PeerJ Computer Science

A deep learning-based ensemble method for helmet-wearing detection

Zheming Fan[1], Chengbin Peng[1,2], Licun Dai[1], Feng Cao[1], Jianyu Qi[1] and Wenyi Hua[1]

[1] College of Information Science and Engineering, Ningbo University, Ningbo, China
[2] Ningbo Institute of Industrial Technology, Chinese Academy of Sciences, Ningbo, China

ABSTRACT

Recently, object detection methods have developed rapidly and have been widely used in many areas. In many scenarios, helmet wearing detection is very useful, because people are required to wear helmets to protect their safety when they work in construction sites or cycle in the streets. However, for the problem of helmet wearing detection in complex scenes such as construction sites and workshops, the detection accuracy of current approaches still needs to be improved. In this work, we analyze the mechanism and performance of several detection algorithms and identify two feasible base algorithms that have complementary advantages. We use one base algorithm to detect relatively large heads and helmets. Also, we use the other base algorithm to detect relatively small heads, and we add another convolutional neural network to detect whether there is a helmet above each head. Then, we integrate these two base algorithms with an ensemble method. In this method, we first propose an approach to merge information of heads and helmets from the base algorithms, and then propose a linear function to estimate the confidence score of the identified heads and helmets. Experiments on a benchmark data set show that, our approach increases the precision and recall for base algorithms, and the mean Average Precision of our approach is 0.93, which is better than many other approaches. With GPU acceleration, our approach can achieve real-time processing on contemporary computers, which is useful in practice.

Subjects Computer Vision, Data Mining and Machine Learning, Social Computing
Keywords Ensemble method, Deep learning, Helmet-wearing detection, Face detection

Submitted 6 August 2020
Accepted 8 October 2020
Published 7 December 2020

Corresponding author
Chengbin Peng, pengchengbin@nbu.edu.cn

Academic editor
Faizal Khan

Additional Information and Declarations can be found on page 18

DOI **10.7717/peerj-cs.311**

INTRODUCTION

Helmets can play a vital role in protecting people. For example, many severe accidents in production and work sites and roads have been related to violations of wearing helmets. Some personnel may lack safety awareness in a working site and often do not or forget to wear helmets. On the road, craniocerebral injury is the leading cause of serious injury to cyclists in road traffic (*World Health Organization, 2006*). However, wearing a helmet reduces the risk of head injury of motorcycle riders by 69% (*Liu et al., 2008*), and wearing a helmet reduces the risk of head injury for cyclists by 63%–88% (*Thompson, Rivara & Thompson, 1999*).

Monitoring helmet-wearing manually can have many limitations, as people can be fatigue and costly. Reducing manual monitoring while ensuring that relevant personnel wearing helmets all the time in the working area has become an urgent problem.

Image recognition technology can reduce the workforce and material expenditures, and can significantly protect workers in many areas. Developments of computer vision algorithms and hardware (*Feng et al., 2019*) have paved the road for the application in helmet detection. Deep neural networks have gained much attention in image classification (*Krizhevsky, Sutskever & Hinton, 2017*), object recognition (*Donahue et al., 2013*), and image segmentation (*Garcia-Garcia et al., 2017*).

Previous computer vision algorithms for helmet detection are usually used in relatively simple scenes. For helmet detection, *Rubaiyat et al, (2016)* used a histogram of oriented gradient and a support vector machine to locate persons, then used a hough transform to detect helmet for the construction worker. *Li et al. (2017b)* identified helmets by background subtraction. *Li et al. (2017a)* used ViBe background modeling algorithm and human body classification framework C4 to select people and heads, and then identified whether people wore helmets through color space transformation and color feature recognition. However, such approaches are typically not suitable for complex scenes and dynamic backgrounds, such as construction sites, workshops, and streets.

Choudhury, Aggarwal & Tomar (2020) and *Long, Cui & Zheng (2019)* use single shot object detector algorithm to detect helmets. *Siebert & Lin (2020)* used RetinaNet which uses a multi-scale feature pyramid and focal loss to address the general limitation of one-stage detectors in accuracy, it works well in certain situations but its performance is highly scene dependent and influenced by light. *Bo et al. (2019)* use the You Only Look Once (YOLO) algorithm to accurately detect helmet wear in images with an average of four targets. However, most of these approaches are not suitable for both small and large helmets at the same time.

In this work, we propose a framework to integrate two complementary deep learning algorithms to improve the ability of helmet-wearing detection in complex scenes. Our approach is able to identify regular-size and tiny-size objects at the same time for helmet-wearing detection, and can be used for detection in complex scenes. This framework can outperform traditional approaches on benchmark data.

RELATED WORK

The starting point of CNN is the neurocognitive machine model (*Fukushima & Miyake, 1982*). At this time, the convolution structure has appeared. The classic LeNet (*LeCun et al., 1998*) was proposed in 1998. However, CNN's edge began to be overshadowed by models such as SVM (support vector machine) later. With the introduction of ReLU (Rectified Linear Units), dropout, and historic opportunities brought by GPU and big data, CNN ushered in a breakthrough in 2012: AlexNet (*Krizhevsky, Sutskever & Hinton, 2017*). In the following years, CNN showed explosive development, and various CNN models emerged. CNN has gradually gained the favor of scholars due to its advantages of not having to manually design features when extracting image features (*Shi, Chen & Yang, 2019*).

Many recent object detection approaches are based on RCNN (Region-based Convolutional Neural Networks) algorithms and YOLO algorithms (*Redmon et al., 2016*). RCNN is an improved algorithm based on CNN. Girshick et al. propose an epoch-making RCNN algorithm (*Girshick et al., 2014*) in the field of object detection. The central idea is to use a search selective method to extract some borders from the image. Then the size of the area divided by the border is normalized to the convolutional neural network input size, and then the SVM is used to identify the target. The bounding box of the target is obtained through a linear regression model. It brought deep learning and CNN to people's sight. However, there are disadvantages such as cumbersome training steps, slow test training speed, and large space occupation.

In order to improve the training and testing speed of RCNN, Fast RCNN algorithm (*Girshick, 2015*) was developed. It uses fewer layers while adding an ROI pooling layer to adjust the convolution area, and using softmax instead of the original SVM for classification. Compared with RCNN, Fast RCNN has improved training and testing speed. However, because the selective search method is also used to extract the borders of the region of interest, the speed of this algorithm is still not ideal for working with large data sets. Later, Faster RCNN (*Ren et al., 2015*) integrates feature extraction, proposal extraction, bounding box regression, classification, etc. into a network. The overall performance is far superior to CNN, and at the same time, it runs nearly much faster than CNN. Thus, Faster RCNN is commonly used in many applications. The Faster RCNN performs well for relatively large objects, but when detecting small faces or helmets, there will be a large false negative rate.

Tiny Face has made certain optimizations for small face detection. It mainly optimizes face detection from three aspects: the role of scale invariance, image resolution, and contextual reasoning. Scale invariance is a fundamental property of almost all current recognition and object detection systems, but from a practical point of view, the same scale is not applicable to a sensor with a limited resolution: the difference in incentives between a 300px face and a 3px face is undeniable (*Hu & Ramanan, 2017*). Ramanan et al. conducted an in-depth analysis of the role of scale invariance, image resolution, and contextual reasoning. Compared with mainstream technology at the time, the error rate can be significantly reduced (*Hu & Ramanan, 2017*).

Boosting algorithm was initially proposed as a polynomial-time algorithm, and the effectiveness has been experimentally and theoretically proved (*Schapire, 1990*). Afterward, Freund et al. improved the Boosting algorithm to obtain the Adaboost algorithm (*Freund & Schapire, 1997*). The principle of the algorithm is to filter out the weights from the trained weak classifiers by adjusting the sample weights and weak classifier weights. The weak classifiers with the smallest coefficients are combined into a final robust classifier.

In this work, in order to identify a variety of heads and helmets in complex scenes, we propose a framework to use incorporate multiple complementary deep learning algorithms to improve the joint performance.

MATERIALS & METHODS

Method

To address the helmet-wearing detection problem, we compare several object detection methods, such as the naive Bayes classifier, SVM, and Artificial Neural Networks classifier. Naive Bayes usually needs independent distributive premises. SVM is difficult to training for various scenes. In the case of a complex scene and huge training data, artificial neural networks are expected to have better accuracy and reliability, so we propose to use artificial neural networks, especially, convolutional neural networks, to solve this issue. To address the disadvantages raised by long-range cameras, we further improve the performance by integrating multiple complementary deep neural network models.

Base algorithms

Faster RCNN for detecting faces and helmet-wearing. After images are fed, Faster RCNN firstly extracts image feature maps through a group of basic conv+relu+pooling layer. Next, RPN (Region Proposal Networks) will set a large number of anchors on the scale of the original image, and randomly select 128 positive anchors and 128 negative anchors from all anchors for binary training, and use these anchors and a softmax function to initially extract positive anchors as the candidate area. At this time, the candidate regions are not accurate and require bounding boxes.

For a given image I, we use A to represent the ground-truth anchors. We use A_F and c_F to represent the identified bounding boxes and helmet-wearing confidence scores, respectively, computed by the Faster-RCNN algorithm. If we use F to represent the algorithm, W_F to represent the weight of the network, this approach can be written as follows.

$$A_F, c_F = F(I, W_F) \tag{1}$$

If we consider $A_F = F(I, W_F)[0]$ and $c_F = F(I, W_F)[1]$, we can use

$$Loss(F(I, W_F)[0], F(I, W_F)[1], A) \tag{2}$$

to represent the loss function (*Fukushima & Miyake, 1982*) when to minimize differences between the detected anchors and the ground-truth.

Thus, when we train this model, the optimization is as follows.

$$W_F^* = \text{argmin}_{W_F} Loss(F(I, W_F)[0], F(I, W_F)[1], A) \tag{3}$$

Tiny Face for detecting faces. The overall idea of Tiny Face is similar to RPN in Faster RCNN, which is a one-stage detection method. The difference is that some scale specific design and multi-scale feature fusion are added, supplemented by image pyramid so that

the final detection effect for small faces is better. The training data were changed by three scales, with one-third of the probability respectively and sent to the network for training. Multiple scales could be selected to improve the accuracy rate in the prediction.

For a given image I, we can also use A_T and c_T to represent the identified bounding boxes and confidence scores computed by the Tiny Face algorithm, so if we use T to represent the Tiny Face algorithm and W_T to represent the corresponding weight, we can have

$$A_T, c_T = T(I, W_T) \tag{4}$$

However, Tiny Face can only be used to determine whether the detection target contains a human face and cannot directly distinguish whether the target is wearing a helmet. Thus, we propose to use CNN to overcome this disadvantage.

CNN for detecting helmet-wearing. For anchors determined by Tiny Face, we can use a CNN to detect helmets above the face. We enlarge the face area detected by the Tiny Face and feed it into the CNN model for prediction. The confidence scores indicating whether there is a helmet above the face can be computed by the CNN algorithm

$$c_C = C(A_T, I, W_C), \tag{5}$$

where C is a function representing the forward propagation of CNN. Here, C is a composition of two convolution layers and one fully connected layer.

The loss function is again to minimize the difference between detected helmets and the ground-truth

$$Loss(A_T, C(A_T, I, W_C), A). \tag{6}$$

Ensemble model detecting high and low resolution helmets

For the two lists of face anchors A_F and A_T detected by the base algorithms above, we merge them with the following strategy. We first initialize an empty anchor list A_S and two score vector c_{SF} and c_{SC}.

For the ith anchor in A_F and the corresponding score in c_F, namely, A_F[i] and c_F[i], we first insert them into A_S and c_{SF} respectively. Then A_F[i] is compared with all the anchors in A_T. If some anchors in A_T have more than 60% overlapping area with A_F[i], we remove these anchors in A_T and remove the corresponding entries in c_C. We also take the mean value of the removed entries in c_C and insert it into c_{SC}. If no overlapped anchors in A_T is found, we insert zero into c_{SC}.

After all the anchors in A_F in processed, the remaining anchors in A_T, the remaining confidence values in c_C, and a zero vector of the same length is inserted into A_S, c_{SC}, and c_{SF}, respectively. At last, we compute the covering area of each anchor in A_S and store them in δ.

The pseudocode of the merge process can be described as follows.

```
def merge(A_F,A_T,c_F,c_C):
    A_s = [ ]
        c_SF = [ ]
        c_SC = [ ]
    for i in (0,size(A_F)):
        append A_F[i] to A_s
        append c_F[i] to c_SF
        idx = idx of all the anchors in A_T, each of which has at least 60% overlapping with
    A_F[i].
        if idx == [ ]:
            append avg(c_C[idx]) to c_SC
            A_T[idx]=[ ] % remove corresponding entries
            c_C[idx]=[ ]
        else:
            append zero to c_SC

    append A_T to A_s
    append c_C to c_SC
    append zeros to c_SF
    compute covering area of each anchor in A_s and store into δ
    return A_s,δ_S,c_SF,c_SC
```

Algorithm 1. definition of function merge()

After the data preparation, many ensemble learning methods can be used for model integration. In this work, we consider a basic ensemble model defined as follows

$$S(c_{SF}, c_{SC}, \delta, \alpha) = \sum_i \alpha_i h_i(c_{SF}, c_{SC}, \delta) \tag{7}$$

where α is the model parameter, δ is a vector containing the area of corresponding anchors, and $h_i()$ is a classifier. We choose decision trees with maximum depth of two in the experiment, and i is ranged from 0 to 1000. The variable δ is used here because the two base algorithms are good at identifying relatively large and small objects respectively, and adding covering areas of anchors can help improve the accuracy.

Thus, in the ensemble method, A_S is the anchor lists, and $c_S = S(c_{SF}, c_{SC}, \delta, \alpha)$ contains the corresponding confidence values about helmet-wearing. To train this model, we merge the anchor set A_S and the ground-truth set A in a similar manner as merging A_F and A_T, and we use $\hat{c}_{SF}$, $\hat{c}_{SC}$ and $\hat{c}$ to represent the corresponding variables after merging. Zeros are filled if the corresponding anchor does not exist before merging. Then, the loss between the identified anchors in A_S and the ground-truth anchors A is

$$E(\delta, \alpha, \hat{c}_{SF}, \hat{c}_{SC}, \hat{c}) = \sum_{i=0}^{n} (S(\hat{c}_{SF}[i], \hat{c}_{SC}[i], \delta, \alpha) - \hat{c}[i])^2 \tag{8}$$

where n is the total anchors after merging. The optimal value of α can be computed by minimizing the error

$$\alpha^* = \text{argmin}_\alpha E(\alpha, \hat{c}_{SF}, \hat{c}_{SC}, \hat{c}) \tag{9}$$

The whole process can be described by the pseudocode below.

% Training
Training data: images I, ground-truth anchors A, ground-truth confidence c .
% Training base models
$W_F^* = \text{argmin}_{W_F} \text{Loss}(F(I,W_F)[0], F(I,W_F)[1], A)$.
$A_F, c_F = F(I,W_F^*)$ % Detect helmet-wearing with Faster-RCNN
$A_T, c_T = T(I, W_T^*)$ % Detect faces with Tiny Face
$W_T^* = \text{argmin}_{W_T} \text{Loss}(A_T, C(A_T,I, W_C), A)$.
$c_C = C(A_T,I, W_C^*)$ % Detect helmet-wearing with CNN
$A_s,\delta_S,c_{SF},c_{SC} = \text{merge}(A_F,A_T,c_F,c_C)$

% Training the ensemble method.
$\hat{A},\hat{\delta},[\hat{c}_{SF}, \hat{c}_{SC}],\hat{c} = \text{merge}(A_S,A,[c_{SF},c_{SC}], c)$
$\alpha^* = \text{argmin}_\alpha E(\hat{\delta}, \alpha,\hat{c}_{SF},\hat{c}_{SC},\hat{c})$

% Testing
Testing data: images I'.
$A_F, c_F = F(I',W_F^*)$
$A_T, c_T = T(I', W_T^*)$
$c_C = C(A_T,I', W_C^*)$
$A_s,\delta_S,c_{SF},c_{SC} = \text{merge}(A_T,A_F,c_F,c_C)$ % Integrate detection results
$c_S = S(c_{SF},c_{SC},\delta_S,\alpha^*)$ % Compute confidence scores
A_s is the anchor list, and c_S is the corresponding confidence vector

---

Algorithm 2. pseudocode of the whole framework

Experiments

In order to evaluate the performance of our framework, we use five criteria:

$$TPR = m/n \tag{10}$$

$$FPR = l/k \tag{11}$$

$$RE = m/N \tag{12}$$

$$FNR = 1 - RE \tag{13}$$

$$PRE = m/(m+l) \tag{14}$$

where TPR is the true positive rate, FPR is the false positive rate, FNR is the false negative rate, RE is the recall rate, PRE is the precision rate, m is the correct prediction by models under the current threshold, n is the number of parts of the model detection result that are identical to the truth ground, l is the false prediction by models under the current threshold, k is the number of parts of the model detection result that are different from the truth ground, and N is the number of targets that actually exist.

Figure 1 **Faster RCNN detecting big faces.**
Full-size DOI: 10.7717/peerjcs.311/fig-1

To evaluate our approach, we take the publicly available benchmark data set (*Safety Helmet Wearing-Dataset,*), containing images from construction sites, roads, workshops, and classrooms. The data set consists of a total of 7,581 images. We use five-fold cross validations for experiments. We randomly divide all the images into five parts. Training set, validation set, and testing set contains 3/5, 1/5, and 1/5 of the total images respectively.

Preliminary analysis

The detection results of Faster RCNN for faces are shown in Figs. 1 and 2. From these two figures, we can see that Faster-RCNN is suitable for detecting large objects, but not finding small ones.

The detection results of Tiny Face are shown in Fig. 3. From this result, we can see that Tiny Face is good at finding small faces.

To compare the differences between the two models, we used Faster RCNN and Tiny Face to test the 1000 images from the data set, and count the number of faces of different sizes detected by the two models. Figure 4 is the histogram of real data, and Fig. 5 is the histogram of face sizes detected by Faster RCNN.

Taking the number of pixels (px^2) as the area measurement, a face with an area smaller than $500px^2$ is defined as a small face, and a face larger than $500px^2$ is defined as a large face. Because of the large area span, the smallest face is only $90px^2$, while the largest face can reach $2000000px^2$. In order to prevent the histograms from crowding together, only faces with an area less than 2000px are shown in the figure.

Figure 2 Faster RCNN detecting small faces.

Full-size DOI: 10.7717/peerjcs.311/fig-2

Figure 3 Tiny Face detecting small faces.

Full-size DOI: 10.7717/peerjcs.311/fig-3

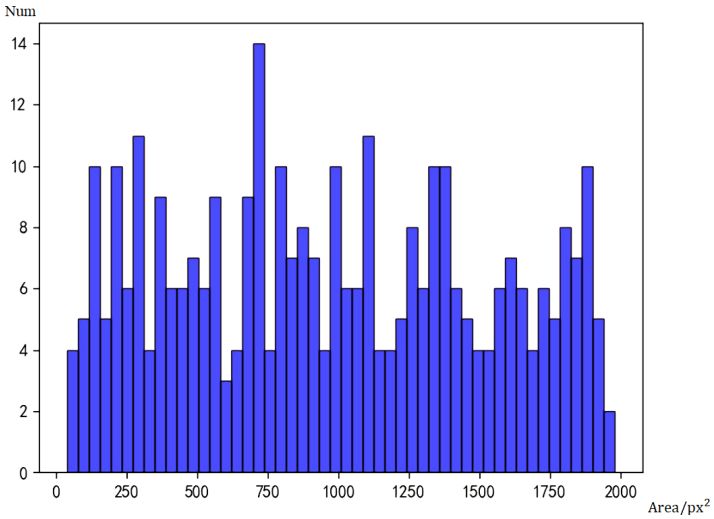

Figure 4 **Histogram of real data.**

Full-size DOI: 10.7717/peerjcs.311/fig-4

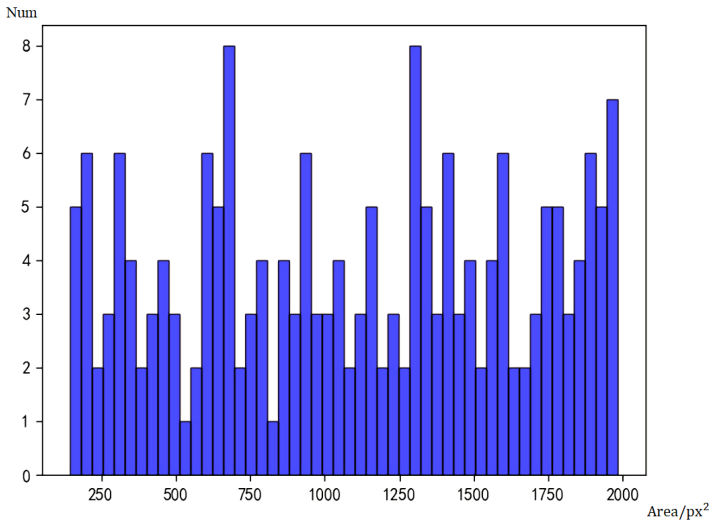

Figure 5 **Histogram of face sizes detected by Faster RCNN.**

Full-size DOI: 10.7717/peerjcs.311/fig-5

According to statistics, there are actually 1,568 big faces and 83 small faces. The initial model of Faster RCNN can detect 1,468 big faces and 37 small faces. Under the assumption that the labels are correct, the false negative rate of big faces is 5.2%, and that of small faces is 55.5%. Obviously, the Faster RCNN model has lower accuracy for small faces.

Then we performed statistics on Tiny Face and got the histogram of Tiny Face detection results in Fig. 6. Tiny Face can detect 1306 large faces and 44 small faces. The false negative rate for large faces is 16.8%, which is 11.6% higher than Faster RCNN, and the false negative rate for small faces is 47.0%, which is 8.5% lower than Faster RCNN. Although it is only a preliminary model, the model has not been adjusted and the amount of training has been

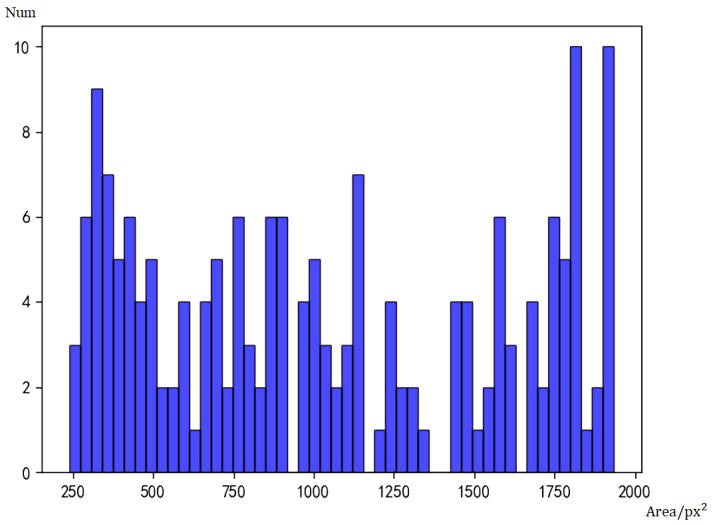

Figure 6 **Histogram of face sizes detected by Tiny Face.**
Full-size DOI: 10.7717/peerjcs.311/fig-6

adjusted to improve the accuracy of the model, but it is not difficult to see from the current data that the detection capabilities of the Faster RCNN and Tiny Face models have their own focus. When Faster RCNN detects large faces, it has a great advantage, and Tiny Face's ability to detect small faces is better than Faster RCNN.

We can find that Faster RCNN has a higher true positive rate for detecting large faces and Tiny Face has a higher true positive rate for detecting small faces. The overall effect can be better if we can combine the two methods.

Accuracy of base algorithms for helmet detection

Accuracy of $F(I, W_F^*)$. In this part, we evaluate the accuracy of $T(I, W_T^*)$ alone in Algo. 2. Theoretically, the more training steps the model has, the better, but in order to prevent overfitting, we still need to observe the accuracy of the model under different training steps.

In the beginning, we select some images from training dataset to evaluate the model. We trained 5,000 steps and used the model to test the images of the training set, but it was obvious that the effect was not very satisfactory. Because $T(I, W_T^*)$ is based on Faster RCNN, which has high accuracy, it is easy to miss the mark of small faces. Therefore, the quality of the model can be preliminarily judged by the number of detected targets, and then we gradually increased the number of training steps.

When the number of training steps reaches 20,000 steps, the number of detected targets in the detection results of 1,000 test set images is basically maintained at about 1,300. As the number of training steps increases, the number of detected targets increases slightly. When the number reaches 60,000 steps, the number of detected targets is 1,523. At this time, precision rate of the model is 87.3%,and the recall rate is 85.9%. When the number of training steps reaches 70,000 steps, the number of detected targets is close to 1,700. At this time, the precision rate of the model is 81.2%, and the recall rate is 86.3%. We find that

Table 1 Relationship between training steps and accuracy.

Steps	Precision rate	Recall rate
5,000	80.0%	72.4%
20,000	84.0%	82.0%
40,000	86.1%	85.1%
60,000	87.3%	85.9%
70,000	81.2%	86.3%

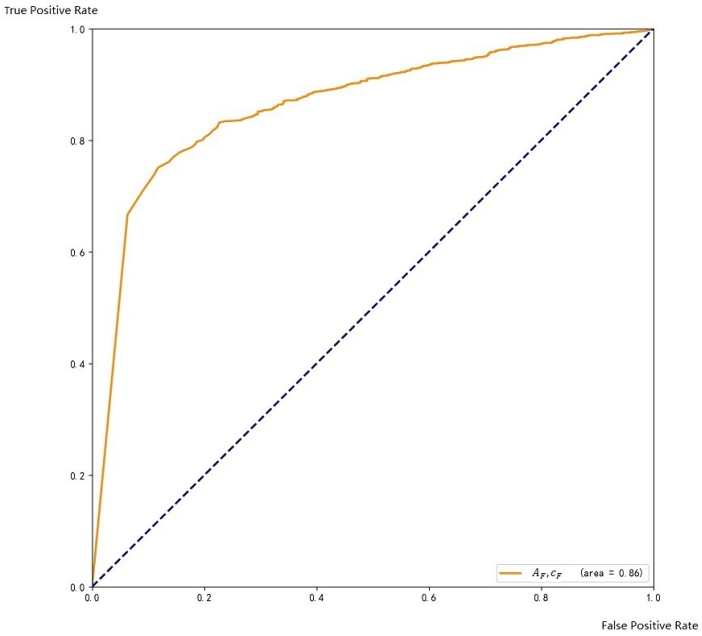

Figure 7 ROC with respect to A

Full-size DOI: 10.7717/peerjcs.311/fig-7

although the recall rate has a slight increase, but the precision rate is much lower, so we chose the model with 60,000 training steps as the final model. See Table 1 for the accuracy of $F(I, W_F^*)$ under different training steps.

Regarding to the scoring threshold, it is 0.5 by default, which means that when the score is lower than 0.5, the result will be discarded. We successively set the threshold to 0.3, 0.4, 0.5, 0.6, 0.7, and tested the validation data to choose the one that works best. Finally, we found that when the threshold is 0.6, the precision rate of the test result is 87.3%, and the recall rate is 85.9%, which is better than other thresholds. After comprehensive consideration, we finally keep 0.6 as the threshold for the ensemble. The ROC curve on the training set is shown in Fig. 7.

When training this model, in order to distinguish whether an individual wearing a helmet, we use two labels: people wearing and without wearing a helmet. It makes the final trained model more accurately distinguish whether the target wears a helmet.

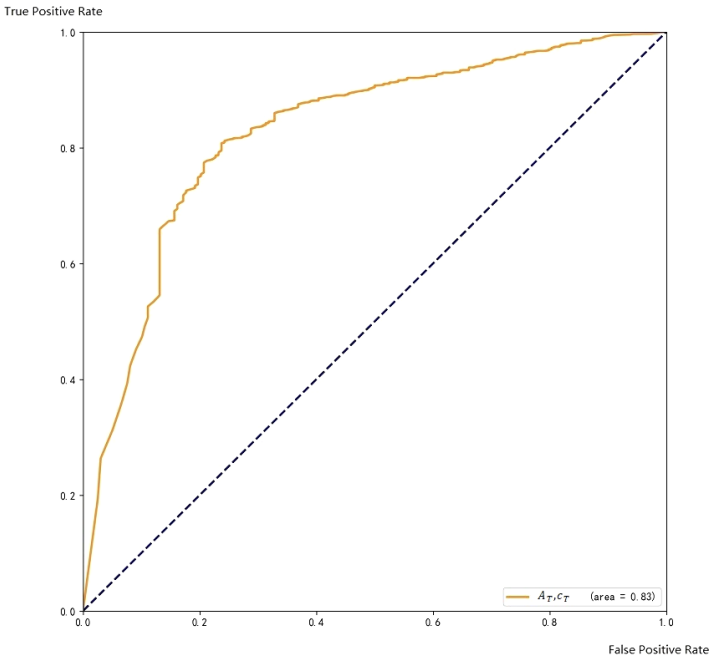

Figure 8 **ROC with respect to A_T and c_C.**
Full-size DOI: 10.7717/peerjcs.311/fig-8

Accuracy of $T(I, W_T^*)$. In this part, we consider the accuracy of $T(I, W_T^*)$ alone in Algo. 2. It is basically a trained Tiny Face model. We lowered the scoring threshold of Tiny Face to 0.5, requiring the Tiny Face model to be able to determine the location of the small face, and it does not need it to accurately return the scoring value. The precision rate of face detection was 85.6%, and the recall rate was 69.4%.

Accuracy of $C(A_T, I, W_C^*)$. In this part, we consider the accuracy of $C(A_T, I, W_C^*)$ alone in Algo. 2. Function $C(A_T, I, W_C^*)$ is basically a CNN model, which requires only one target in an image, so we select over 2,000 images from the training set, cropped the target according to the corresponding anchor labels and get 20,000 images with only one target in each image. We select 18,000 images as training data for CNN, and the other 2,000 images as a validation set to detect the accuracy of CNN, also the cropped images are divided into two sets, people wearing helmets and without wearing helmets. In addition, we rotate some images to get richer training samples.

With cross-validation, we choose to use four pairs of convolution and pooling layers, of which the first layer and the size of the convolution kernel of the second convolution layer are [5,5], and the size of the convolution kernel of the third and fourth convolution layers is [3,3]. The precision rate of the final two-class CNN reached 90.3% when we use it to test the validation set of CNN.

The ROC curve on the training set is shown in Fig. 8.

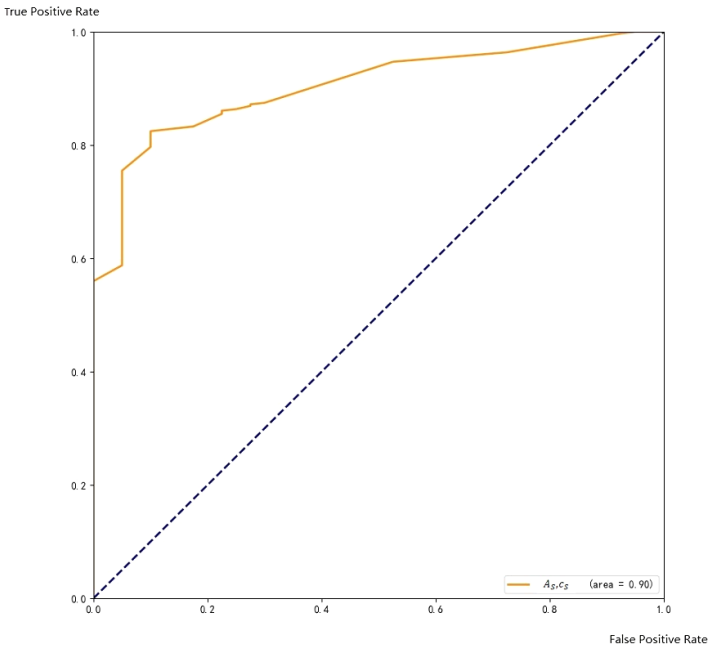

Figure 9 **ROC with respect to A_S and c_S <!–[if !msEquation]–> <!–[endif]–>.**
Full-size DOI: 10.7717/peerjcs.311/fig-9

Accuracy of Ensemble Method $S(c_{SF}, c_{SC}, \alpha^*)$. The ROC curve coverage areas of Faster CNN and A_T, c_C are 0.86and 0.83, respectively. The ensemble method can further improve the accuracy of the final result.

Among the data, c_F and c_C are the results from two base methods, respectively, and the area is the size of the target frame. Obviously, c_F and c_C can be used as the characteristic values of the ensemble method. We test the trained model, and the area under the ROC curve coverage is larger, becoming 0.90. The ROC curve on the training set is shown in Fig. 9.

Obviously, the ROC curve covered by the ensemble method has the largest coverage area, which proves that the ensemble method is effective in our model.

Comparison of different algorithms. In this part, we demonstrate the effectiveness of our ensemble framework by combining Faster RCNN and Tiny Face+CNN together with ROC curve and PR curve. The ROC and PR curves are calculated from testing results through 5-fold cross-validation as Figs. 10 and 11. From Figs. 10 and 11, we can see that combination with our framework (in green) is better than single algorithms (in black and orange). Our framework can also gain the largest area under the ROC curve (0.83) in Fig. 10, and the largest area under the PR curve (0.93), namely the mAP score. It means our framework works best on average over all the possible threshold choices.

Tables 2 and 3 reveals a similar phenomenon when a reasonable threshold is chosen. It indicates that, with a well-chosen threshold, our framework works better than others in terms of TPR, FPR,FNR, precision, and recall.

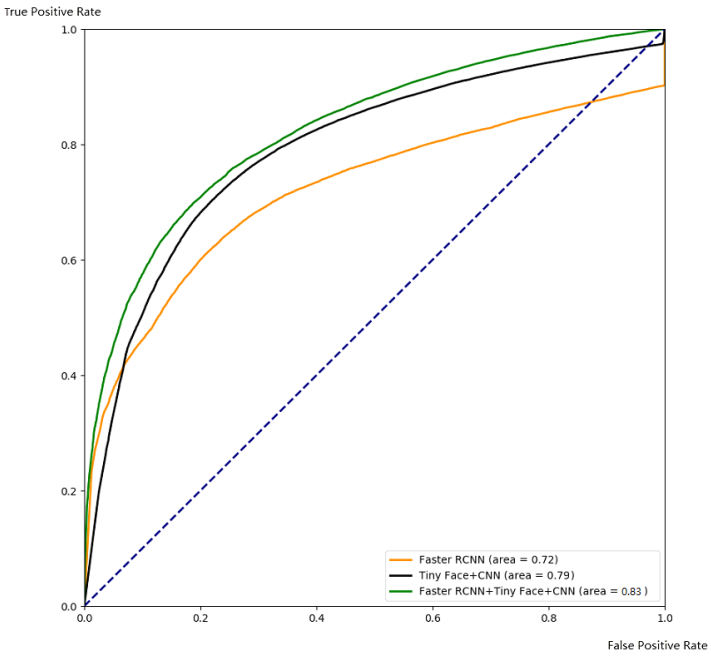

Figure 10 **Comparison with ROC.**

Full-size DOI: 10.7717/peerjcs.311/fig-10

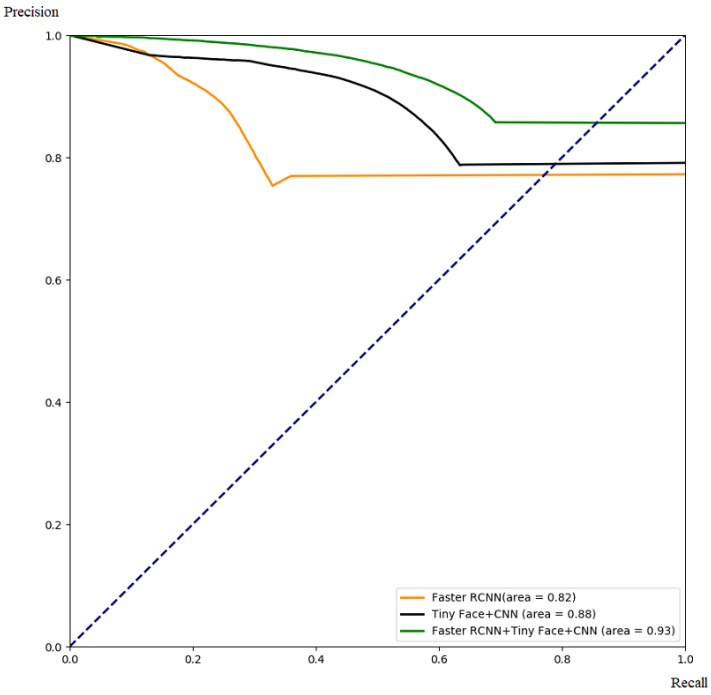

Figure 11 **Comparison with PR.**

Full-size DOI: 10.7717/peerjcs.311/fig-11

Table 2 Comparison with TPR, FPR, FNR.

Algorithm	True positive rate	False positive rate	False negative rate
Faster RCNN	74.7%	43.1%	72.7%
TinyFace + CNN	73.8%	25.8%	51.9%
Faster RCNN + Tiny Face + CNN	75.6%	18.3%	42.5%

Table 3 Comparison with precision and recall.

Algorithm	Precision	Recall
Faster RCNN	85.4%	27.3%
Tiny Face + CNN	91.5%	48.1%
Faster RCNN + Tiny Face + CNN	92.5%	57.5%

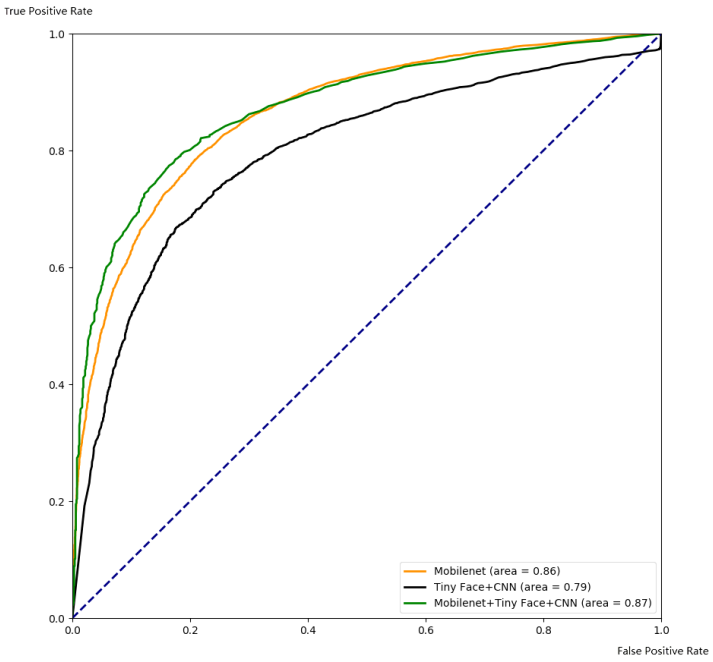

Figure 12 Comparison with ROC for integrating Mobilenet and Tiny Face.
Full-size DOI: 10.7717/peerjcs.311/fig-12

Our framework can also be used to integrate other complementary deep learning methods to improve their performance. As an example, we use our framework to combine Mobilenet and TinyFace+CNN, and compare the integrated results with single algorithms. The performance is shown in Figs. 12 and 13. Similar to the previous case, the algorithm performance is generally improved. Our framework also works well when a specific threshold is chosen, as shown in Tables 4 and 5.

Through these experiments, we can find that the integrated framework for two complementary models can improve the performance of single algorithms by increasing

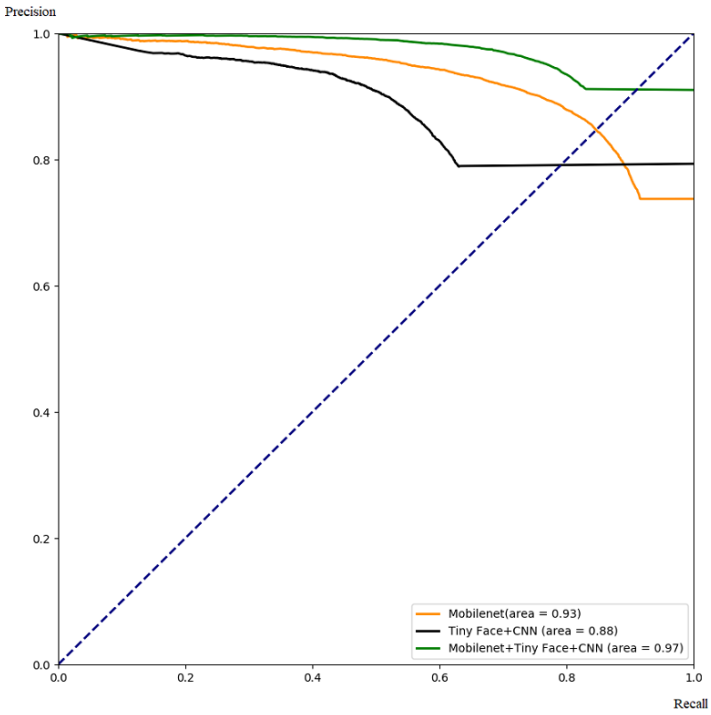

Figure 13 **Comparison with PR for integrating Mobilenet and Tiny Face.**
Full-size DOI: 10.7717/peerjcs.311/fig-13

Table 4 **Comparison with TPR, FPR, FNR for integrating Mobilenet and Tiny Face.**

Algorithm	True positive rate	False positive rate	False negative rate
Mobilenet	74.3%	17.4%	32.0%
TinyFace + CNN	73.3%	25.2%	52.5%
Mobilenet + Tiny Face + CNN	80.0%	17.2%	35.2%

Table 5 **Comparison with Precision and Recall for integrating Mobilenet and Tiny Face.**

Algorithm	Precision	Recall
Mobilenet	92.0%	69.4%
Tiny Face + CNN	91.9%	47.7%
Mobilenet + Tiny Face + CNN	94.7%	77.7%

the true positive rate, the precision rate, and the recall rate, while reducing the false positive rate and false negative rate.

DISCUSSION

The detection accuracy of a single model is usually not a satisfactory, so we use an ensemble method to integrate models to get better results. Considering complementary behaviors of different algorithms, using an ensemble method for integration can effectively improve the accuracy of the detection results. For example, through our experiments, we can use

Tiny Face model with CNN to overcome the shortcomings that the Faster RCNN model possesses when detecting small faces. Although the proportion of small faces in the test set of this experiment is not very large, the missing rate is still one percent lower than that of a single model. In the test set with a large proportion of small faces, the detection accuracy of the integrated model can be improved further.

CONCLUSION

When the detection accuracy of a single deep learning model could not meet the demand for helmet-wearing detection, we can integrate a complementary model with it to get better results. In addition, our framework can make single algorithms more robust to data sets from different scenarios, because it can utilize the advantage of the complementary algorithms.

By analyzing a variety of object detection models, we find that many models are difficult to achieve high-precision for helmet-wearing detection in different scenarios. Therefore, we carefully select two complementary base models and add additional modules to make them suitable for helmet-wearing detection. We ensemble the base models and build a more powerful helmet-wearing detection algorithm to further improve the detection capability. Our approach can be accelerated by GPU and be deployed on distributed computers to reduce processing time, and thus, can be useful in real-world scenarios. In the future, the model can also be extended by integrating additional features or models and upgraded to mixed neural network models.

ADDITIONAL INFORMATION AND DECLARATIONS

Funding

This work was supported by the National Natural Science Foundation of China (NO. 61802372), the Natural Science Foundation of Zhejiang Province (NO. LGG20F020011), the Ningbo Science and Technology Innovation Project (NO. 2018B10080), and the Qianjiang Talent Plan (NO. QJD1702031). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Grant Disclosures

The following grant information was disclosed by the authors:
National Natural Science Foundation of China: 61802372.
Natural Science Foundation of Zhejiang Province: LGG20F020011.
Ningbo Science and Technology Innovation Project: 2018B10080.
Qianjiang Talent Plan: QJD1702031.

Competing Interests

The authors declare there are no competing interests.

Author Contributions

- Zheming Fan and Chengbin Peng conceived and designed the experiments, performed the experiments, analyzed the data, performed the computation work, prepared figures and/or tables, authored or reviewed drafts of the paper, and approved the final draft.
- Licun Dai conceived and designed the experiments, performed the experiments, analyzed the data, performed the computation work, authored or reviewed drafts of the paper, and approved the final draft.
- Feng Cao, Jianyu Qi and Wenyi Hua performed the experiments, performed the computation work, authored or reviewed drafts of the paper, and approved the final draft.

Data Availability

The following information was supplied regarding data availability:

Code is available as a Supplemental File.

The data set is available at GitHub: https://github.com/njvisionpower/Safety-Helmet-Wearing-Dataset.

Supplemental Information

Supplemental information for this article can be found online at http://dx.doi.org/10.7717/peerj-cs.311#supplemental-information.

REFERENCES

Anonymous. 2020. Safety helmet wearing-dataset. *Available at https://github.com/njvisionpower/Safety-Helmet-Wearing-Dataset* (accessed on 3 August 2020).

Bo Y, Huan Q, Huan X, Rong Z, Hongbin L, Kebin M, Weizhong Z, Lei Z. 2019. Helmet detection under the power construction scene based on image analysis. In: *2019 IEEE 7th international conference on computer science and network technology (ICCSNT)*. Piscataway: IEEE, 67–71.

Choudhury T, Aggarwal A, Tomar R. 2020. A deep learning approach to helmet detection for road safety. *Journal of Scientific and Industrial Research* **79(06)**:509–512.

Donahue J, Jia Y, Vinyals O, Hoffman J, Zhang N, Tzeng E, Darrell T. 2013. *A deep convolutional activation feature for generic visual recognition*. Berkeley: UC Berkeley & ICSI.

Feng X, Jiang Y, Yang X, Du M, Li X. 2019. Computer vision algorithms and hardware implementations: a survey. *Integration* **69**:309–320 DOI 10.1016/j.vlsi.2019.07.005.

Freund Y, Schapire RE. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* **55(1)**:119–139 DOI 10.1006/jcss.1997.1504.

Fukushima K, Miyake S. 1982. Neocognitron: a self-organizing neural network model for a mechanism of visual pattern recognition. In: *Competition and cooperation in neural nets*. Berlin, Heidelberg: Springer, 267–285.

Garcia-Garcia A, Orts-Escolano S, Oprea S, Villena-Martinez V, Garcia-Rodriguez J. 2017. A review on deep learning techniques applied to semantic segmentation. ArXiv preprint. arXiv:1704.06857.

Girshick R. 2015. Fast r-cnn. In: *Proceedings of the IEEE international conference on computer vision*. Piscataway: IEEE, 1440–1448.

Girshick R, Donahue J, Darrell T, Malik J. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 580–587.

Hu P, Ramanan D. 2017. Finding tiny faces. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 951–959.

Krizhevsky A, Sutskever I, Hinton GE. 2017. Imagenet classification with deep convolutional neural networks. *Communications of the ACM* **60(6)**:84–90.

LeCun Y, Bottou L, Bengio Y, Haffner P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* **86(11)**:2278–2324 DOI 10.1109/5.726791.

Li K, Zhao X, Bian J, Tan M. 2017a. Automatic safety helmet wearing detection. In: *2017 IEEE 7th annual international conference on CYBER technology in automation, control, and intelligent systems (CYBER)*. Piscataway: IEEE, 617–622.

Li J, Liu H, Wang T, Jiang M, Wang S, Li K, Zhao X. 2017b. Safety helmet wearing detection based on image processing and machine learning. In: *2017 Ninth International Conference on Advanced Computational Intelligence (ICACI)*. Piscataway: IEEE, 201–205.

Liu BC, Ivers R, Norton R, Boufous S, Blows S, Lo SK. 2008. Helmets for preventing injury in motorcycle riders. *Cochrane Database of Systematic Reviews* **1**:CD004333 DOI 10.1002/14651858.CD004333.pub3.

Liu XH, Ye XN. 2014. Skin color detection and hu moments in helmet recognition research. *Journal of East China University of Science and Technology* **3**:365–370.

Long X, Cui W, Zheng Z. 2019. Safety helmet wearing detection based on deep learning. In: *2019 IEEE 3rd information technology, networking, electronic and automation control conference (ITNEC)*. Piscataway: IEEE, 2495–2499.

Redmon J, Divvala S, Girshick R, Farhadi A. 2016. You only look once: unified, real-time object detection. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. Piscataway: IEEE, 779–788.

Ren S, He K, Girshick R, Sun J. 2015. Faster r-cnn: towards real-time object detection with region proposal networks. In: *Advances in neural information processing systems*. 91–99.

Rubaiyat AH, Toma TT, Kalantari-Khandani M, Rahman SA, Chen L, Ye Y, Pan CS. 2016. Automatic detection of helmet uses for construction safety. In: *2016 IEEE/WIC/ACM International Conference on Web Intelligence Workshops (WIW)*. Piscataway: IEEE, 135–142.

Schapire RE. 1990. The strength of weak learnability. *Machine Learning* **5(2)**:197–227.

Shi H, Chen X, Yang Y. 2019. Safety helmet wearing detection method of improved YOLOv3. *Computer Engineering and Applications* **55**:213–220 DOI 10.3778/j.issn.1002-8331.1811-0389.

Siebert FW, Lin H. 2020. Detecting motorcycle helmet use with deep learning. *Accident Analysis & Prevention* **134**:105319 DOI 10.1016/j.aap.2019.105319.

Thompson DC, Rivara F, Thompson R. 1999. Helmets for preventing head and facial injuries in bicyclists. *Cochrane Database of Systematic Reviews* **1992(2)**:CD001855 DOI 10.1002/14651858.CD001855.

World Health Organization. 2006. *Helmets: a road safety manual for decision-makers and practitioners*. Geneva: World Health Organization.

Yunbo LIU, Huang H. 2015. Research on monitoring of workers' helmet wearing at the construction site. *Electronic Science and Technology* **28(4)**:69–72.