Hindawi
Journal of Advanced Transportation
Volume 2021, Article ID 1396326, 11 pages
https://doi.org/10.1155/2021/1396326

Research Article

The Automatic Detection of Pedestrians under the High-Density Conditions by Deep Learning Techniques

Cheng-Jie Jin,[1,2] **Xiaomeng Shi**,[1,2] **Ting Hui**,[1,2] **Dawei Li**,[1,2] **and Ke Ma**[1,2]

[1]*Jiangsu Key Laboratory of Urban ITS, Southeast University of China, Nanjing, Jiangsu 210096, China*
[2]*Jiangsu Province Collaborative Innovation Center of Modern Urban Traffic Technologies, Nanjing, Jiangsu 210096, China*

Correspondence should be addressed to Cheng-Jie Jin; yitaikongtiao@gmail.com

Received 21 January 2020; Revised 12 October 2020; Accepted 8 April 2021; Published 19 April 2021

Academic Editor: Alain Lambert

The automatic detection and tracking of pedestrians under high-density conditions is a challenging task for both computer vision fields and pedestrian flow studies. Collecting pedestrian data is a fundamental task for the modeling and practical implementations of crowd management. Although there are many methods for detecting pedestrians, they may not be easily adopted in the high-density situations. Therefore, we utilized one emerging method based on the deep learning algorithm. Based on the top-view video data of some pedestrian flow experiments recorded by an unmanned aerial vehicle (UAV), we produce our own training datasets. We train the detection model by using Yolo v3, a very popular deep learning model among many available detection models in recent years. We find the detection results are good; e.g., the precisions, recalls, and F1 scores could be larger than 0.95 even when the pedestrian density is as high as 9.0 ped/m^2. We think this approach could be used for the other pedestrian flow experiments or field data which have similar configurations and can also be useful for automatic crowd density estimation.

1. Introduction

The study of pedestrian flow dynamics dates back to the 1960s [1–3]. Ever since, the pedestrian data collection is one of the fundamental works for this field. Generally speaking, there are two major ways to obtain the data: the first approach is to gather empirical data, which comes from the real life; the second one is to conduct experiments and ask the recruited participants to move according to the orders [4]. Since the parameters and conditions could be controlled in the experiments, it is a great help for theoretical studies [5, 6]. In recent years the second way becomes more and more popular in this field [7, 8].

In most pedestrian flow experiments, for the convenience of measuring the positions and velocities, the participants are usually required to wear markers such as caps [6, 9–14]. At earlier stage, pedestrian motion data were extracted manually or using semi-automatic tracking techniques [10, 15–17]. Some researchers used some simple software to extract the data of these caps manually, e.g., Tracker (http://physlets.org/tracker/) [18–21]. But it needs lots of time to finish, and the efficiency is quite low. Meanwhile, some researchers adopted the unsupervised methods from computer vision field to detect and track pedestrians automatically [10, 15, 17, 22]. Boltes et al. developed automatic detecting and tracking software named PeTrack [9, 23]). This software has been used in many recent studies of pedestrian flow experiments [24–28].

Nevertheless, the limitations of PeTrack are also clear. The preparation work for PeTrack is heavy in terms of the camera views and parameters. Although the detection results of PeTrack are good when the pedestrian density is not high, in some experiments under high-density [20, 21] its performance is not good, especially when the density reaches 8~9ped/m^2. Besides, the operation of PeTrack is difficult, since many important parameters are not easy to be determined. Sometimes, the adjustment of these parameters is very complex. Therefore, we think a better method is needed.

In recent years, the development of deep learning techniques gives us an easier way to solve this problem. Many new algorithms of object detection are proposed in recent years, e.g., RCNN [29], Fast RCNN [30], Faster

RCNN [31], R-FCN [32], SSD [33], Yolo series [34–36], etc. Among them, Yolo ("You Only Look Once") is a balanced object detection model in terms of the speed and accuracy. It is a one-stage detection model, which skips the region proposal stage, and runs detection directly over a dense sampling of possible locations. In many previous studies, it was found that the speed of Yolo v3 [36] is faster than the other models. Its accuracy is much higher than the two previous versions, including Yolo v1 [34] and Yolo v2 (also named Yolo 9000) [35]. And the performance of detecting small objects is also significantly improved. Therefore, in this paper we choose Yolo v3 to do the job.

The pretrained version of Yolo v3 with open datasets (such as ImageNet and COCO) has good performance when detecting multi-types of objects, including pedestrians [37, 38]. However, since the pretrained labels are primarily annotated using real life images from side-view cameras, the performance is not good when directly using for many pedestrian flow experiments, especially when the cameras are perpendicular to the ground. The features of pedestrians appeared in the top-view cameras are quite different from that when shooting from the side view. Therefore, we have to train the new model with the samples of various caps found in the video data. We find the trained results are satisfactory: most pedestrians could be immediately detected by Yolo v3, and both the precision and recall are high enough for extracting our required pedestrian flow parameters.

The focus of this paper is not to develop the state-of-the-art pedestrian detection models or algorithms, but rather try to explore the capabilities of applying the new detection approaches into pedestrian traffic domain. We will demonstrate that, through a simple training process, the deep learning model Yolo V3 could be able to achieve a good detection results for pedestrian flow analysis. And we open-source a series of training datasets for pedestrians wearing caps at top-view from UAV videos. The reproducibility of our methods could be proved by using another dataset for validation.

The rest of this paper is organized as follows. In Section 2, the brief introduction of Yolo v3 is given. Section 3 introduces the characteristics of our pedestrian flow experiments under high-density conditions and also shows the training process. Section 4 discusses the testing results of the detection model. Section 5 presents some notes about the application of this model, and the conclusion is shown in Section 6.

2. The Brief Introduction of Yolo v3

Since the main topic of this paper is to discuss the detection of pedestrians by training, here we only give a brief introduction for the improvement in Yolo v3, which could be found in Figure 1.

Firstly, Yolo v3 creates one new feature extraction network named the Darknet-53, rather than the Darknet-19 used in Yolo v2. The accuracy of the Darknet-53 is close to that of the ResNet-101 and ResNet-152, but much faster.

And then, Yolo v3 introduces prediction cross scales by using the concept of feature pyramid networks. It predicts boxes at 3 different scales and extracts features from those scales. In the scale 1, the objects are sampled by the convolutional layer of the penultimate layer. In the scale 2, the $16 * 16$ size feature map is added, and the accuracy of detecting medium objects can be improved. In the scale 3, the $32 * 32$ size feature map is used, which makes the detection accuracy of small-scale objects similar to that of medium objects.

Besides, The YOLOv3 no longer uses the Softmax function to classify each box, in order to avoid the overlapping category labels for the objects. Instead, the independent multiple logical classifiers are used, and the classification loss is represented by the binary cross entropy loss. Due to the above improvements, on the COCO dataset Yolo v3 can finally achieve the accuracy which is similar to that of RetinaNet, but Yolo v3 is nearly four times faster.

3. The Video Data and Training Process

For the use of Yolo v3, there are many versions written by different programming languages. Since Python is very popular and easy to use for deep learning, we choose a Python version of Yolo v3 based on the framework of Keras (https://github.com/qqwweee/keras-yolo3). Figure 2 shows the detection results by the pretrained datasets of Yolo v3. Figure 2(a) is a snapshot of one simple pedestrian flow experiment conducted about 5 years ago, and the camera is not high. We can find that nearly all the pedestrians can be easily detected in Figure 2(b). For such a situation, training is not necessary.

However, if we record the experimental video by UAV, and the camera is very high from the ground, the situation will be significantly different. In the winter of 2016, 2017, and 2018, we conducted three large-scale pedestrian flow experiments in the campus of Southeast University [20, 21, 39]. We asked the students recruited to move on the circular road, and the boundaries of the road were built by plastic stools. The widths of the road were 1.5 m in the 2016 experiment, 2.5 m in the 2017 experiment, and 2m in the 2018 experiment: these differences have no influence on the detection of pedestrians. At the same time, the colors of plastic stools were also different: they were blue in 2016 and 2017 and white in 2018. These differences can slightly influence the detection results, which will be discussed later.

In these experiments, the pedestrians wore two types of caps. The first part was unidirectional movement, and all the pedestrians just moved together. The second part was bidirectional movement: the ones with red caps went on moving forward, while the ones with blue caps turned around and moved forward again. In each experiment, the proportions of red ones and blue ones were about 50 : 50. In the 2016 experiment, we asked one pedestrian to wear yellow cap and considered it as a special one. But in the 2017 and 2018 experiment, we thought it was not necessary, and no longer used such configuration. Therefore, we do not have enough samples of yellow caps, and we only try to do the detection on the red ones and blue ones.

In the experiments, the authors use different pedestrians in different runs, in order to study the flow-density and

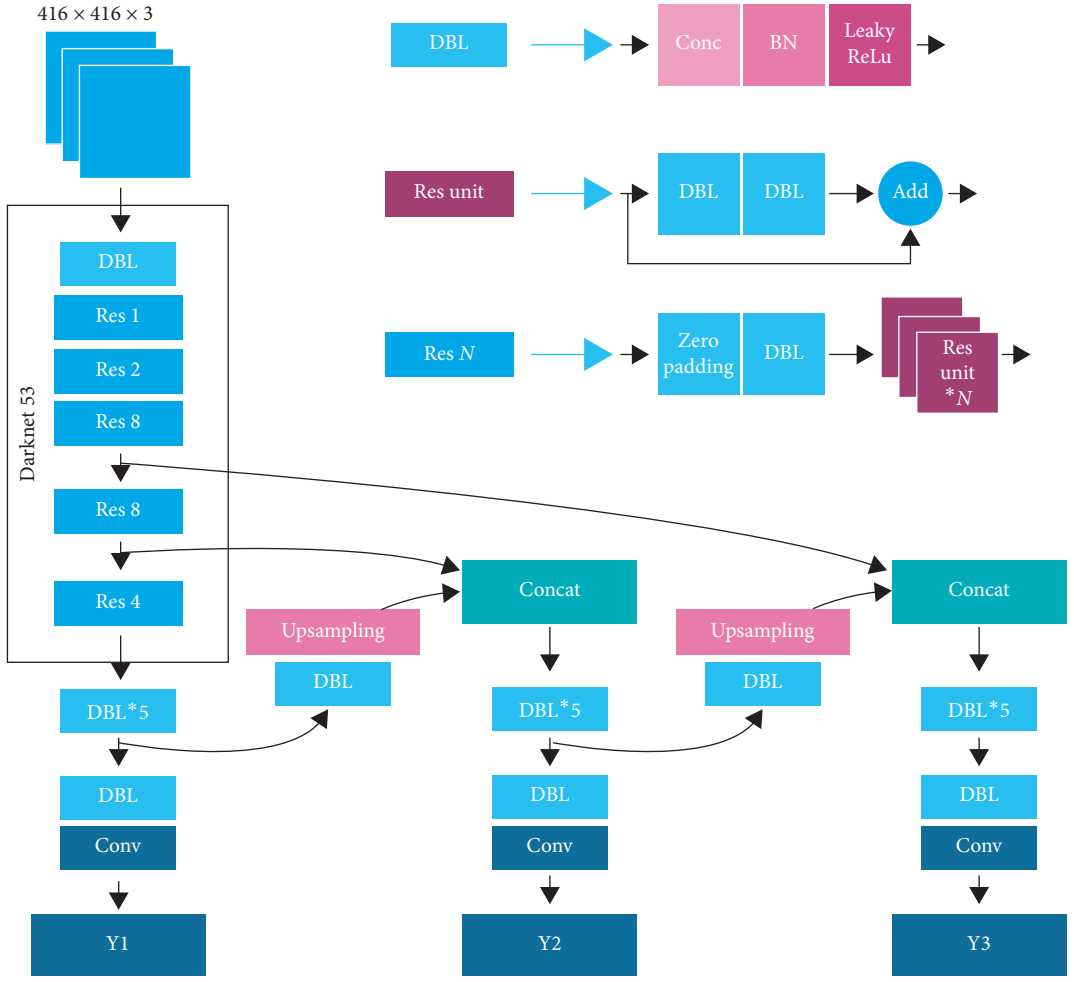


Figure 1: The structure of Yolo v3.

(a) (b)

Figure 2: The images of one simple pedestrian flow experiment under circular boundary conditions. It was conducted at 2014. (a) The original image. (b) The results detected by the pretrained Yolo v3.

velocity-density relationship of pedestrian movement. Here, the authors show 4 typical images with different global densities (the local density in some area could be a little larger or smaller than the global value, but the calculation and discussion of this topic is out of the scope of this paper) in Figure 3. For example, in Figure 3(c), there are 175 red ones and 182 blue ones, and the area of the circular road is about 51.1 m2. Therefore, the averaged global density is about 7.0ped/m^2. In this paper, the authors name all the experimental runs as "year-predetermined density (ρ_P) (in some runs, the predetermined density is a little different from the actual value. But these differences have no influence on the topic discussed in this paper)-order of run," e.g., "16-8-2" means $\rho_P = 8\,\mathrm{m}^{-2}$, the run is the second run, and the experiment was conducted in 2016.

We find, in all of these images, no pedestrian could be detected by the pretrained Yolo v3. We think it is possible that, in the training datasets of original Yolo v3, most

samples of pedestrians are shoot from the side, rather than from the top. Therefore, we need to help Yolo v3 recognize them, and use the red caps and blue caps as the training datasets. Actually, these caps could be considered as small objects. Thanks to the recent improvement of Yolo v3, the detection of these small objects could have good performances.

In order to check how many images are enough for the training process, we try to use different training sets, and the 4 images in Figure 3 are used for testing the differences between them. The training images are extracted from the video of pedestrian flow experiments with low densities, and the time interval is 1s, as shown in Table 1. For example, in dataset T2, we use two videos. The first one (16-1-1) is 110 s, and 25 pedestrians with red and blue caps are used in this run (this number does not change, since we use circular road). The second one (16-2-1) is 128 s, and 55 ones are used in this run. And then, we use the software named LabelImg (https://github.com/tzutalin/labelImg) to manually label the red caps and blue caps in these images. In each dataset, 25% images are used as validation sets. For the training, the learning rate is set as 10^{-3}. We have tried some other hyperparameters, e.g., set the learning rate as 10^{-4}. But it is found that the training results keep nearly unchanged.

Here, we show some typical training processes in Figure 4. The GPU we used for training is Tesla T4 with 16G RAM. Due to the limitation of resources, if we choose larger batch size, e.g., 16 or 32, our program cannot work. Since batch size = 8 is possible, in this paper we set batch size = 10, which is just between 8 and 16.

We also show the results when batch size = 4 and 8 for comparisons. It is clear that, in Figures 4(a) and 4(c), all the losses gradually decrease with time, although sometimes the results of validation set increase a little. No obvious overfitting phenomenon is observed during the training. In Figure 4(d), we can further confirm that the influence of different batch sizes on the training is very small. Besides, here we adopt the early stopping mechanism to determine the end of training. The criterion is when the validation loss is smaller than 120, we just stop. Therefore, the epochs needed for batch size = 4, 8, 10 are 447, 424, 441, which are also close to each other.

4. Results and Evaluations

In this section, we evaluate the training results. We show the statistics of different datasets in Table 2, including precisions, recalls, and F1 scores. Here, we set score = 0.4, IOU = 0.2, since we find these values are suitable for the video data after tuning up the parameters. We can find the following.

(1) When the training set is small (e.g., T1), the testing results are always not good.

(2) For the low-density situation (e.g., Figure 3(a)), starting from T2, all the three results become much better.

(3) But if the densities become higher, T2 is also not enough, and the growth of sample size is very necessary. Until T4 is used, the results seem satisfactory, as shown in Figure 5(d). Thus, in the following tests, we will use the result of T4.

(4) For the fixed parameters (score = 0.4, IOU = 0.2), we can find that the precisions are always very high in our detection: most of them are larger than 0.95. In Table 2, the differences between different datasets mainly come from the increase of recalls.

And then, we check the results of T4 on the testing sets. The model trained in low-density situations is used for the detection under high-density condition. In Table 3, the data of 32 runs are used for testing. For each experimental video, we get the snapshots with the interval of 15s (e.g., if the total running time is 263s, the image number we choose will be $263/15 + 1 \approx 19$). The values of precisions, recalls, and F1 scores are the averaged results of these images. It is clear that for all the runs the values of precisions, recalls, and F1 scores are greater than 0.95, which means the results are good enough.

Although the performance of our model is quite good, there are small amount of errors in the detection results. After carefully checking, we find they mainly result from the following factors:

(1) The high densities when the pedestrians are too close to each other. As shown by the yellow circles in Figure 6(a), some red caps and some blue caps are not detected. Sometimes they are overlapped, which makes the detection more difficult. Actually, many smaller recalls come from this factor, e.g., 16-9-1, 16-9-2, etc.

(2) The color of the clothes. Some pedestrians' clothes (or shoes) are red or blue, which are similar to the color of caps. This also makes the detection difficult: some errors are marked by the two yellow circles in Figure 6(b).

(3) The interference of stools. Sometimes the stools are recognized as blue caps, as marked by the two yellow circles in Figure 6(c). This type of errors only occasionally occurs in our results.

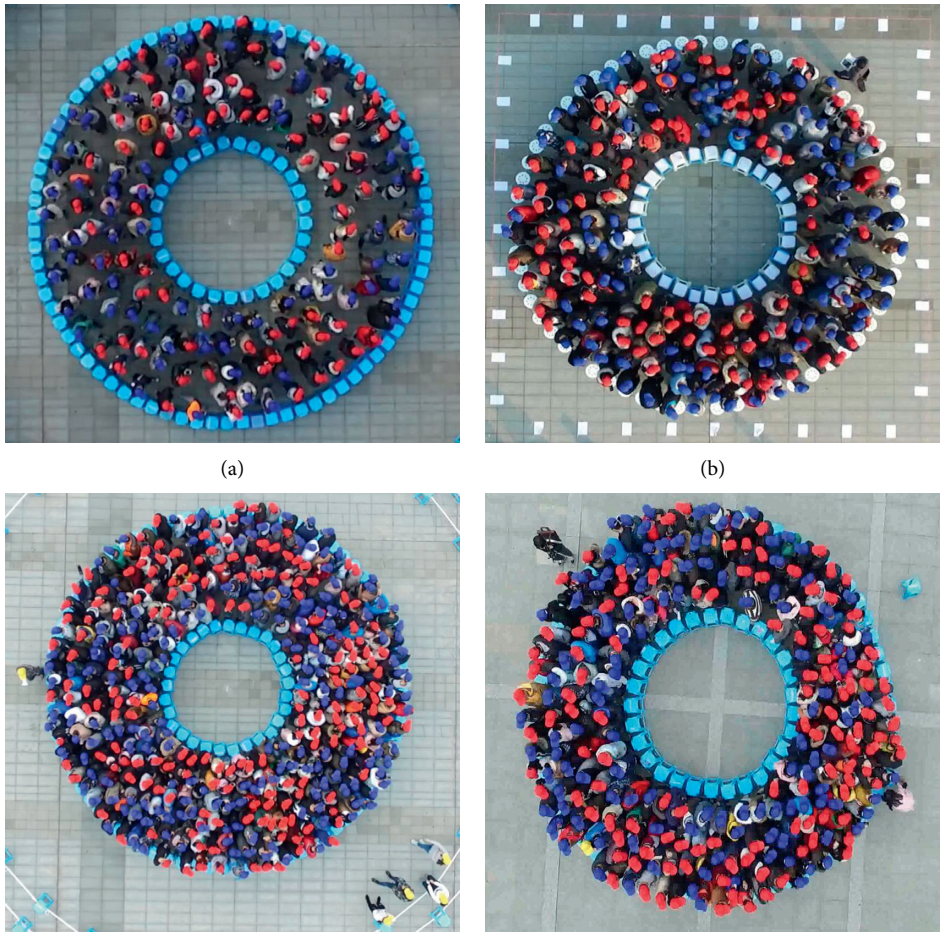
(4) Some special cases. For example, in Figure 6(d), we can see that one pedestrian rises his head. This behavior makes his cap not detected at this moment. For such a situation, it is not the responsibility of our model.

5. Some Notes about the Applications

Since this program is a great help for extracting data from the video of pedestrian flow experiments, we have uploaded the codes and the trained model in Github.com. The link is https://github.com/chengjie-jin/detection-model-for-pedestrian-experiments. We hope this program could be beneficial for the other researchers in the field of pedestrian flow dynamics.

Finally, about the application of this model, some notes should be introduced:

(1) In our detection results, usually the precision values are close to 1.0, while the recall values are a little



(a) (b) (c) (d)

Figure 3: Some typical images of our pedestrian flow experiments. (a)ρ = 2.9ped/m^2, 145 objects, in 17-3-1; (b)ρ = 4.7ped/m^2, 175 objects, in 18-5-1; (c)ρ = 7.0ped/m^2, 357 objects, in 17-7-1; (d)ρ = 9.0ped/m^2, 234 objects, in 16-9-1.

Table 1: The information of training sets.

Name	Run	Number of images	Objects in each image	Total objects used
T1	16-1-1	110	25 * 110	2750
T2	16-1-1, 16-2-1	238	25 * 110, 55 * 128	9790
T3	16-1-1, 16-2-1, 17-1-1, 18-1-1	543	25 * 110, 55 * 128, 49 * 150, 39 * 155	23185
T4	16-1-1, 16-2-1, 17-1-1, 18-1-1, 17-2-1, 18-2-1	933	25 * 110, 55 * 128, 49 * 150, 39 * 155 99 * 210, 74 * 180	57295

smaller. For such a situation, the pedestrians which are not detected could be added by hand. The simplest way is to use *Microsoft Paint* in Windows: move the mouse pointer on the center point of one cap, and we can immediately see the coordinate (the position of only one yellow cap in 2016 experiments could also be recorded in such a way). Just input the numbers and the color in the csv file to finish the work. Even for the hardest level (in 16-9-1), usually only 5–10 pedestrians need to be added manually, and it could be quite fast. After that, we can make some transformations for the position data according to the perspective of camera and set up the coordinate system. And then, related analysis could be performed on the pedestrian dynamics in the experiments, which could be found in other paper [39].

(2) Although the training datasets in this paper are obtained from the UAV video, the trained models also could be used for the other videos of pedestrian flow experiment, as long as the participants wear similar caps. For example, in Figure 7(a) we can see

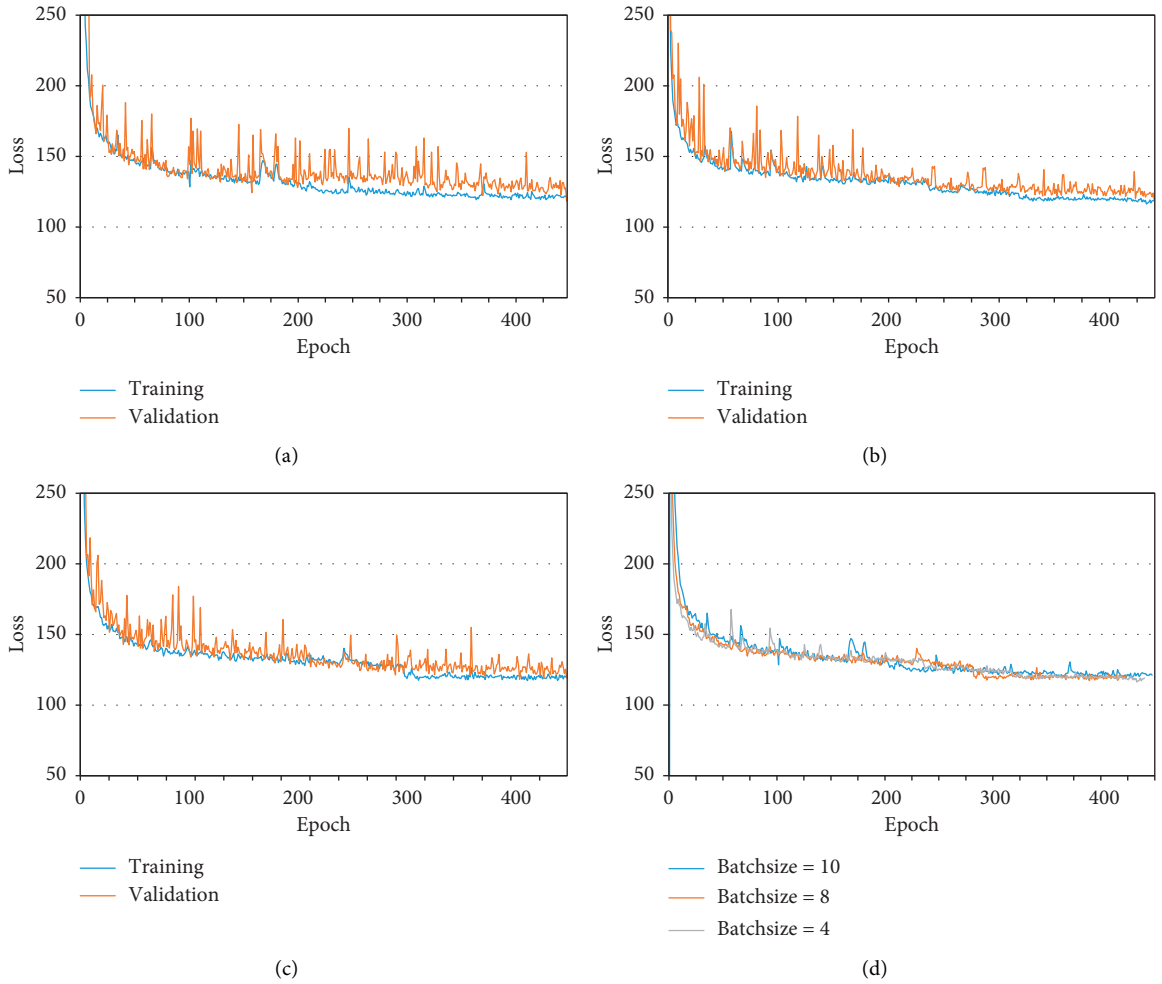
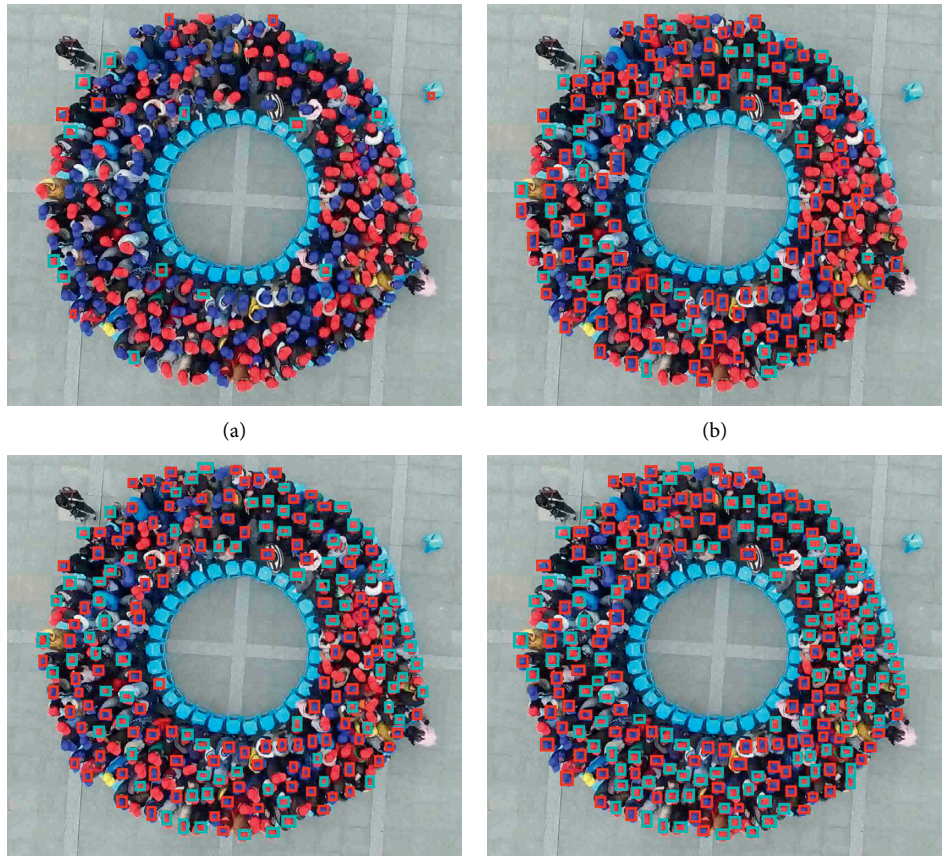


Figure 4: The losses during the training with dataset T4. (a) When batch size = 10; (b) when batch size = 8; (c) when batch size = 4; (d) the comparisons of training losses between different batch sizes.

Table 2: The statistics when detecting the 4 typical samples by our model.

	T1	T2	T3	T4
(a) The results of Figure 3(a)				
Precision	0.96	1.00	1.00	0.99
Recall	0.16	0.89	0.99	1.00
F1 score	0.27	0.94	1.00	0.99
(b) The results of Figure 3(b)				
Precision	0.86	0.94	0.97	0.98
Recall	0.14	0.88	0.97	1.00
F1 score	0.25	0.91	0.97	0.99
(c) The results of Figure 3(c)				
Precision	0.98	1.00	1.00	0.99
Recall	0.10	0.79	0.87	0.98
F1 score	0.17	0.88	0.93	0.99
(d) The results of Figure 3(d)				
Precision	0.95	0.99	0.99	0.99
Recall	0.08	0.64	0.81	0.95
F1 score	0.15	0.78	0.89	0.97



(a) (b) (c) (d)

Figure 5: The detection results of Figure 3(d) with different training sets. (a) T1; (b) T2; (c) T3; (d) T4. The scores of each object are not presented, since they could mask the positions of other pedestrians.

Table 3: The statistics of testing sets.

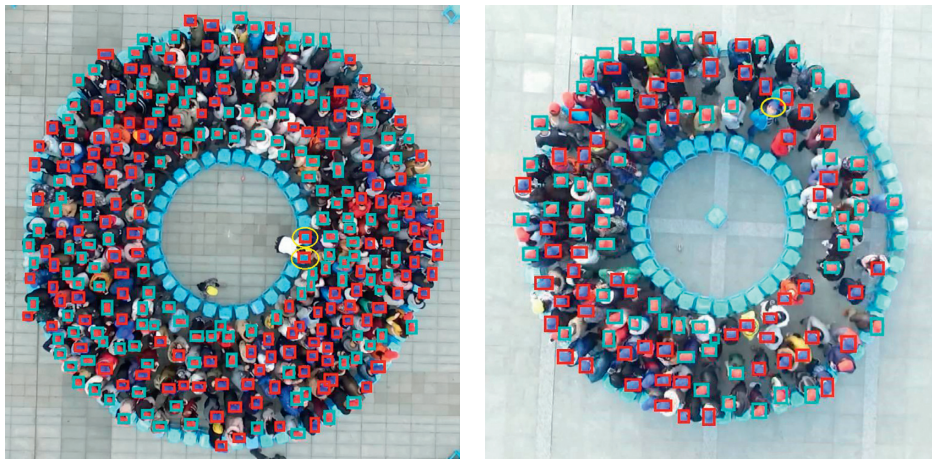
Name	Red ones	Blue ones	Total objects	Total running time (s)	Images chosen	Averaged precision	Averaged recall	Averaged F1 score
16-3-1	42	40	82	106	8	0.986	0.998	0.992
16-4-1	52	52	104	197	14	0.976	0.996	0.986
16-5-1	65	60	125	263	19	0.977	0.994	0.985
16-5-2	64	58	122	250	18	0.985	0.974	0.979
16-6-1	70	76	146	423	29	0.975	0.971	0.973
16-6-2	74	73	147	267	19	0.981	0.978	0.980
16-7-1	78	84	162	555	38	0.981	0.987	0.984
16-7-2	83	83	166	411	28	0.988	0.977	0.982
16-8-1	100	99	199	623	43	0.992	0.959	0.975
16-8-2	99	99	198	360	25	0.987	0.951	0.969

Table 3: Continued.

Name	Red ones	Blue ones	Total objects	Total running time (s)	Images chosen	Averaged precision	Averaged recall	Averaged F1 score
16-9-1	117	117	234	1198	41	0.983	0.958	0.970
16-9-2	111	105	216	383	27	0.990	0.962	0.976
17-3-1	77	68	145	548	38	0.994	0.996	0.995
17-4-1	96	109	205	304	21	0.993	0.986	0.990
17-4-2	100	98	198	205	15	0.994	0.990	0.992
17-5-1	128	120	248	339	24	0.991	0.992	0.991
17-5-2	124	125	249	291	20	0.997	0.985	0.991
17-6-1	148	148	296	362	25	0.989	0.985	0.987
17-6-2	147	154	301	278	20	0.995	0.983	0.989
17-7-1	175	182	357	470	32	0.996	0.984	0.990
17-7-2	176	182	358	357	25	0.997	0.969	0.983
18-3-1	55	51	106	540	37	0.999	1.000	0.999
18-4-1	74	73	147	528	36	0.982	0.995	0.989
18-4-2	74	72	146	295	21	0.981	0.982	0.981
18-5-1	89	86	175	413	29	0.965	0.993	0.979
18-5-2	84	94	178	373	26	0.986	0.989	0.987
18-6-1	110	114	224	428	30	0.980	0.977	0.979
18-6-2	103	106	209	346	24	0.981	0.982	0.982
18-7-1	133	134	267	452	31	0.970	0.977	0.973
18-7-2	126	125	251	456	31	0.987	0.977	0.982
18-8-1	147	147	294	713	49	0.952	0.981	0.965
18-8-2	146	149	295	487	33	0.981	0.971	0.976

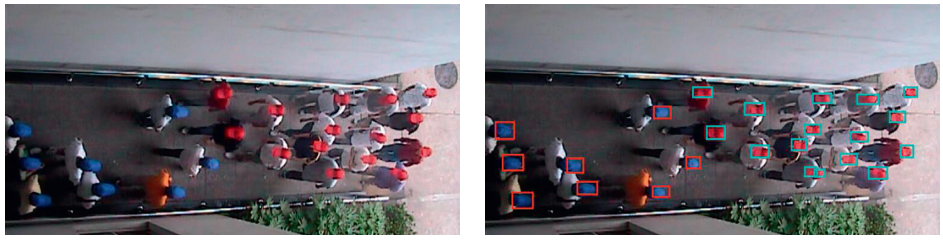
the image of one experiment under open boundary conditions (the discussion of the detail of this bi-directional experiment could be found in [5]), and the height of the camera is much smaller than that of UAV. Most of the pedestrians with red and blue caps could be correctly detected in Figure 7(b). Besides, if the pedestrians in other experiments have different colors of caps (e.g., black or white), the method introduced in this paper could be copied: use Yolo v3 to do the similar training if possible.

(a) (b)



(c) (d)

Figure 6: Some typical errors in the detection results. (a) Some caps are not detected due to the very high densities, in 16-9-1; (b) some clothes (or shoes) are recognized as caps due to the similar colors, in 16-5-1; (c) some stools are occasionally recognized as blue caps, in 17-6-1; (d) one cap is not detected, since the pedestrian rises his head at this moment, in 16-5-2.



(a) (b)

Figure 7: The images of another pedestrian flow experiment under open boundary conditions. It was conducted at 2015. (a) The original image; (b) the results detected by our Yolo v3 model.

6. Conclusions

In this paper, we testify the usability of an object detection model based on deep learning techniques in terms of detecting the pedestrians recorded in top-view cameras. Since the previous software (e.g., PeTrack) does not perform well for the high-density situation, it is necessary to use some new approach. Yolo v3 is chosen due to the fast speed and decent accuracy, but its pretrained version does not work for our experimental video. Therefore, training is necessary. We choose some images under low-density condition as training datasets, and all the situations with different densities are chosen as testing datasets. We find that when the datasets become larger, the recalls significantly increase, while the

precision is always close to 1.0. For the final model, the performance is good: all the precisions, recalls, and F1 scores are larger than 0.95, even when the pedestrian density is as high as 9.0ped/m^2. This model could be used in other pedestrian flow experiments, as long as they have similar configurations.

Although we have made some progress in the pedestrian detection under high-density conditions, one deficiency of this program is that it does not include the part of tracking pedestrians' trajectories. However, tracking objects is not the function of Yolo v3, which is out of the scope of this paper. In recent years, some tracking programs based on deep learning have been proposed, e.g., Sort and Deepsort. It is possible to merge the detection model and tracking model into an integrated one, which may bring more convenience for the scholars in the field of pedestrian flow studies. In the future, we will try to solve this technical problem and contribute more to this field.

Data Availability

The codes and the trained model are available in Github.com. The link is https://github.com/chengjie-jin/detection-model-for-pedestrian-experiments.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

The authors are very grateful to Ting Hui, Yi-Xiao Zou, Fei Xie, and Hong-Feng Liang for their help to process the video data and label the objects in the images. This work was funded by the National Natural Science Foundation of China (Nos. 71801036, 71901060, and 71971056), the Fundamental Research Funds for the Central Universities, and the Science and Technology Project of Jiangsu Province, China (BZ2020016).

References

[1] J. J. Fruin, "Designing for pedestrians: a level-of-service concept," *Highway Research Record*, vol. 377, pp. 1–15, 1971.

[2] S. J. Older, "Movement of pedestrians on footways in shopping streets," *Traffic Engineering and Control*, vol. 10, no. 4, pp. 160–163, 1968.

[3] F. P. Navin and R. J. Wheeler, "Pedestrian flow characteristics," *Traffic Engineering - Institute of Transportation Engineers*, vol. 39, 1969.

[4] X. Shi, Z. Ye, N. Shiwakoti, and Z. Li, "A review of experimental studies on complex pedestrian movement behaviors," in *Proceedings of the "CICTP 2015" (American Society of Civil Engineers, 2015)*, pp. 1081–1096, Reston, VA, USA, July 2015.

[5] C.-J. Jin, R. Jiang, J.-L. Yin, L.-Y. Dong, and D. Li, "Simulating bi-directional pedestrian flow in a cellular automaton model considering the body-turning behavior," *Physica A: Statistical Mechanics and Its Applications*, vol. 482, pp. 666–681, 2017.

[6] S. Xue, F. Claudio, X. Shi, and T. Li, "Revealing the hidden rules of bidirectional pedestrian flow based on an improved floor field cellular automata model," *Simulation Modelling Practice and Theory*, vol. 100, Article ID 102044, 2020.

[7] X. Shi, Z. Ye, N. Shiwakoti, and O. Grembek, "A state-of-the-art review on empirical data collection for external governed pedestrians complex movement," *Journal of Advanced Transportation*, vol. 2018, pp. 1–42, 2018.

[8] N. Shiwakoti, X. Shi, and Z. Ye, "A review on the performance of an obstacle near an exit on pedestrian crowd evacuation," *Safety Science*, vol. 113, pp. 54–67, 2019.

[9] M. Boltes, A. Seyfried, B. Steffen, and A. Schadschneider, "Automatic extraction of pedestrian trajectories from video recordings," in *Pedestrian and Evacuation Dynamics 2008*, W. W. F. Klingsch, C. Rogsch, A. Schadschneider, and M. Schreckenberg, Eds., Springer, Berlin, Germany, pp. 43–54, 2010.

[10] W. Tian, W. Song, J. Ma, Z. Fang, A. Seyfried, and J. Liddle, "Experimental study of pedestrian behaviors in a corridor based on digital image processing," *Fire Safety Journal*, vol. 47, pp. 8–15, 2012.

[11] A. Garcimartín, D. R. Parisi, J. M. Pastor, C. Martín-Gómez, and I. Zuriguel, "Flow of pedestrians through narrow doors with different competitiveness," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 11, pp. 1–16, 2016.

[12] A. Nicolas, S. Bouzat, and M. N. Kuperman, "Pedestrian flows through a narrow doorway: effect of individual behaviours on the global flow and microscopic dynamics," *Transportation Research Part B: Methodological*, vol. 99, pp. 30–43, 2017.

[13] M. S. Sharifi, K. Christensen, A. Chen, and Z. Song, "Exploring effects of environment density on heterogeneous populations' level of service perceptions," *Transportation Research Part A: Policy and Practice*, vol. 124, pp. 115–127, 2019.

[14] Y. Zhao, T. Lu, W. Su, P. Wu, L. Fu, and M. Li, "Quantitative measurement of social repulsive force in pedestrian movements based on physiological responses," *Transportation Research Part B: Methodological*, vol. 130, pp. 1–20, 2019.

[15] S. P. Hoogendoorn, W. Daamen, and H. L. Piet, "Bovy: "extracting microscopic pedestrian characteristics from video data"" in *Proceedings of the Transportation Research Board 82nd Annual Meeting*, pp. 1–15, Washington, DC, USA, January 2003.

[16] J. Wang, W. Weng, and X. Zhao, "Comparison of turbulent pedestrian behaviors between mina and love parade," *Procedia Engineering*, vol. 84, pp. 708–714, 2014.

[17] A. Johansson, D. Helbing, and P. K. Shukla, "Specification of the social force pedestrian model by evolutionary adjustment to video tracking data," *Advances in Complex Systems*, vol. 10, no. 2, pp. 271–288, 2007.

[18] N. Shiwakoti, Y. Gong, X. Shi, and Z. Ye, "Examining influence of merging architectural features on pedestrian crowd movement," *Safety Science*, vol. 75, pp. 15–22, 2015.

[19] X. Shi, Z. Ye, N. Shiwakoti, D. Tang, C. Wang, and W. Wang, "Empirical investigation on safety constraints of merging pedestrian crowd through macroscopic and microscopic analysis," *Accident Analysis & Prevention*, vol. 95, pp. 405–416, 2016.

[20] C.-J. Jin, R. Jiang, W. Wei, D. Li, and N. Guo, "Microscopic events under high-density condition in uni-directional pedestrian flow experiment," *Physica A: Statistical Mechanics and Its Applications*, vol. 506, pp. 237–247, 2018.

[21] C. J. Jin, R. Jiang, R. Li, and D. Li, "Single-file pedestrian flow experiments under high-density conditions," *Physica A: Statistical Mechanics and its Applications*, vol. 531, 2019.

[22] B. Zhou, X. Wang, and X. Tang, "Understanding collective crowd behaviors: learning a mixture model of dynamic pedestrian-agents," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2871–2878, Washington, DC, USA, June 2012.

[23] M. Boltes and A. Seyfried, "Collecting pedestrian trajectories," *Neurocomputing*, vol. 100, pp. 127–133, 2013.

[24] J. Zhang, W. Klingsch, A. Schadschneider, and A. Seyfried, "Transitions in pedestrian fundamental diagrams of straight corridors and T-junctions," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 6, Article ID P06004, 2011.

[25] J. Zhang, W. Klingsch, A. Schadschneider, and A. Seyfried, "Ordering in bidirectional pedestrian flows and its influence on the fundamental diagram," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2, Article ID P02002, 2012.

[26] S. Cao, J. Zhang, D. Salden, J. Ma, C. Shi, and R. Zhang, "Pedestrian dynamics in single-file movement of crowd with different age compositions," *Physical Review E*, vol. 94, no. 1, Article ID 012312, 2016.

[27] X. Shi, Z. Ye, N. Shiwakoti, D. Tang, and J. Lin, "Examining effect of architectural adjustment on pedestrian crowd flow at bottleneck," *Physica A: Statistical Mechanics and Its Applications*, vol. 522, pp. 350–364, 2019.

[28] Y. Xiao, Z. Gao, R. Jiang, X. Li, Y. Qu, and Q. Huang, "Investigation of pedestrian dynamics in circle antipode experiments: analysis and model evaluation with macroscopic indexes," *Transportation Research Part C: Emerging Technologies*, vol. 103, pp. 174–193, 2019.

[29] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Columbus, OH, USA, June 2014.

[30] R. Girshick, "Fast R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, Santiago, Chile, December 2015.

[31] S. Ren, K. He, R. Girshick, and 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, 2017.

[32] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: object detection via region-based fully convolutional networks," *Advances in Neural Information Processing Systems*, vol. 35, 2016.

[33] W. Liu, D. Anguelov, and D. Erhan, "SSD: single shot multibox detector," in *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*Springer, Berlin, Germany, 2016.

[34] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Seattle, WA, USA, June 2016.

[35] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition*, CVPR, Honolulu, Hawaii, July 2017.

[36] J. Redmon and A. Farhadi, "YOLOv3: an incremental improvement," 2018.

[37] T. Y. Lin, M. Maire, and S. Belongie, "Microsoft COCO: common objects in context," in *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*Springer, Berlin, Germany, 2014.

[38] D. Jia, W. Dong, R. Socher, Li-J. Li, K. Li, and Li Fei-Fei, "ImageNet: a large-scale hierarchical image database," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, June 2009.

[39] C.-J. Jin, R. Jiang, S. C. Wong et al., "Observational characteristics of pedestrian flows under high-density conditions based on controlled experiments," *Transportation Research Part C: Emerging Technologies*, vol. 109, pp. 137–154, 2019.