and the PNet threshold was adjusted from 0.6 to 0.7). The experimental results show that the performance of the algorithm is reduced by 3.5% after optimization, but the speed is increased by 70.1%, achieving the purpose of improvement. Compared with other face detection algorithms, the results show that the algorithm in this paper has better performance. In the test and analysis stage, through the analysis of the specific detection images of the face detection algorithm on the FDDB data set, it is concluded that the algorithm has good robustness in terms of attitude change and occlusion to meet the design requirements. Finally, through the actual face detection module test and quantitative robustness analysis of pose changes, it is further verified that the reproduced MTCNN algorithm can

meet the design requirements of the channel bayonet face recognition system after adjusting the parameters.

6 Summary

With the continuous development of deep learning and computer vision in recent years, the research interest of face recognition system related technologies is increasing year by year. This paper uses channel bayonet as the actual application scenario to study the most critical face detection of the face recognition system, and analyzes the time-consuming analysis of each cascaded network module (PNet module, RNet module, ONet module) in MTCNN, and finds the PNet module takes the most time (about 70%). According to the results of time-consuming analysis, two improvement suggestions are put forward: reducing the number of face frames input to the PNet network candidates and reducing the number of face frames output from the PNet network. Then, aiming at the problem that MTCNN algorithm has low detection speed in high-resolution video and cannot meet the real-time requirement, a series of optimization such as minnisize parameter adjustment and PNet threshold value are carried out in combination with the specific application scene of channel bayonet face recognition in this paper and the speedup suggestion obtained from time-consuming analysis. According to the detection rate of the MTCNN algorithm on the FDDB face test set and the detection speed of the actual application in 720p high-resolution video, it can be seen that the detection speed has increased by 70.1% when the detection rate has dropped by only 3.5%, and the improvement plan has been achieved good effect. And the performance comparison with opencv-VJ and SURF face detection algorithm on FDDB shows that the optimized MTCNN algorithm performs better. By analyzing the detection results of specific FDDB data set images, it is found that the undetected face conditions do not meet the practical application scenarios in this paper, which proves the excellent performance of the optimized algorithm in practical application. Finally, the face detection module was designed and implemented, and actual engineering tests were carried out; the test results reflected from the specific data that the reproduced and tuned MTCNN face detection algorithm has good robustness to face pose changes, and fully meets the requirements of the channel bayonet face recognition system.

The shortcoming of the research work of this article is that in the selection of tools for implementing algorithms and modules, an environment based on the Python language is selected. In terms of speed, the Python language is not as good as C/C++. This puts forward higher requirements for the operating speed of modules and systems. In the future, you can try to replace the tensorflow framework with the caffe deep learning framework, which runs faster, and use the C/C++ language to develop related systems.

Acknowledgments

This work is supported by Beijing Natural Science Foundation (Grant No.4192023 and 4202024); The Qin Xin Talents Cultivation Program of BISTU (Grant No.QXTCPC201704)

References

[1] L. Y. Chen, F. Y. Zhao, Overlapped Face Detection Based on Deep Learning, *Computer Technology and Development*, Vol. 30, No. 2, pp. 28-32, February, 2020.

[2] F. Filipovic, M. Despotovic-Zrakic, B. Radenkovic, B. Jovanic, L. Živojinovic, An Application of Artificial Intelligence for Detecting Emotions in Neuromarketing, *2019 International Conference on Artificial Intelligence: Applications and Innovations (IC-AIAI)*, Belgrade, Serbia, 2019, pp. 49-53.

[3] D. Poster, S. Hu, N. Nasrabadi, B. Riggan, An Examination of Deep-Learning Based Landmark Detection Methods on Thermal Face Imagery, *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Long Beach, CA, USA, 2019, pp. 980-987.

[4] H. Luo, Q. Hou, Y. Liu, L. Zhang, Y. Z. Li, Fuzzy Clustering Algorithm for Interval Data Based on Feedback RBF Neural Network, *Journal of Internet Technology*, Vol. 21, No. 3, pp. 799-810, May, 2020.

[5] K. Zhang, Z. Zhang, H. Wang, Z. Li, Y. Qiao, W. Liu, Detecting Faces Using Inside Cascaded Contextual CNN, *IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 2017, pp. 3190-3198.

[6] H. C. Ku, W. Dong, Face Recognition Based on MTCNN and Convolutional Neural Network, *Frontiers in Signal Processing*, Vol. 4, No. 1, pp. 37-42, January, 2020.

[7] H. Qin, J. Yan, X. Li, X. Hu, Joint Training of Cascaded CNN for Face Detection, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 3456-3465.

[8] K. Chou, Y. Cheng, W. Chen, Y. Chen, Multi-task Cascaded and Densely Connected Convolutional Networks Applied to Human Face Detection and Facial Expression Recognition System, *2019 International Automatic Control Conference (CACS)*, Keelung, Taiwan, 2019, pp. 1-6.

[9] M. Zhang, W. Liao, J. Zhang, H. Gao, F. Wang, B. Lin, Embedded Face Recognition System Based on Multi-task Convolutional Neural Network and LBP Features, *2019 IEEE International Conference of Intelligent Applied Systems on Engineering (ICIASE)*, Fuzhou, China, 2019, pp. 132-135.

[10] M. S. Ejaz, M. R. Islam, Masked Face Recognition Using Convolutional Neural Network, *2019 International Conference on Sustainable Technologies for Industry 4.0 (STI)*, Dhaka, Bangladesh, 2019, pp. 1-6.

[11] C. Guo, Y. Yang, Implementation of a Specified Face Recognition System Based on Video, *2019 IEEE 4th*

Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chengdu, China, 2019, pp. 79-84.

[12] S. Ji, K. Wang, X. Peng, J. Yang, Z. Zeng, Y. Qiao, Multiple Transfer Learning and Multi-label Balanced Training Strategies for Facial AU Detection In the Wild, *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Seattle, WA, USA, 2020, pp. 1657-1661.

[13] G. Lu, W. Zhang, Happiness Intensity Estimation for a Group of People in Images Using Convolutional Neural Networks, *2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE)*, Xiamen, China, 2019, pp. 1707-1710.

[14] J. Du, High-Precision Portrait Classification Based on MTCNN and Its Application on Similarity Judgement, *Journal of Physics: Conference Series*, Vol. 1518, No. 1, pp. 1-9, April, 2020.

[15] S. Yi, J. S. Zhu, H. Jing, Face Recognition Technology Applies in Railway Scene Based on MTCNN Face Occlusion Technology Research, *Computer Simulation*, Vol. 37, No. 5, pp. 96-99, May, 2020.

[16] N. Antony, R. KR, S. Patel, S. S, N. M, Driver Drowsiness Detection Using Convoluted Neural Networks, *2019 1st International Conference on Advanced Technologies in Intelligent Control, Environment, Computing & Communication Engineering (ICATIECE)*, Bangalore, India, 2019, pp. 92-97.

[17] W. Deng, Z. Zhan, Y. Yu, W. Wang, Fatigue Driving Detection Based on Multi Feature Fusion, *2019 IEEE 4th International Conference on Image, Vision and Computing (ICIVC)*, Xiamen, China, 2019, pp. 407-411.

[18] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation, *IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, USA, 2014, pp. 580-587.

[19] K. He, X. Zhang, S. Ren, J. Sun, Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 37, No. 9, pp. 1904-1916, September, 2015.

[20] R. Girshick, Fast R-CNN, *IEEE International Conference on Computer Vision (ICCV)*, Santiago, Chile, 2015, pp. 1440-1448.

[21] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, No. 6, pp. 1137-1149, June, 2017.

[22] R. Ranjan, V. M. Patel, R. Chellappa, HyperFace: A Deep Multi-Task Learning Framework for Face Detection, Landmark Localization, Pose Estimation, and Gender Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 41, No. 1, pp. 121-135, January, 2019.

[23] B. Y. Yao, H. Zhou, J. H. Yin, G. Q. Li, C. C. Lv, Small Sample Image Recognition Based on CNN and RBFNN, *Journal of Internet Technology*, Vol. 21, No. 3, pp. 881-889, May, 2020.

[24] C. Zhu, Y. Zheng, K. Luu, M. Savvides, *CMS-RCNN: Contextual Multi-scale Region-based CNN for Unconstrained Face Detection*, https://arxiv.org/abs/1606.05413, 2016.

[25] J. J. Li, J. X. Wang, X. C. Chen, Z. X. Luo, Z. G. Song, Multiple Task-driven Face Detection Based on Super-resolution Pyramid Network, *Journal of Internet Technology*, Vol. 20, No. 4, pp. 1263-1272, July, 2019.

[26] M. Najibi, P. Samangouei, R. Chellappa, L. S. Davis, SSH: Single Stage Headless Face Detector, *IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 2017, pp. 4885-4894.

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

[28] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, A. C. Berg, *SSD: Single Shot MultiBox Detector*, https://arxiv.org/abs/1512.02325, 2016.

[29] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, S. Z. Li, FaceBoxes: A CPU Real-time Face Detector with High Accuracy, *2017 IEEE International Joint Conference on Biometrics (IJCB)*, Denver, CO, USA, 2017, pp. 1-9.

[30] Y. T. Chang, W. K. T. M. Gunarathne, T. K. Shih, Deep Learning Approaches for Dynamic Object Understanding and Defect Detection, *Journal of Internet Technology*, Vol. 21, No. 3, pp. 783-790, May, 2020.

[31] Z. W. Cai, Q. F. Fan, R. S. Feris, N. Vasconcelos, *A Unified Multi-scale Deep Convolutional Neural Network for Fast Object Detection*, https://arxiv.org/abs/1607.07155, 2016.

[32] T. Lin, P. Goyal, R. Girshick, K. He, P. Dollar, Focal Loss for Dense Object Detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 42, No. 2, pp. 318-327, February, 2020.

[33] J. S. Li, I. H. Liu, C. Y. Lee, C. F. Li, C. G. Liu, A Novel Data Deduplication Scheme for Encrypted Cloud Databases, *Journal of Internet Technology*, Vol. 21, No. 4, pp. 1115-1125, July, 2020.

[34] W. Ke, J. Chen, J. Jiao, G. Zhao, Q. Ye, SRN: Side-Output Residual Network for Object Symmetry Detection in the Wild, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 2017, pp. 302-310.

[35] X. Tang, D. K. Du, Z. Q. He, J. T. Liu, *PyramidBox: A Context-assisted Single Shot Face Detector*, https://arxiv.org/abs/1803.07737, August, 2018.

[36] W. Zhang, Z. Zhang, H. C. Chao, M. Guizani, Toward Intelligent Network Optimization in Wireless Networking: An Auto-learning Framework, *IEEE Wireless Communications*, Vol. 26, No. 3, pp. 76-82, June, 2019.

[37] Z. Zhang, W. Zhang, F. H. Tseng, Satellite Mobile Edge Computing: Improving QoS of High-speed Satellite-terrestrial Networks Using Edge Computing Techniques, *IEEE Network*, Vol. 33, No. 1, pp. 70-76, January/February, 2019.

[38] X. S. Jia, S. Y. Zeng, B. Pan, Y. Zhou, Fast Detection of

Target Face Based on the Improved MTCNN Network, *Computer Engineering and Science*, Vol. 42, No. 7, pp. 1262-1266, July, 2020.

[39] W. Zhang, Z. Zhang, S. Zeadally, H. C. Chao, V. C. M. Leung, MASM: A Multiple-Algorithm Service Model for Energy-Delay Optimization in Edge Artificial Intelligence, *IEEE Transactions on Industrial Informatics*, Vol. 15, No. 7, pp. 4216-4224, July, 2019.

Biographies

Yinggang Xie received the B.Sc. degree in Automatic Control of Engineering from University of Science and Technology Beijing, Beijing, China, in 2001, and the M.E. and Ph.D. degrees in Control theory and control engineering from the University of Science and Technology Beijing, Beijing, China, in 2003 and 2007 respectively. He is currently a professor at the Department of Internet of things, Beijing Information Science and Technology University, China, His current research interests include multiple working modes control design for modular and reconfigurable robots, collaborative robots, Internet of Things.

Hui Wang is currently a Master's degree in Beijing Information Science and Technology University, Beijing, China, Her current research interests include real-time reconstruction of unstructured scenes and multiple working modes control design for modular, robotic arm control, target recognition.

ShaoHua Guo received the B.Sc. degree in Bachelor of Engineering from the QingDao University of Science & Technology, ShangDong, China, in 2018. She is currently a Master's degree in Beijing Information Science and Technology University, Beijing, China, Her current research interests include the Internet of Things, machine vision, and face recognition.