International Core Journal of Engineering Volume 7 Issue 8, 2021
ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0067

Optimization of Face Detection Algorithm based on MTCNN

Gaiping Liu, Jianmei Xiao, Xihuai Wang

Shanghai Maritime University, Shanghai 201306, China.

Abstract

Face detection is an important fundamental problem in computer vision, and it is a key step for future applications such as face analysis, face verification, face labeling and retrieval. With the development of image processing and deep learning, face detection and recognition have been widely used in all aspects of life, so face detection technology has put forward higher requirements. In practice, the image acquired in face detection is greatly affected by the environment, the image is fuzzy and contains noise, and there are some difficulties in detection. In this paper, based on the MTCNN face detection algorithm to solve the above problems, first of all, the image using SRGAN ultra-high resolution restoration technology to image clearness processing, make the face clearer, more convenient detection. In order to better process the feature map, the InceptionV2 module was introduced to optimize the network structure of MTCNN. It further improves the accuracy of face detection in complex background. The training was validated on WIDER FACE and CelebA datasets. The final accuracy of the optimized algorithm can reach 98.8%, 2.4% higher than that of the unoptimized network, which better meets the application requirements of modern society for face detection.

Keywords

Face Detection; Network Optimization; Inception Module; Image Sharpening.

1. Introduction

With the development of society and the progress of science and technology, the research on face information processing has become one of the current research hotspots. It is the research basis of biometric recognition, video surveillance, human-computer interaction, security, content retrieval, video conferencing and other fields, and plays an important role in the development of the above fields [1]. Face detection is the basic link in face information processing, which directly affects the result of face information system processing [2]. Among them, frontal face detection is more widely used in practical application [1].

With the rapid development of deep learning, face detection based on CNN [8] develops rapidly. Classical convolutional networks, such as AlexNet [8], VGG Net [13], GoogLeNet [16] and ResNet [9], have achieved great development in image target recognition and detection based on deep learning. Current mainstream target detection frameworks include YOLO [12], SSD [6], Faster R-CNN [13] and Mask R-CNN [3], etc., all of which have achieved good results in various target detection data sets. Among the existing face detection methods, the representative ones include MTCNN [5], Faceboxes [7], Pyramidbox [4], and RefineFace [3]. Faceboxes and Pyramidbox are mainly for small face detection. MTCNN face detection has a high accuracy, but in a complex environment has a greater impact on the detection effect of MTCNN.

The paper is based on the structure of MTCNN face detection network, the image is preprocessed first. Introducing the image sharpening module in the image processing to make the face in the image clearer and easier to detect; Inception [16] module in GoogLeNet is introduced into the training network of each layer of MTCNN to improve the network's processing of feature map and further

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0067

improve the accuracy of network detection. The improved network structure is trained and verified on WIDERFACE and CelebA datasets. The improved MTCNN model has high accuracy and robustness, which further improves the accuracy and efficiency of face detection.

2. MTCNN

MTCNN is a face detection and face alignment algorithm based on deep convolutional neural network. MT (Multi-task), this method can simultaneously complete face detection and face alignment two tasks. Compared with traditional methods, MTCNN has better detection performance, which can more accurately locate the position of the face and meet the requirements of real-time detection. MTCNN is composed of three layers of PNet [4], RNet [4] and ONet [5]. The original image input first needs to generate the multi-scale image pyramid, and then the candidate region is generated by the full convolution form of PNet. The candidate box with high degree of correspondence is removed by non-maximum suppression, and the correction is carried out by box regression. The output data of PNet layer is used as the input data of RNet layer, and the output data of RNet layer is used as the input data of ONet layer. The image is processed by three-layer network to detect the face and the position of key points. The algorithm introduces online difficult sample mining to ensure the learning ability of the model and solves the problem of sample imbalance.

Intersection over union (IOU) is used to evaluate the positioning accuracy of the frame in MTCNN. Bounding box regression was used to fine-tune the window. Non-maximum Suppression (NMS) was used to identify accurate rectangular frames of faces.

In the realization of face classification, determination of the position of the face frame, and localization of the face feature points, MTCNN uses cross entropy loss to judge whether it is a face, as shown in Equation (1). The Euclidean distance is used to calculate the regression loss in the box as shown in Equation (2) and the regression loss determined by the key points as shown in Equation (3), and the total loss function as shown in Equation (4).

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

In Equation $y_i^{det} \in \{0,1\}$, y_i^{det} Represents the actual label of the sample, p_i Represents the probability that the sample is a face.

$$L_i^{box} = \left\| \hat{y}_i^{box} - y_i^{box} \right\|_2^2 \quad (2)$$

In Equation $\hat{y}_i^{box} \in R^4$, $\hat{y}_i^{box}$ is the location of the box predicted by the network, y_i^{box} is the actual location of the target.

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

In Equation $\hat{y}_i^{landmark} \in R^{10}$, $\hat{y}_i^{landmark}$ is the face key point position predicted by the network, $y_i^{landmark}$ represents the actual face key point position.

$$min \sum_{i=1}^{N} \sum_{j \in \{det, box, lanmark\}} \alpha_j \beta_i^j L_i^j \quad (4)$$

In Equation $\beta_i^j \in \{0,1\}$, α_j represents the weight.

3. Improved method

The paper is based on the principle of MTCNN face detection, the image is firstly preprocessed for the definition of face, so that the image face is clearer and its application features are more prominent and easier to detect. Inception module is added into PNet, RNet and ONet to improve the learning performance of the network, which not only retains the advantages of the original network structure but also further improves the efficiency and accuracy of face detection algorithm. The experiment was carried out in the framework of TensorFlow based on Python and verified by training on Wider Face and CelebA datasets. The structure diagram of the improved network model is shown in Figure 1.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0067

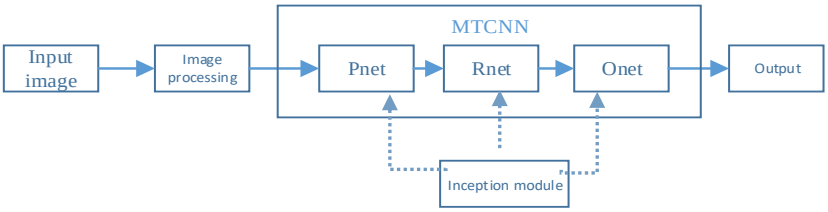

Fig. 1 Schematic diagram of improved MTCNN network structure

3.1 Image preprocessing module

In actual operation, the face detection images are greatly influenced by external factors, SRGAN [5] (Super - Resolution Using a Generative Adversarial Network), the purpose of the super-resolution restoration (Super -Resolution) is to convert the low pixel image high pixel image, SRGAN use against learning method, combining with the pixel mean square error, VGG [13] high-dimensional feature mean square error and against losses to training depth convolution Network to realize the super-resolution restoration. The network structure of SRGAN is shown in Figure 2. The network input does not introduce any random variables, but directly input the picture and then output the picture, its structure is similar to the autoencoder. The generator first encodes the input image into high-dimensional features, then processes the features through the Residual Network [9], and finally decodes the restored high-pixel image. Fig. 3 and Fig. 4 are the schematic diagram of comparison before and after image clearness processing.

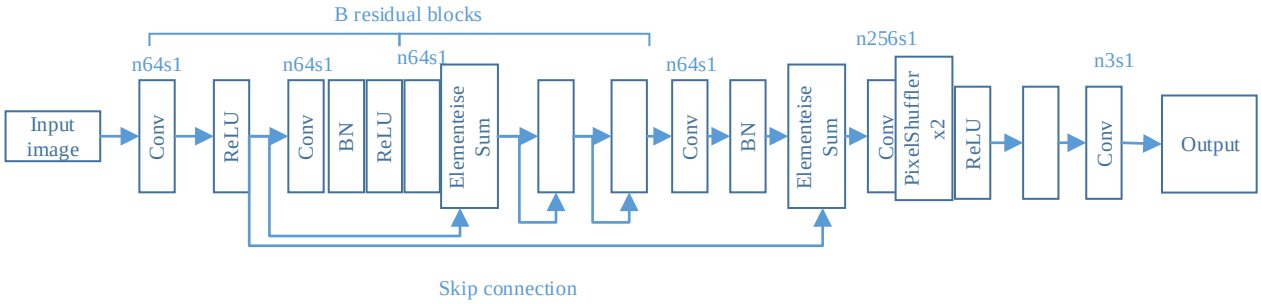

Fig. 2 Schematic diagram of SRGAN image processing network structure

Fig. 3 Before image processing

Fig. 4 After image processing

3.2 Network model optimization module

3.2.1 Inception model

The function of the Inception module is to provide a method for parallel processing of multiple feature graphs. For the feature map generated by the previous layer, there are often many choices: you can add a pooling layer to the back, or you can add another convolution layer, and the size of the convolution layer can also be different. Confronted with multiple options, these feature diagrams can

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0067

be processed simultaneously through the Inception module, which has multiple channels corresponding to different structural choices, and the output of each channel is concatenated and sent to the next layer. The Inception module is schematically shown in Figure 5.

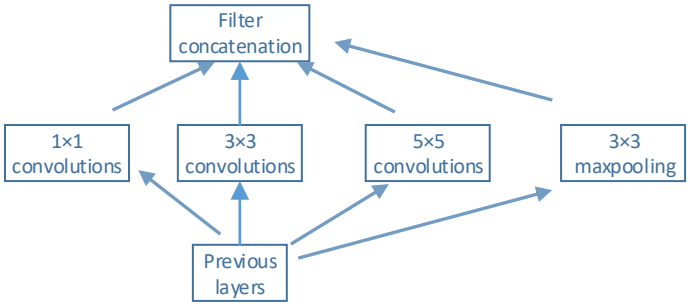

Fig. 5 Schematic diagram of the Inception module

The module extracts more features by carrying out three different convolution (1×1,3×3,5×5) on the feature map. Different convolution checks have different learning effects on faces at different locations. Using different convolution kernels to learn is equivalent to integrating different resolutions, which can better achieve ideal learning effects. The feature graphs extracted after convolution kernel convolution and the results of pooling layer are aggregated as output, but the rapid increase of parameters in this structure will result in a great amount of computation. In the improved scheme, a 1×1convolution is added before the3×3,5×5 convolution and after the pooling layer to reduce dimension. It not only extracts a lot of features but also reduces the computation. The neuron of 1×1 convolution kernel deepens the dimension of the feature map without changing much information of the original image, and introduces nonlinear factors into the generated feature map.

This paper adopts the InceptionV2 [15] module. The Inception module adds Batch Normalization operations [12] to the Inception module, which not only improves the network generalization performance, but also makes it easier and more stable to train network models and speeds up the rate of model convergence. The Inception V2 module introduced adds a 1×1 convolution kernel in front of the 3×3 convolution kernel and the 3×3 pooling layer in the original MTCNN network structure, and optimizes the original 3×3 convolution kernel to 1×3 and 3×1 to improve the learning rate of the network. The optimized network structure is shown in Figure 6.

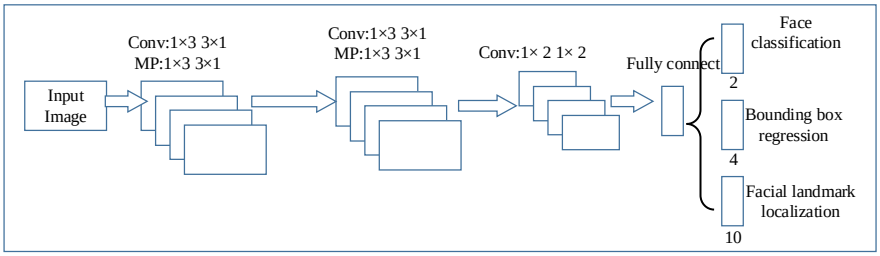

Fig. 6 Optimized MTCNN network structure diagram

3.2.2 Activation function optimization

In this paper, Swish[10]is used to activate the function in the hidden layers of PNet, RNet and ONet. The function expression is shown in Equation (5), and its derivative is shown in Equation (6).

The paper gives when $\beta = 1$ the image of Swish activation function is shown in Figure 7 and its derivative function is shown in Figure 8.

$$f(x) = x \cdot \sigma(\beta x),\ \sigma(z) = \frac{1}{1+e^{-z}} \tag{5}$$

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0067

$$f'(x) = \sigma(\beta x) + \beta x \cdot \sigma(\beta x)(1 - \sigma(\beta x)) = \sigma(\beta x) + \beta x \cdot \sigma(\beta x) - \beta x \cdot \sigma(\beta x)^2$$
$$= \beta x \cdot \sigma(x) + \sigma(\beta x)(1 - \beta x \cdot \sigma(\beta x)) = \beta f(x) + \sigma(\beta x)(1 - \beta f(x)) \quad (6)$$

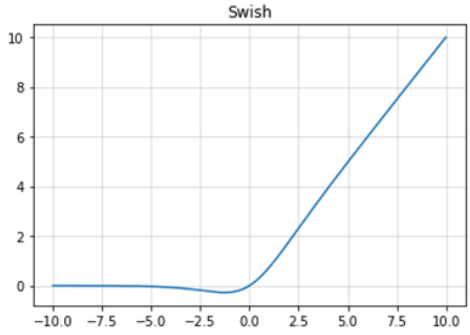

Fig. 7 Swish function image when $\beta = 1$

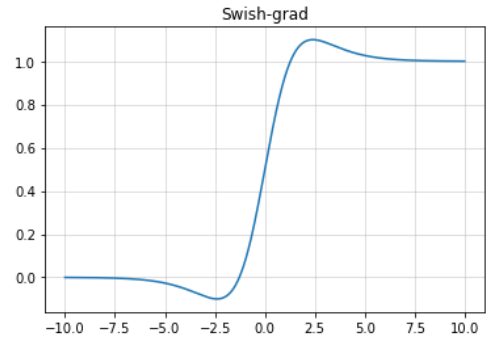

Fig. 8 Swish derivative function image when $\beta = 1$

When $\beta = 0$, $f(x) = \frac{x}{2}$; when $\beta \to \infty$, $\sigma(\beta x)$ is close to function $0 - 1$. In this case, the Swish function is the ReLU [14] function. That is, the Swish function is a smooth function between the linear function and the ReLU function. It improves the disadvantages of reducing the effective capacity of the model due to the sparse processing forced by ReLU, shielding some features and preventing the model from learning effective features, When $x \leq 0$ the disadvantages of negative gradient being set to zero due to the time gradient of ReLU being 0 and the possibility that this neuron will never be activated by any data again [10]. Among them β is a parameter that can be trained. The experiment proves that the function has obvious advantages in the network model and the effect is better.

4. Experiment and Analysis

Experimental environment: build the network model on TensorFlow framework based on Python 3.6.

Server configuration: operating system Windows 10 64-bit; CPU i5-3210MB @ 2.50GHz; Memory 8 GB; Nvidia GeForce GT 635M (2 GB).

Experimental data: In this paper, 457604 positive samples, 1129342 partial FACE samples and 10,00096 non-face samples were obtained from the Wider Face dataset (total number of images is 12880). Crops face tag data from CelebA dataset. Online difficult sample mining was adopted, and the top 70% with large loss was taken as the difficult sample.

Parameters setting: In the training, in order to improve the accuracy, the initial learning rate was set to 0.001, and the learning rate was reduced by 10 times in 3 epoch, 7epoch and 10epoch, respectively. According to the FACE size distribution of Wider Face data set, the minimum FACE size is set to 20, which can effectively reduce the generated pyramid level and improve the training speed while ensuring accuracy. Thresholds of the three networks were set at 0.6, 0.7 and 0.7 respectively, which not only improved the efficiency of candidate box screening but also ensured a high face detection rate. The number of iterations is set to 30,22,22; Batch_size is set to 384.The batch that generates hard_example is set to 2048,256,16, respectively.

In the experiment, PNet and RNet parameters of the loss function were set as α_{det} =1, α_{box}=0.5, $\alpha_{landmark}$=0.5; ONet Parameter set to α_{det} =1, α_{box}=0.5, $\alpha_{landmark}$=1; 0< IoU <0.3 mean it is not a human face, 0.4< IoU <0.65 mean it is a part of a human face, 0.65< IoU <1 mean it is a human face. Comparison data of experimental accuracy, face classification error, face key point position error and total error are shown in Table 1. The blacken number is the final accuracy obtained after network optimization.

Visualize the training process of each layer of the network as shown in Fig. 9 to 14. This paper lists the training comparison and visualization process of accuracy and total error of each layer of the

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0067

network. The red curve represents the training process of the unoptimized network, while the blue curve represents the training process of the improved network.

Table 1. Comparison of experimental results before and after optimization

network	accuracy		cls loss	
	before	after	before	after
PNet	0.943061	0.951493	0.262198	0.184743
RNet	0.953125	0.96875	0.194791	0.148225
ONet	0.964844	0.988281	0.170924	0.081185

network	bbox loss		Landmark loss		Total Loss	
	before	after	before	after	before	after
PNet	0.068332	0.071914	0.045918	0.044924	0.335969	0.260096
RNet	0.090558	0.081788	0.046603	0.029594	0.288167	0.229419
ONet	0.056091	0.044497	0.030206	0.012998	0.278863	0.162242

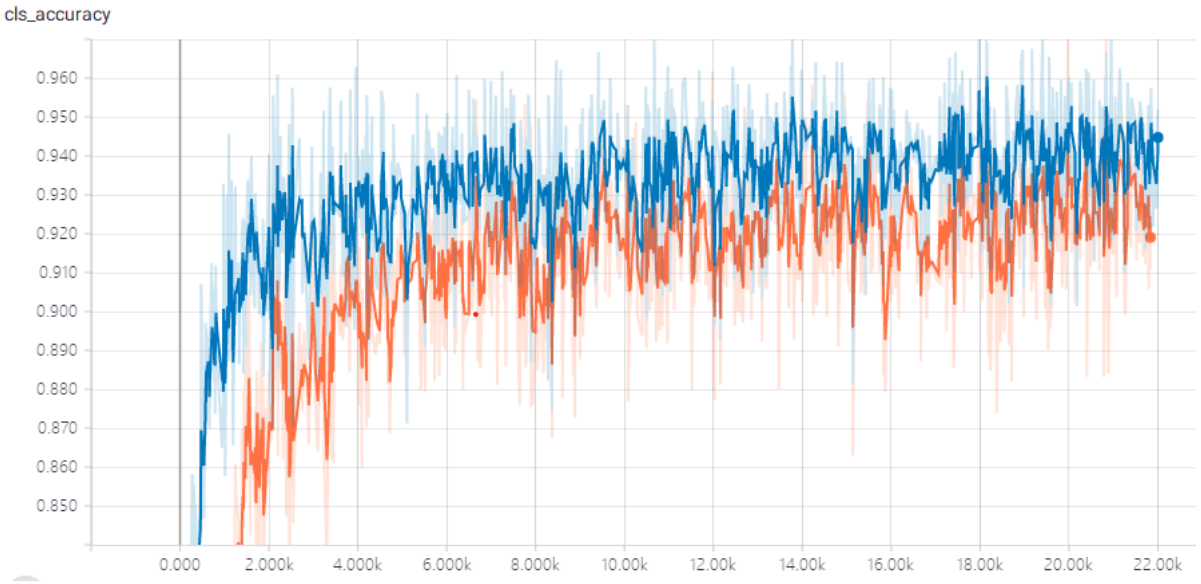

Fig. 9 Comparison of PNet accuracy optimization

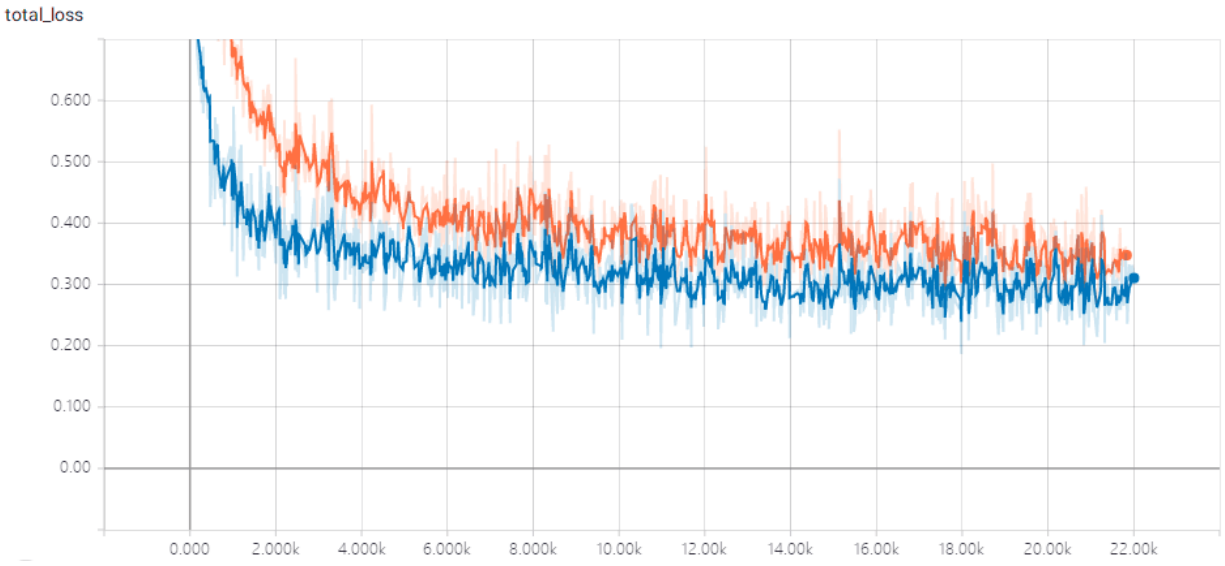

Fig. 10 Comparison of PNet total error optimization

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0067

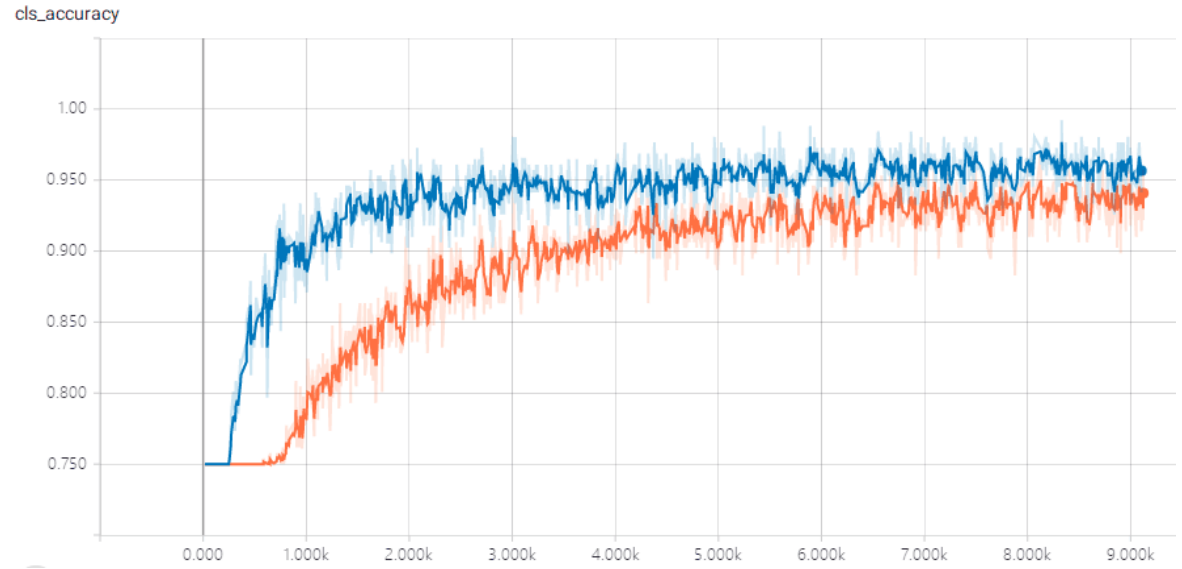

Fig. 11 Comparison of RNet accuracy optimization

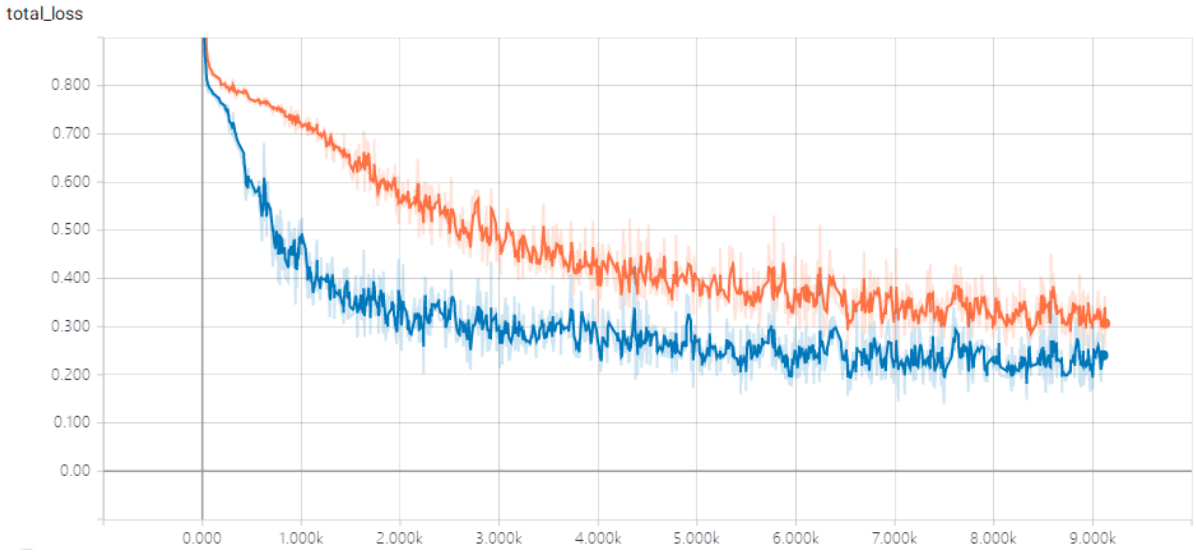

Fig. 12 Comparison of RNet total error optimization

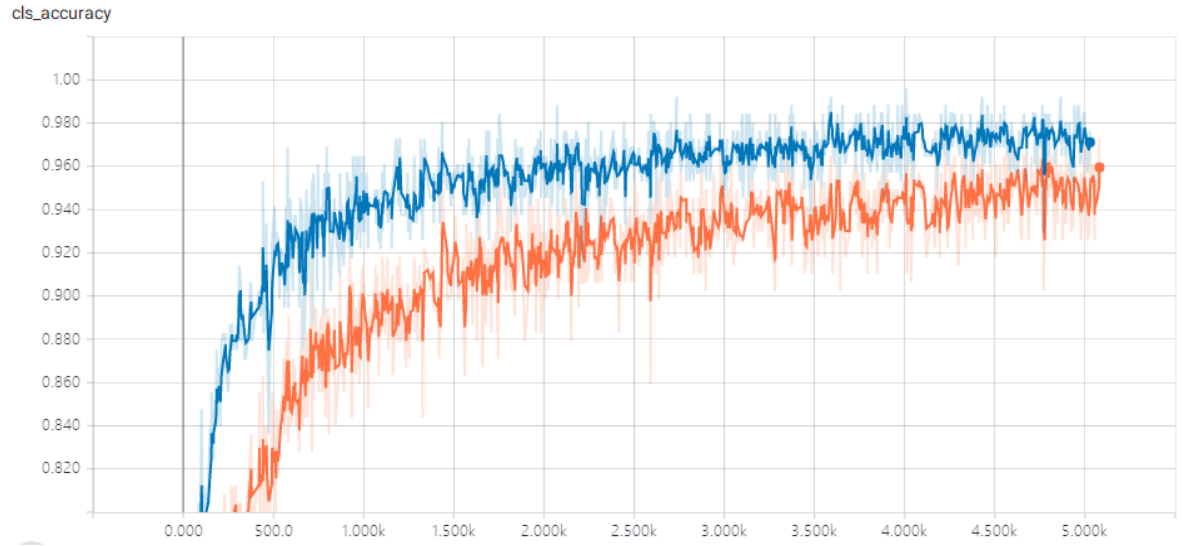

Fig. 13 Comparison of ONet accuracy optimization

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0067

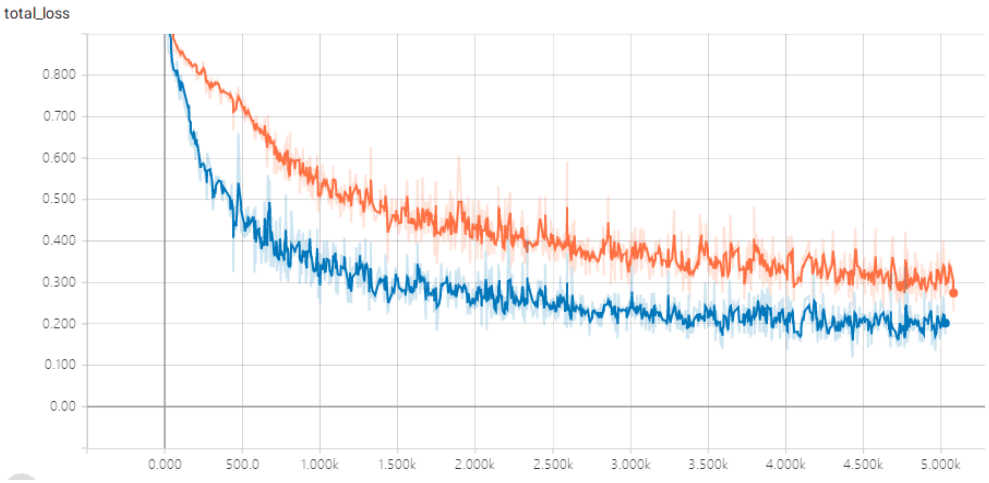

Fig. 14 Comparison of total error optimization results of Onet

The experimental results from Figure 9 to Figure 14 and Table 1 show that the optimization method in this paper improves the accuracy and efficiency of face detection against a complex background, with an accuracy up to 98.8%, and can accurately locate the key points of the face with an error reduced to 1.2998%.Compared with the unoptimized scheme, the position error of face key points is reduced by 60.0%;The box regression loss is reduced to 4.4%, and the box regression loss is reduced by 20.7% compared with the unoptimized scheme. Compared with the unoptimized scheme, the total error is reduced by 41.8%. Comparison of test results is shown in Figure 15 and Figure 16

Fig. 15 Effect before optimization

The effect before optimization is shown in Figure 15. It can be concluded from the results that the detection effect of MTCNN is not good under the condition of complex background environment and poor image quality.

Fig. 16 Optimized face detection effect

ISSN: 2414-1895 DOI: 10.6919/ICJE.202108_7(8).0067

The optimized effect is shown in Figure 16. According to the results, the detection effect of the optimized MTCNN algorithm has been improved in a complex background environment, but the detection of small face still needs to be improved.

5. Conclusion

Based on the principle of MTCNN face detection, the image is preprocessed and the model is optimized. The experiment is based on Python in TensorFlow framework and on WiderFace and CelebA datasets for training verification and simulation. The final accuracy of face detection can reach 98.8%. From the experimental results, it can be seen that the improved algorithm has improved the performance of face detection for single face, face dense and complex environment. It shows that the improved method in this paper improves the accuracy and efficiency of face detection in complex background, and better meets the application requirements of face detection in modern society, which has a certain practical value and practical significance.

References

[1] Haitao Zhang, Meng Zhang. SSD target detection algorithm with channel attention mechanism [J]. Computer Engineering, 2020, 46(08):264-270.

[2] Yani Zhu, Xuan Ni, Ye Yao. Face recognition combined with singular value face and attention deep learning [J]. Small Microcomputer System, 2020, 41(08):1763-1767.

[3] Shifeng Zhang, Cheng Chi, Zhen Lei, Stan Z. Li: “RefineFace: Refinement Neural Network for High Performance Face Detection”, 2019; [arXiv:1909.04376].

[4] Xu Tang, Daniel K. Du, Zeqiang He, Jingtuo Liu: “PyramidBox: A Context-assisted Single Shot Face Detector”, 2018, ECCV2018; [arXiv:1803.07737].

[5] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, Wenzhe Shi: “Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network”, 2016; [arXiv:1609.04802].

[6] ZHANG K P, ZHANG Z P, LI Z F, et al. Joint face detection and align-ment using multi-task cascaded convolutional networks [J]. IEEE Sig-nal Processing Letters, 2016.23 (10): 1499-1503.

[7] Shifeng Zhang, Xiangyu Zhu, Zhen Lei, Hailin Shi, Xiaobo Wang, Stan Z. Li: “FaceBoxes: A CPU Real-time Face Detector with High Accuracy”, 2017; [arXiv:1708.05234].

[8] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, 2012.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun: “Deep Residual Learning for Image Recognition”, arXiv: 1512.03385,2015.

[10] Prajit Ramachandran, Barret Zoph, Quoc V. Le: “Searching for Activation Functions”, 2017.

[11] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna: “Rethinking the Inception Architecture for Computer Vision”, arXiv: 1512.00567, 2015.

[12] Sergey Ioffe, Christian Szegedy: “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”, arXiv: 1502.03167v3, 2015.

[13] Karen Simonyan, Andrew Zisserman: Very Deep Convolutional Networks for Large-Scale Image Recognition, arXiv: 1409.1556,2014.

[14] Xavier Glorot, Antoine Bordes, Yoshua Bengio, “Deep Sparse Rectifier Neural Networks”, 2011.

[15] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna: “Rethinking the Inception Architecture for Computer Vision”, arXiv: 1512.00567, 2015.

[16] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich: “Going Deeper with Convolutions”, arXiv: 1409.4842, 2014.