electronics

Article

Scale-Adaptive KCF Mixed with Deep Feature for Pedestrian Tracking

Yang Zhou [1,2], Wenzhu Yang [1,2,*] and Yuan Shen [1]

[1] School of Cyber Security and Computer, Hebei University, Baoding City 071002, China; zy1020546839@hotmail.com (Y.Z.); jiqishijue306@hotmail.com (Y.S.)
[2] Hebei Machine Vision Engineering Research Center, Hebei University, Baoding City 071002, China
* Correspondence: wenzhuyang@hbu.edu.cn

Abstract: Pedestrian tracking is an important research content in the field of computer vision. Tracking is achieved by predicting the position of a specific pedestrian in each frame of a video. Pedestrian tracking methods include neural network-based methods and traditional template matching-based methods, such as the SiamRPN (Siamese region proposal network), the DASiamRPN (distractor-aware SiamRPN), and the KCF (kernel correlation filter). The KCF algorithm has no scale-adaptive capability and cannot effectively solve the occlusion problem, and because of many defects of the HOG (histogram of oriented gradient) feature that the KCF uses, the tracking target is easy to lose. For those defects of the KCF algorithm, an improved KCF model, the SKCFMDF (scale-adaptive KCF mixed with deep feature) algorithm was designed. By introducing deep features extracted by a newly designed neural network and by introducing the YOLOv3 (you only look once version 3) object detection algorithm, which was also improved for more accurate detection, the model was able to achieve scale adaptation and to effectively solve the problem of occlusion and defects of the HOG feature. Compared with the original KCF, the success rate of pedestrian tracking under complex conditions was increased by 36%. Compared with the mainstream SiamRPN and DASiamRPN models, it was still able to achieve a small improvement.

Keywords: pedestrian tracking; improved KCF; deep features; object detection

check for updates

Citation: Zhou, Y.; Yang, W.; Shen, Y. Scale-Adaptive KCF Mixed with Deep Feature for Pedestrian Tracking. *Electronics* **2021**, *10*, 536. https://doi.org/10.3390/electronics10050536

Academic Editor: Arturo de la Escalera

Received: 19 January 2021
Accepted: 21 February 2021
Published: 25 February 2021

1. Introduction

Pedestrian tracking is an important research topic in the field of computer vision, and it has great application values, such as for the use of intelligent monitoring, pedestrian flow observation, and other scenarios. In reality, tracking is achieved by determining the location of a specific pedestrian in each frame of a video. Pedestrian tracking methods include neural network-based methods and traditional template matching-based methods. Regarding the neural network-based method, the mainstream method is the use of a Siamese neural network [1] based on the RPN (region proposal network) [2] for tracking. Bo Li and Junjie Yan proposed the Siamese region proposal network (SiamRPN) [3], which is different from the standard RPN because it extracts candidate area from related feature maps, and then the target appearance information on the template branch is encoded into the RPN feature to distinguish the foreground from the background. However, it is still difficult for the SiamRPN to distinguish between similar objects in an image. Due to a lack of model updates and the suppression of interference objects, Zheng Zhu and Qiang Wang proposed the DASiamRPN (distractor-aware SiamRPN) [4]. There are also pedestrian tracking methods based on template matching. SIFT (scale-invariant feature transform) features [5] can be used to describe the characteristics of pedestrians. Using this feature as a template, the location of pedestrians can be predicted by sliding window matching on the next frame of video. Joao F. Henriques proposed the KCF (kernel correlation filter) algorithm [6] for pedestrian tracking. The KCF algorithm uses pedestrian image information and surrounding background image information to train a target detector to predict the position of a pedestrian in subsequent frames. However, the KCF algorithm has three flaws. One is its scale problem, as the size of the target detector is unchanged all the time. However, in the video, the size of the pedestrian target changes due to its distance from the camera. Thus, the algorithm inevitable tracks the target inaccurately. The second problem is the defect of the HOG (histogram of oriented gradient) feature [7] used by the KCF. The HOG feature uses gradient feature representation, so it is insensitive to pedestrian posture changes and color information, which leads to tracking errors or tracking loss during the tracking process. The third is the occlusion problem. When the pedestrian target is occluded, the detector cannot give the accurate position of the pedestrian target in the next frame of the video.

For the shortcomings of the KCF algorithm, an improved KCF model that incorporates deep features was designed. The neural network framework YOLOv3 (you only look once version 3) [8] used for target recognition is used for pedestrian detection, and then the newly detected image of the pedestrian by YOLOv3 is used as a new template of the KCF to train its target detector so as to solve the scale change problem. When the HOG feature is not capable of distinguishing different pedestrians, the deep feature is integrated to determine the location of the pedestrian target, and the convolutional neural network is used to extract the deep feature of pedestrians for comparison. When the pedestrian is occluded and the KCF target is lost, the convolutional neural network used for extracting the deep features of pedestrians is used to compare the last deep feature before the occlusion with the deep features of all pedestrians recognized by YOLOv3 after the occlusion disappears and to re-determine the location of the pedestrian according to similarity.

YOLOv3 runs very fast, but its non-maximum suppression (NMS) algorithm has caused many correctly predicted bounding boxes to be removed by mistake. We added the retrieval algorithm to recover the person detection box that was mistakenly removed by NMS, and we replaced NMS with Soft-NMS to further improve the accuracy. Experiments on the PASCAL VOC (Pattern Analysis, Statistical Modeling and Computational Learning, Visual Object Classes) dataset showed that YOLOv3, which uses Soft-NMS and the improved retrieval algorithm, improved the accuracy by approximately 3.1% compared to the original algorithm, while the operating speed did not change much.

2. KCF with Deep Feature and Adaptive Scale

With the aim of fixing these three flaws of the KCF algorithm, the SKCFMDF (scale-adaptive KCF mixed with deep feature) algorithm is proposed here. This new algorithm solves the three problems of the KCF algorithm. A schematic diagram is shown in Figure 1.

2.1. KCF Tracking Algorithm

The KCF algorithm is proposed by Joao F. Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista in 2014. The KCF algorithm uses the image of a tracking target to extract the HOG feature, and then it takes the surrounding images as the training sample to train the target detector. After training, the Gaussian kernel function is used to calculate the correlation response between the HOG feature of the tracking target image and the HOG features of the surrounding images (the image with the highest response value is the latest position image of the tracking target in the frame of the image), and then the algorithm uses the image with the highest response in the frame to retrain the target detector. By using the discrete Fourier transform to convert the above process from the time domain to the frequency domain, the calculation can be greatly reduced.

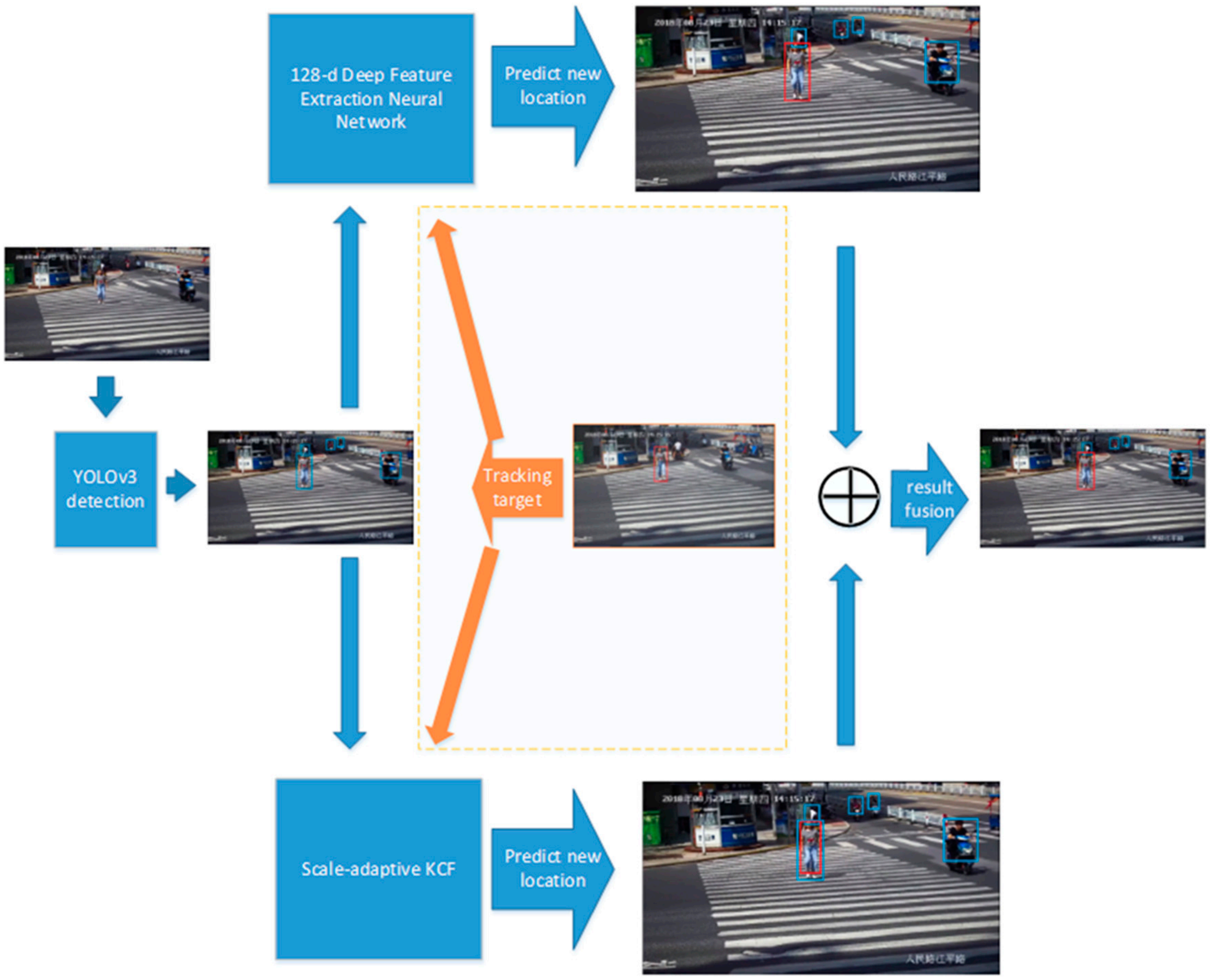


Figure 1. Schematic diagram of the scale-adaptive KCF (kernel correlation filter) mixed with the deep feature (SKCFMDF) algorithm. YOLOv3: you only look once version 3.

2.2. KCF Scale Adaptation

The KCF has a scale problem. In the KCF algorithm, the scale of the extracted image is always the pixel size of the initial target image tracking area, so if the movement of the target causes the distance from the camera to change, the relative scale of the target in the image also changes. If the size of the target bounding box does not change, the extracted features will be incomplete or variable background information will be introduced, thus leading to a failure of tracking; as such, the KCF has problems with scale changes.

Because the magnitude of the target scale change is not too large, fixing the KCF's scale change problem is not difficult. In a new image frame, the target bounding box predicted by the KCF could be combined with the pedestrian bounding box detected by YOLOv3 to obtain a new scale that matches the size of the tracking target. Then, the new target box could be used as the training template of the KCF target detector so that the scale adaptation of the KCF can be realized. The process is shown in Figure 2.

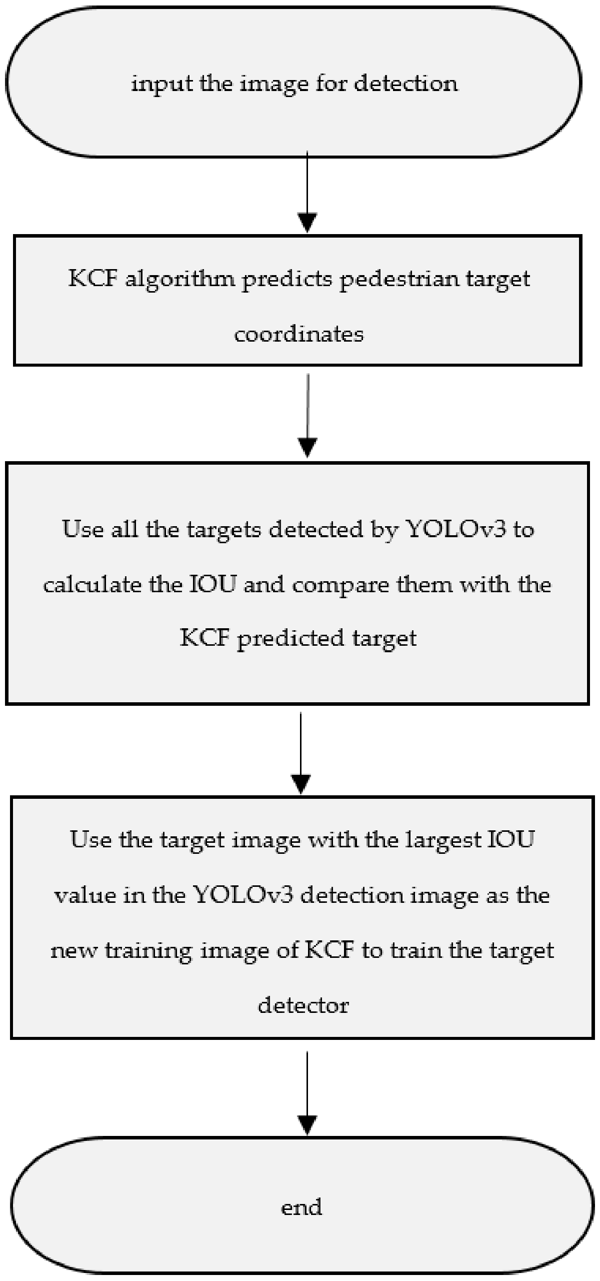

input the image for detection
KCF algorithm predicts pedestrian target coordinates
Use all the targets detected by YOLOv3 to calculate the IOU and compare them with the KCF predicted target
Use the target image with the largest IOU value in the YOLOv3 detection image as the new training image of KCF to train the target detector
end

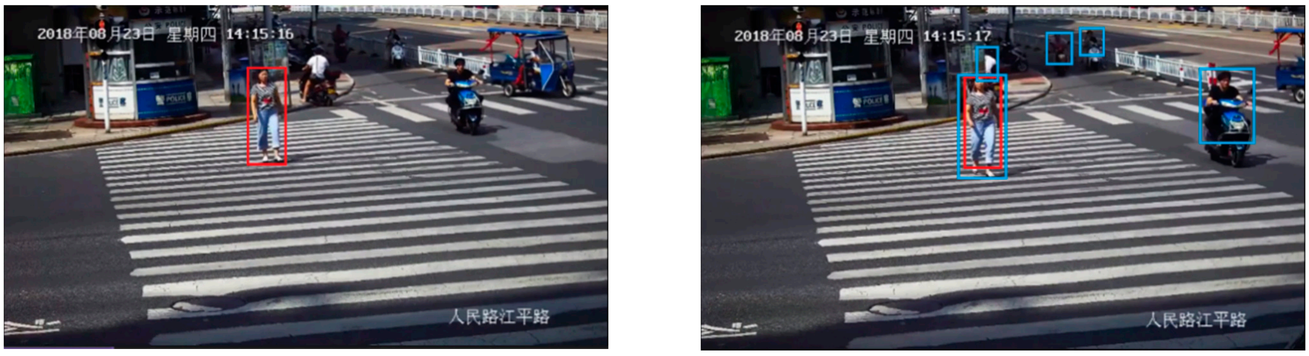
Figure 2. KCF's implementation of the scale adaptation process. IOU: intersection over union.

The pedestrian detection framework we use is YOLOv3. YOLOv3 is a neural network with darknet-53 as a backbone network to extract the characteristics of an image. Because the receptive field of each layer in the network is different, YOLOv3 extract targets of different scales in different layers; a total of three scales were identified, and the MS-COCO (Microsoft Common Objects in Context) dataset [9] was used to train the network.

For the smallest detection scale, YOLOv3 divides the image into 52 $\times$ 52 grids because the 52 $\times$ 52 size is suitable for commonly seen small-scale objects and each grid is used as an anchor point. The input image for YOLOv3 is compressed to a fixed size, so the grid size is independent from the input image. Three bounding boxes of different shapes are identified at the anchor point. For the middle scale, the model divides the image into 26 $\times$ 26 grids as anchor points, because the grid size of 26 $\times$ 26 is suitable for commonly seen middle-scale objects. For the largest scale, the model divides the image into 13 $\times$ 13 grids as anchor points because the grid size of 13 $\times$ 13 is suitable for commonly seen large-scale objects. In the smallest-scale recognition, the feature maps used in the middle-scale are combined with the feature maps of the largest-scale, thereby increasing the accuracy of the smallest-scale recognition. In this way, the scale change of target recognition can be realized to recognize from three scales in similar way to a feature pyramid network [10].

The determination of the new training template of the KCF is realized by the intersection ratio of the KCF prediction box and the detection box of YOLOv3.

As shown in the image in Figure 3a, in a pedestrian video, the pedestrian bounding box detected by the KCF is marked with a red box; in the next frame, which is the image in Figure 3b, the KCF predicting box is still marked in red because the woman has walked closer to the camera. The image scale of the pedestrian relative to the image has increased, but the scale of the KCF red box has not changed, resulting in the box not completely covering the target. As the woman gets closer, the scale of the target continues to relatively increase, and continuing to use the KCF would cause a loss of tracking. At this time, the YOLOv3 model is introduced to detect the coordinates of people in the frame, which are marked by the blue boxes. Since multiple pedestrians appear in the image, by calculating the value of the intersection over union (IOU) between the blue box and the red box, the blue box with the largest IOU can be selected as the new target to train the KCF target detector.


2018年08月23日 星期四 14:15:16
人民路江平路
2018年08月23日 星期四 14:15:17
人民路江平路

(**a**) (**b**)

Figure 3. Schematic diagram of variable kcf scale. (**a**) Pedestrian tracked with the KCF; (**b**) KCF scale problem corrected by YOLOv3.

2.3. *Pedestrian Feature Extraction Based on Convolutional Neural Network*

There are flaws for the HOG feature used by the KCF. The HOG feature uses gradient feature representation, so it is not sensitive to pedestrian posture changes and color information, and the gradient feature of HOG is also sensitive to noise, which leads to tracking errors or tracking losses during the tracking process.

In order to make up for these HOG feature defects, a neural network used to learn and extract the deep features of pedestrian targets was designed and trained.

The network is a nine-layer convolutional neural network. The network structure is shown in Table 1. Since the entire network is small, the deep features of the image can be quickly extracted. The network uses a convolution kernel with a size of 3×3 and a stride size of 1, resizes the pedestrian image recognized by YOLOv3 to a pixel size of 128×64, and uses an RGB three-channel image with this fixed size as the input image of the entire network, so the sizes of the feature maps are independent from the original picture size. The network uses the Adam descent algorithm [11], and each layer uses L_2 regularization and batch normalization. Additionally, each layer uses an ELU (exponential linear unit) as the activation function, so the convergence speed is fast and the training speed can be accelerated.

Table 1. The neural network structure of deep feature extraction.

Layer Name	Output Size	Convolution Kernel	Stride
Convolution layer	32 × 128 × 64	3 × 3	1
Convolution layer	64 × 128 × 64	3 × 3	1
Max pooling layer	64 × 64 × 32	2 × 2	2
Convolution layer	64 × 64 × 32	3 × 3	1
Convolution layer	128 × 64 × 32	3 × 3	1
Max pooling layer	128 × 32 × 16	2 × 2	2
Convolution layer	128 × 32 × 16	3 × 3	1
Max pooling layer	128 × 8 × 4	4 × 4	4
FC (Full Convolution) layer	128		

The number of layers designed by the neural network is derived from the calculation formula of the receptive field. The neural network calculations are used to extract pedestrian features, mainly distinctive texture features. Since the size of the input pedestrian image is 128 × 64 pixels, by observing the difference of the people, texture features of with an approximate size of 20 × 20 pixels can obviously distinguish a pedestrian from another. To distinguish people in different dresses, we calculated a network with eight layers according to the required receptive field size and receptive field calculation formula, and a fully connected layer was also added to calculate a 128-dimensional feature vector.

The receptive field formula is as follows.

$$L_k = L_{k-1} + ((F_k - 1) \times \prod_{i=1}^{k-1} S_i) \tag{1}$$

where L_{k-1} is the receptive field size of the k-1th layer, F_k is the current convolution kernel size, and S_i is the stride of the i-th layer.

According to the formula of the receptive field, the size of the receptive field in the seventh layer is 24 × 24 pixels. After the seventh convolutional layer is calculated, a feature map size of 32 × 16 is obtained, and then a 4 × 4 pooling layer is used to further remove redundant information and reduce the amount of calculation. The feature map is changed to the size of 8 × 4. A fully connected layer is used to extract a 128-dimensional feature vector as the feature representation of the pedestrian target.

2.4. Pedestrian Tracking Based on Fusion Metrics

When there is a long-term occlusion problem, the KCF algorithm loses tracked pedestrian targets. At this time, if a comparison of the deep features is combined for tracking, the deep features of the missing pedestrian target image can be retained. In the next frame, the deep features of the newly emerging pedestrian target are compared. If the similarity is greater than the set threshold, it can be considered that the occluded target has reappeared and can be tracked again. For example, the man in Figure 4a with a red bounding box is the tracking target, and his deep feature of 128-dimensional vector is extracted by the network and restored; however, his slow running causes him to be gradually left by the camera view, as seen in Figure 4b. In Figure 4c, he is completely out of the camera view and the KCF totally loses track, but SKCFMDF compares the 128-dimensional vector of the target with other people and finds that no one matches. Thus, in Figure 4c, the target is lost and no red bounding box is labelled. In Figure 4d, the target reappears, and SKCFMDF compares the restored target vector with the newly appeared man to find the target again; as such, the corresponding bounding box is labelled with red. Because the deep feature of 128-dimensional vector of the target is restored in the model forever, there is no time limit for the occluded target to be retracked. Therefore, SKCFMDF, which combines the scale-adaptive KCF algorithm and the deep feature contrast tracking method, can solve the problem of the KCF losing pedestrian targets under occlusion.

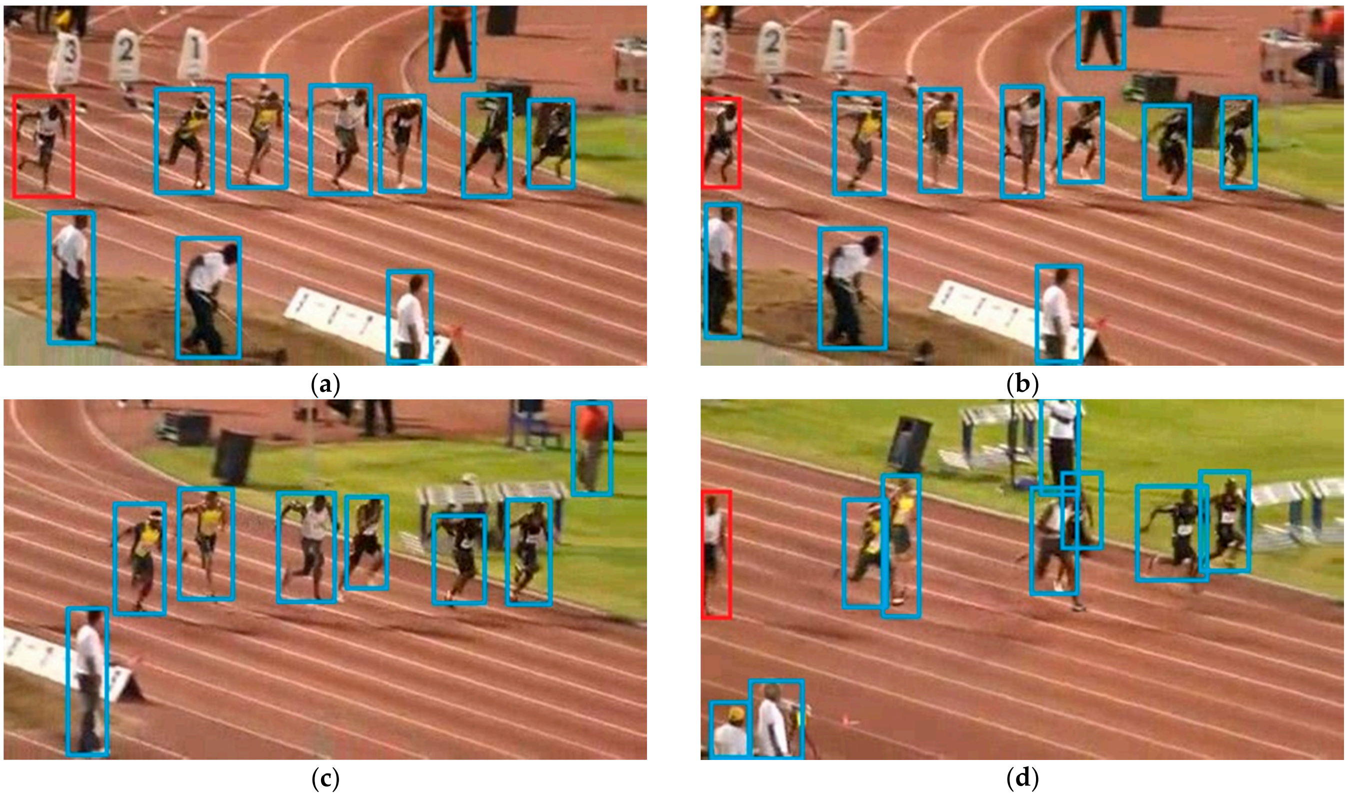


Figure 4. SCKFMDF solves the occlusion problem by comparing the deep features of 128-dimensional vectors of different people. (**a**) Tracking target is labelled with red bounding box; (**b**) tracking target is gradually left by the camera view for his slow running; (**c**) tracking target is completely lost in the camera view; (**d**) tracking target reappears and is found by SCKFMDF and labelled with red bounding box.

After the neural network for calculating the deep features is obtained, the similarity calculated by the deep feature of the pedestrian image and the KCF prediction confidence are merged, and the position of a pedestrian can be tracked more accurately. The formula of the fusion measurement method is as follows.

$$\mathrm{C} = \lambda \mathrm{C}_{\mathrm{kcf}} + (1-\lambda)\mathrm{D}_{\mathrm{network}} \tag{2}$$

where $\mathrm{C}_{\mathrm{kcf}}$ is the recognition confidence of the KCF, and $\mathrm{D}_{\mathrm{network}}$ is the confidence of recognition by the neural network. When the camera shakes, λ can be set to 0.

3. Improved YOLOv3 Algorithm for Pedestrian Recognition

3.1. Use Soft-NMS Algorithm to Filter Incorrectly Predicted Detection Boxes

Soft-NMS was proposed by Boldla et al., and it is like the traditional one except that it does not remove the bounding boxes with high overlap values all at once. It decays the confidence score of the bounding box whose overlap value is higher than or equal to the threshold.

The removal step in traditional NMS could be described as follows,

$$\begin{cases} S_i = S_i,\ iou(M,\ b_i) < N_t, \\ S_i = 0,\ iou(M,\ b_i) \geq N_t \end{cases} \tag{3}$$

where $iou(M,\ b_i)$ is the overlap value between the bounding box with max confidence score and the rest of the boxes.

The equation sets S_i the score of bounding box i by comparing the iou value and the threshold value, N_t. It is hard to make the judgement just by the threshold to determine whether the bounding boxes should be removed. Soft-NMS decays the score of the bounding box when the box's overlap value is higher than or equal to the threshold because according to the principle of the YOLOv3′s convolutional neural network, the higher the overlap value of the box, the more likely the bounding box is a duplicate predicted one that is false positive. When some boxes' overlap values are a little bit higher than the threshold, they need to be removed. However, they have high confidence scores, which means they are more likely to be correctly predicted boxes and should be kept. Thus, Soft-NMS keeps the bounding boxes with overlap values higher than the threshold but not high enough to almost fully overlap with the bounding box M, and these bounding boxes have high enough confidence scores that they are kept after decaying the confidence scores. The bounding boxes that almost fully overlap with the bounding box M are removed because they are likely to be duplicated predicted bounding boxes. The removal standard of Soft-NMS is defined as

$$\begin{cases} S_i = S_i, & iou(M,\ b_i) < N_t \\ S_i = S(1 - iou(M, b_i)), & iou(M,\ b_i) \geq N_t \end{cases} \tag{4}$$

The two equations above work as linear functions to keep or decay scores of detected boxes. Thus, the bounding boxes far away from the bounding box M will be less affected or not affected at all according to the equations. Additionally, if bounding boxes are very close to bounding box M or mostly covered by the bounding box M, their confidence scores will be greatly decreased. Finally, after all the confidence scores of bounding boxes are decayed, another threshold to remove incorrectly predicted bounding boxes is used. Decaying the confidence scores of these bounding boxes does not remove the duplicate ones, so the threshold needs to be set to filter the bounding boxes with low confidence scores after decaying.

Compared to traditional NMS, Soft-NMS does not add any more calculation for YOLOv3 human detection framework. The computational complexity of Soft-NMS is $O(N^2)$, where N is the number of predicted bounding boxes produced by the convolutional neural network, which is the same as traditional NMS. Each bounding box needs to have its overlap value calculated, with the bounding box having max confidence score, so the computational complexity of Soft-NMS is $O(N^2)$.

Soft-NMS is a quite small component for the YOLOv3 human detection framework. It does not need to retrain the convolutional neural network of YOLOv3, so it does not cost too much work to be integrated into the YOLOv3 framework.

3.2. Use the Retrieval Algorithm to Recover the Detected Bounding Box Missed by Soft-NMS

Because Soft-NMS also judges whether to remove the detected bounding box based on the overlap value, bounding boxes that are incorrectly removed and missed by the algorithm must be still detected. In this regard, these error-removed detected bounding box can be retrieved through the retrieval algorithm.

In the retrieval algorithm, the FHOG (Felzenszwalb histogram of oriented gradient) features of face images are extracted from the face dataset, and these feature data are used to train an SVM (Support Vector Machine). Along with using NMS and a sliding window to crop an image, faces can be detected in images. Additionally, in the improved YOLOv3, after having detected all human instances, all the faces in the image need to be detected, as seen in Figure 5. Ideally, each detected face bounding box is fully inside the human bounding box. Thus, if there a detected face bounding box is outside the human bounding box or overlapping with a human bounding box, it must belong to an incorrectly removed human bounding box. Thus, all human bounding boxes removed by Soft-NMS are searched again to find one that fully covers the face bounding box and has the highest confidence. In Figure 5, it can be seen that the green bounding boxes detected faces that were not fully inside the human bounding boxes, so human bounding boxes missed by

Soft-NMS had to exist. Additionally, the missed human bounding boxes are recovered by the retrieval algorithm and bounded in blue.

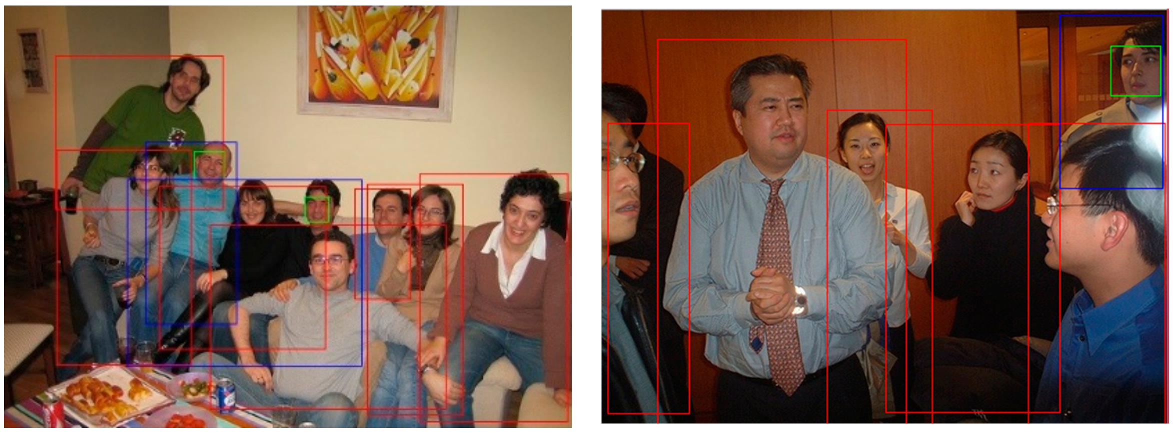

(**a**) (**b**)

Figure 5. The person is detected and the missed detected bounding boxes are restored. (**a**) Face boxes overlapping any human box are restored to their corresponding non-maximum suppression (NMS)-missed-out boxes; (**b**) face boxes not within any human box are restored to their corresponding NMS-missed-out box.

Here, we discuss how to judge whether a face-detected bounding box is completely in a person-detected bounding box. In the following equations, X_1 and Y_1 are the coordinates of the upper right corner of the face-detected bounding box, X_2 and Y_2 are the coordinates of the lower left corner of the face-detected bounding box, M_1 and N_1 are the coordinates of the upper right corner of the person-detected bounding box, and M_2 and N_2 are the coordinates of the lower left corner of the person-detected bounding box. If the coordinates of these points meet the following conditions, the face-detected bounding box is completely inside the person-detected bounding box.

$$M_1 - X_1 > 0,\ Y_1 - N_1 > 0 \tag{5}$$

$$X_2 - M_2 > 0,\ N_2 - Y_2 > 0 \tag{6}$$

If there is a face-detected bounding box outside or overlapping with the person-detected-bounding box, the model searches for all the original detected bounding boxes that have not been deleted by the Soft-NMS algorithm. The model calculates which one of them completely covers the face-detected bounding box and finally finds the person-detected bounding box with the highest confidence. Then, the person-detected bounding box is retrieved and restored, so the accuracy of person detection is improved. Figure 6 shows the flow chart of the retrieval algorithm.

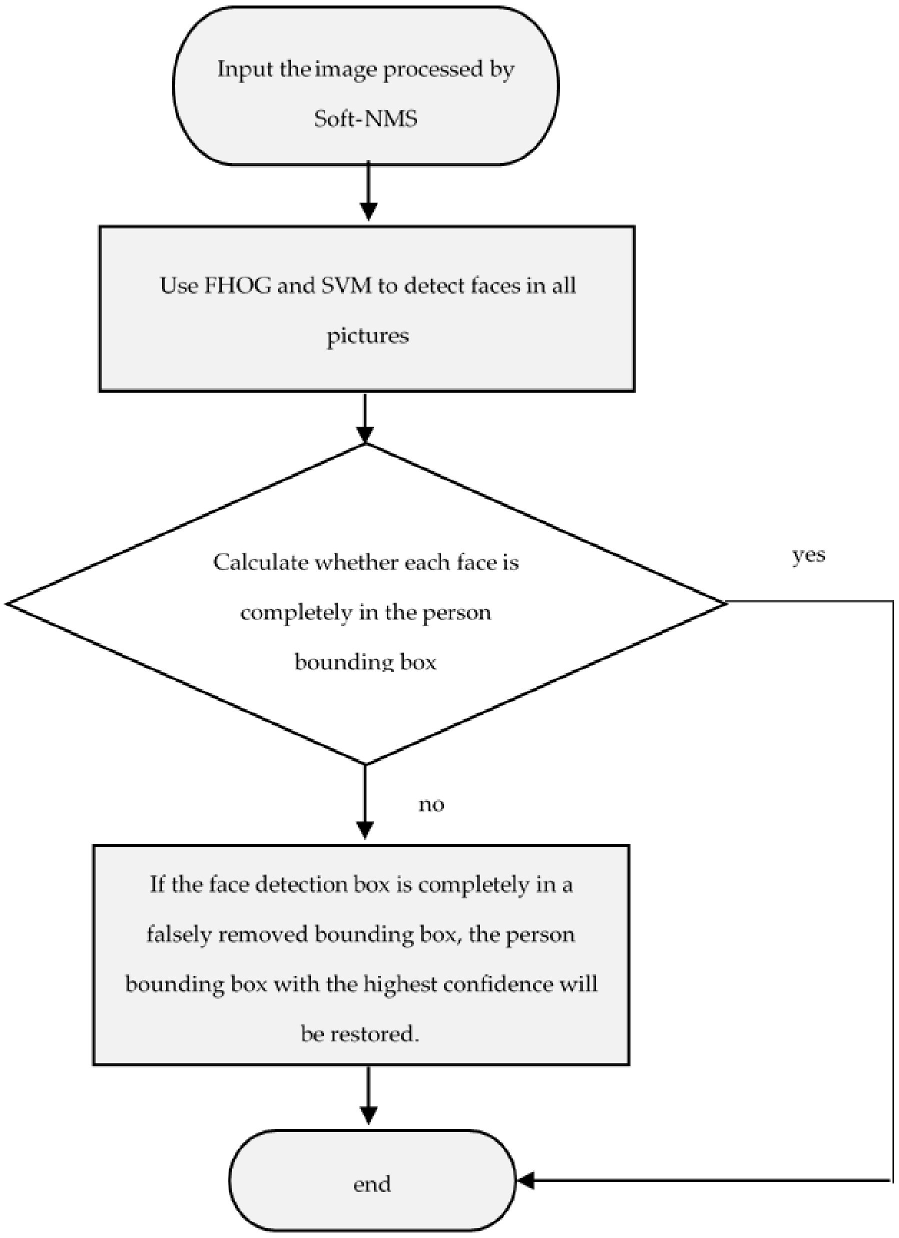

Input the image processed by Soft-NMS
Use FHOG and SVM to detect faces in all pictures
Calculate whether each face is completely in the person bounding box
yes
no
If the face detection box is completely in a falsely removed bounding box, the person bounding box with the highest confidence will be restored.
end

Figure 6. Flow chart of the retrieval algorithm. FHOG: Felzenszwalb histogram of oriented gradient; SVM: support vector machine.

4. Experiments

4.1. Train the Deep Feature Extraction Network

The market-1501 dataset [12] was used to train a neural network for the feature extraction of pedestrian images. The dataset contained 32,668 images of 1501 pedestrians. Each pedestrian was captured by at least two cameras, and each camera took multiple pictures of pedestrians. Since the neural network calculates a feature vector, the training uses triplet loss [13] as the training loss function.

$$\mathrm{L} = \max(\mathrm{d}(a, p) - \mathrm{d}(a, n) + \mathrm{margin}, 0) \tag{7}$$

where a, p, and n are three pedestrian images as training data: a is a training image of a person, p is a sample image of the same person as a, and n is a sample image of a person different from a. Additionally, $\mathrm{d}(a, p)$ is the Euclidean distance of the pedestrian images of *a* and *p* after the neural network calculates the deep features, and d(*a*, *n*) is the Euclidean distance of the pedestrian images of *a* and *n* after the neural network calculates the deep features.

It can be seen from Figure 7 that the training of the neural network reached 98% accuracy after 30 k iterations.

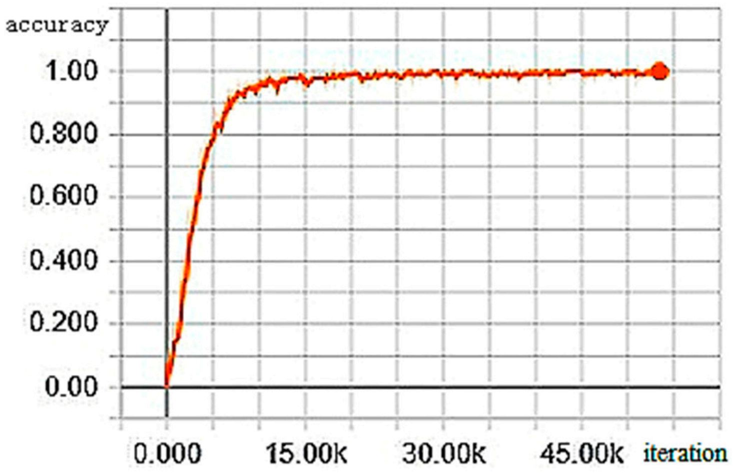


Figure 7. The training accuracy of the pedestrian image feature extraction network.

4.2. Soft-NMS and Retrieval Algorithm to Improve YOLOv3

Regarding the experiment, the used dataset was PASCAL VOC 2007 [14]. The weights used by YOLOv3 are the weights trained by the author of the official website. The dataset used for weight training was MS-COCO. The test part of the PASCAL VOC dataset was used to test the average accuracy of the improved YOLOv3. The PASCAL VOC test section contained about 5000 pictures.

In the experiment, we set the NMS overlap threshold to the default value of 0.3, which was the value found by the author to obtain the highest accuracy. For Soft-NMS, in addition to the overlap threshold, *Nt*, set to 0.3, there was also a, σ, set by the Soft-NMS author. By comparing the object confidence with this threshold, the detected bounding box of the error prediction was finally removed. The σ value was set to 0.4. Setting the threshold too high would have removed all the detected bounding box, and setting the threshold too low would have reduced the detection accuracy because when the detected bounding box has a very high overlap rate, it is more likely to be a repeated detected bounding box. Setting a low threshold meant that the detected bounding box would rarely be removed. After trying many values for this threshold in the PASCAL VOC 2007 dataset, we got the highest accuracy when it was set to 0.4 (as seen in Figure 8). The input resolution of the network was set to 416. After detecting the PASCAL VOC dataset, we calculated the accuracy of the detection. The accuracy of YOLOv3 using traditional NMS and YOLOv3 was improved after using Soft-NMS and the retrieval algorithm, as shown in Table 2.

Table 2. The accuracy results of YOLOv3 improved using Soft-NMS and the retrieval algorithm. The test dataset was PASCAL VOC.

Method	Accuracy (%)
YOLOv3	43.0
YOLOv3 using Soft-NMS and the retrieval algorithm	46.1

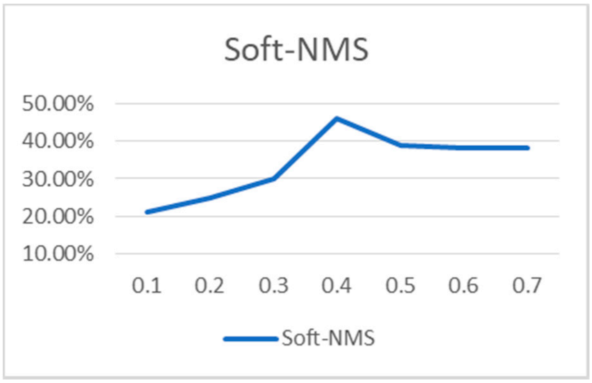


Figure 8. The figure shows the accuracy of person detection under different σ threshold values; the highest accuracy was achieved at 0.4.

In Figure 9, the performance of Soft-NMS and the retrieval algorithm can be seen. In Figure 9b, it can be seen that the girl with the white shirt was missed by NMS, but she was detected using Soft-NMS, as seen in Figure 9a. In Figure 9c, it can be seen that when using the retrieval algorithm, the missed people with blue boxes were retrieved.

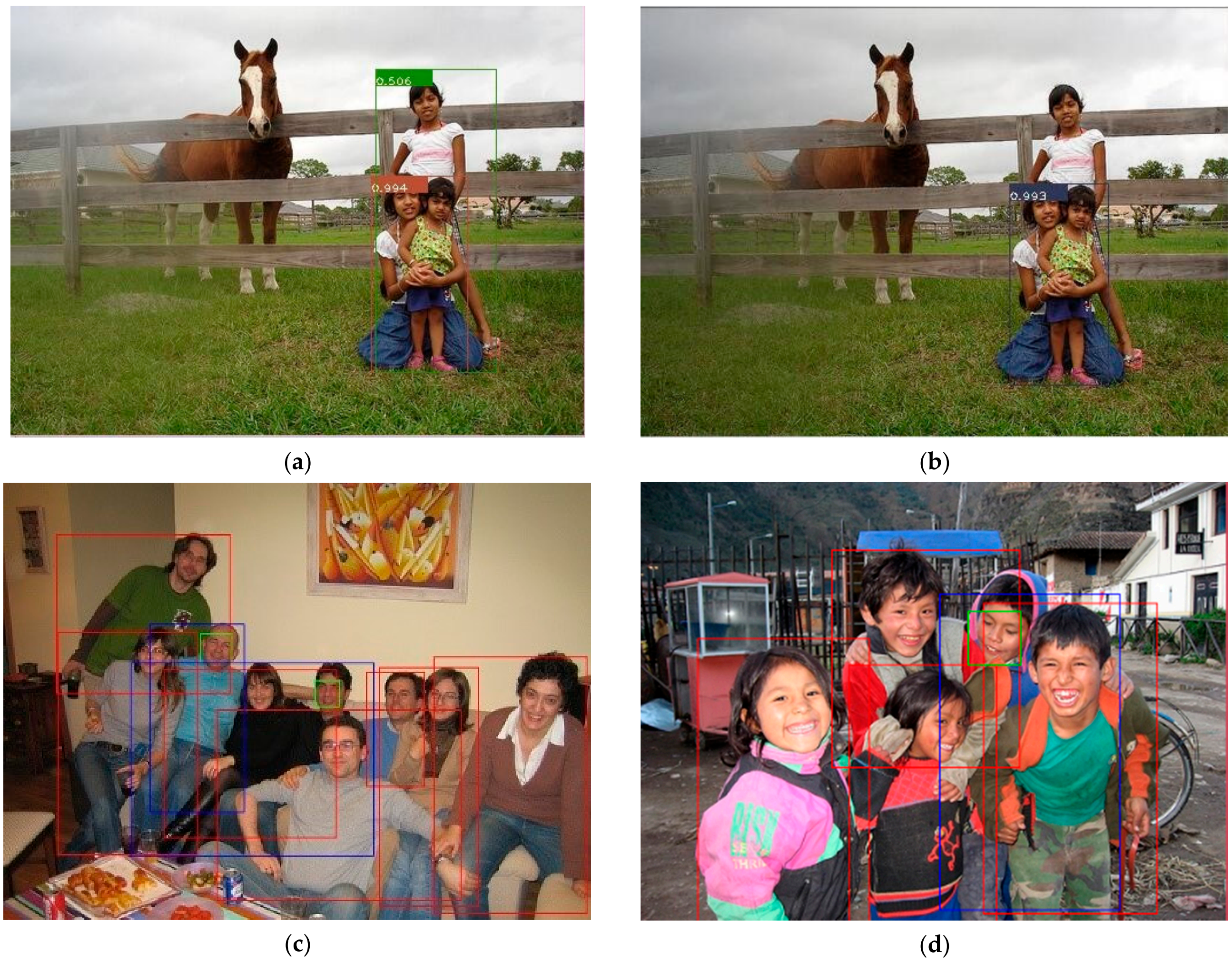


Figure 9. The improvement for YOLOv3 using Soft-NMS and the retrieval algorithm. (**a**) Detection using Soft-NMS; (**b**) detection using NMS; (**c**) detection using Soft-NMS and the retrieval algorithm; (**d**) detection using Soft-NMS and the retrieval algorithm.

4.3. Tracking Effect Analysis

The model uses the OTB-100 (Object Tracking Benchmark 100) dataset [15] to test the accuracy. Because it is used for pedestrian tracking, and the model selects several videos containing pedestrians, including the "human 6 video," the "woman video," and the "girl 2 video," for calculations. The success rate of OTB benchmark was used to indicate the performance of all algorithms, and the overlap score for the success rate was set to 0.5. Additionally, the comparison algorithms were the KCF, ASLA (adaptive structural local sparse appearance) [16], TLD (tracking learning detection) [17], DASiamRPN, and SiamRPN. For the fusion metric, λ was set to 0.2, and the pedestrian detection confidence of YOLOv3 was set to 0.4.

It can be seen from Table 3 that the success rate of our proposed model was greatly improved compared to the traditional KCF, ASLA, and TLD algorithms. For the DASiamRPN, the neural network-based method of the SiamRPN also improved.

Table 3. Tracking success rate list of each tracking algorithm in several pedestrian videos of OTB-100. ASLA: adaptive structural local sparse appearance; TLD: tracking learning detection; FPS: frames per second. SiamRPN: Siamese region proposal network; DASiamRPN: distractor-aware SiamRPN.

Name of Tracking Algorithm	"Human 6 Video"	"Woman Video"	"Girl 2 Video"	FPS
	Success Rate	Success Rate	Success Rate	
KCF	0.20	0.69	0.05	245
ASLA	0.38	0.14	0.55	12
TLD	0.30	0.13	0.07	37
DASiamRPN	0.91	0.93	0.87	160
SiamRPN	0.75	0.94	0.94	200
Ours	**0.97**	**0.94**	**0.94**	**80**

In Table 3, the FPS (frames per second) of all the algorithms are also shown. The KCF, ASLA, and TLD were run on an Inter i3-6100 CPU at 3.70 GHz. The SiamRPN, DASiamRPN, and SKCFMDF are based on neural networks, and they were run on a Nvidia Titan X GPU.

Like the flaws within the KCF, the features that ASLA and TLD extracted were not robust for pedestrian posture changes even though their calculations were fast. However, SKCFMDF extracted deep features using a neural network, so it achieved a higher performance compared to the traditional ones.

Compared to the SiamRPN, there is a lot of improvement for SKCFMDF with the "human 6 video," because the color of the pedestrian in the "human 6 video" is more like the background, which is not the case in the other videos.

Compared to the DASiamRPN, there were not large differences for SCKFMDF in all three videos, because based on the SiamRPN, a distractor-aware module was added to the DASiamRPN, and in the tracking process, the DASiamRPN updated the framework with new samples in real time.

5. Conclusions

Aiming to fix the scale adaptation and occlusion problems of the KCF, an improved KCF pedestrian tracking algorithm, SKCFMDF (which integrates deep features) was proposed. By introducing deep features and target detection algorithms, the accuracy of the tracking algorithm was finally improved. Using Soft-NMS and the retrieval algorithm, YOLOv3′s detection accuracy was increased by 3.1%. By combining the YOLOv3 detection box and the KCF detection box, the scale-adaptive problem of the KCF was solved. By using the deep feature of the pedestrian extracted by a newly designed neural network, the occlusion problem of the KCF and the flaws of the HOG feature were solved. Compared to other mainstream pedestrian tracking algorithms, SKCFMDF is found to be much better.

However, the SKCFMDF algorithm still has room for improvement. The pedestrian target detection framework YOLOv3 can be replaced with a more efficient detection framework in the future, or it can be used in an environment with obvious pedestrian characteristics based on the MobileNet network [18] or YOLO-lite [19]. The target detection networks of ShuffleNet [20] and FBNet [21] can be used to reduce the calculation time. The neural network that extracts deep features could also be replaced with other efficient pedestrian recognition algorithms.

Author Contributions: Conceptualization, Y.Z. and W.Y.; methodology, Y.Z. and W.Y.; software, Y.Z. and W.Y.; validation, Y.Z. and W.Y.; formal analysis, Y.Z. and W.Y.; writing—original draft preparation, Y.Z. and W.Y.; writing—review and editing, Y.S.; visualization, Y.S.; supervision, W.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: We thank the machine vision lab in Hebei University for the equipment and other helps it offers.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Schroff, F.; Kalenichenko, D.; Philbin, J. FaceNet: A Unified Embedding for Face Recognition and Clustering. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 8–12 June 2015; pp. 815–823.
2. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
3. Li, B.; Yan, J.; Wu, W.; Zhu, Z.; Hu, X. High Performance Visual Tracking with Siamese Region Proposal Network. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition IEEE, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8971–8980.
4. Zhu, Z.; Wang, Q.; Li, B.; Wu, W.; Yan, J.; Hu, W. Distractor-Aware Siamese Networks for Visual Object Tracking. *arXiv* **2018**, arXiv:1808.06048.
5. Lowe, D.G. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
6. João, F.; Henriques, C.R.; Martins, P.; Batista, J. High-Speed Tracking with Kernelized Correlation Filters. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *37*, 583–596.
7. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), San Diego, CA, USA, 20–25 June 2005; pp. 886–893.
8. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
9. Lin, T.; Maire, M.; Belongie, S.J.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common objects in context. *arXiv* **2014**, arXiv:1804.02767.
10. Lin, T.-Y.; Dollar, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 936–944.
11. Kinga, D.; Adam, J.B. A Method for Stochastic Optimization. In Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015.
12. Zheng, L.; Shen, L.; Tian, L.; Wang, S.; Wang, J.; Tian, Q. Scalable Person Re-identification: A Benchmark. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Institute of Electrical and Electronics Engineers (IEEE), Santiago, Chile, 7–13 December 2015; pp. 1116–1124.
13. Weinberger, K.Q.; Saul, L.K. Distance Metric Learning for Large Margin Nearest Neighbor Classification. *J. Mach. Learn. Res.* **2009**, *10*, 207–244.
14. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The Pascal Visual Object Classes (VOC) Challenge. *Int. J. Comput. Vision.* **2010**, *88*, 303–338. [CrossRef]
15. Wu, Y.; Lim, J.; Yang, M.-H. Online Object Tracking: A Benchmark. In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 2411–2418.
16. Jia, X.; Lu, H.; Yang, M.-H. Visual tracking via adaptive structural local sparse appearance model. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Rhode Island, RI, USA, 16–21 June 2012; pp. 1822–1829.
17. Kalal, Z.; Mikolajczyk, K.; Matas, J. Tracking-Learning-Detection. *IEEE Trans. Softw. Eng.* **2012**, *34*, 1409–1422. [CrossRef] [PubMed]
18. Howard, A.; Sandler, M.; Chen, B.; Wang, W.; Chen, L.-C.; Tan, M.; Chu, G.; Vasudevan, V.; Zhu, Y.; Pang, R.; et al. Searching for MobileNetV3. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27–28 October 2019; pp. 1314–1324.
19. Huang, R.; Pedoeem, J.; Chen, C. YOLO-LITE: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers. In Proceedings of the 2018 IEEE International Conference on Big Data, Institute of Electrical and Electronics Engineers (IEEE), Seattle, WA, USA, 10–13 December 2018; pp. 2503–2510.

20. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, MA, USA, 18–23 June 2018; Institute of Electrical and Electronics Engineers (IEEE): New York, NY, USA, 2018; pp. 6848–6856.
21. Wu, B.; Keutzer, K.; Dai, X.; Zhang, P.; Wang, Y.; Sun, F.; Wu, Y.; Tian, Y.; Vajda, P.; Jia, Y. FBNet: Hardware-Aware Efficient ConvNet Design via Differentiable Neural Architecture Search. *arXiv* **2018**, arXiv:1812.03443.