applied sciences

MDPI

Article

A Practical App for Quickly Calculating the Number of People Using Machine Learning and Convolutional Neural Networks

Ching-Ta Lu [1,2,*], Chun-Jen Ou [1] and Yen-Yu Lu [1]

[1] Department of Information Communication, Asia University, Taichung City 41354, Taiwan; asia103024024@gmail.com (C.-J.O.); lulujq0924@gmail.com (Y.-Y.L.)
[2] Department of Medical Research, China Medical University Hospital, China Medical University, Taichung City 40402, Taiwan
* Correspondence: lucas@asia.edu.tw

Featured Application: The proposed app can help quickly calculate the number of people, avoid crowd gathering, and cause the risk of group infections for COVID-19.

Abstract: Calculating the number of people is often necessary and repeated in real life. As the number of people increases, the calculation is time-consuming. Efficiently calculating the number of people is helpful to human life. In this article, we propose a valuable app to quickly calculate the number of people in a photo by a convolutional neural network (CNN). Initially, suspected face areas are segmented into micro-blocks. The segmented blocks are then confirmed through the CNN by rejecting the segmented micro-blocks without the human face to ensure the detection accuracy of the face area. The experimental results reveal that the proposed app can efficiently calculate the number of people. The world is now seriously threatened by the COVID-19 epidemic. The proposed app can help quickly calculate the number of people, avoid crowd gathering, and cause the risk of group infections.

Keywords: convolutional neural network; deep learning; people counter; face detection; face segmentation

Citation: Lu, C.-T.; Ou, C.-J.; Lu, Y.-Y. A Practical App for Quickly Calculating the Number of People Using Machine Learning and Convolutional Neural Networks. *Appl. Sci.* **2022**, *12*, 6239. https://doi.org/10.3390/app12126239

Academic Editors: Shoou-Jinn Chang, Sheng-Joue Young and Liang-Wen Ji

Received: 20 May 2022
Accepted: 18 June 2022
Published: 19 June 2022

1. Introduction

Calculating the number of people is needed in daily life, such as a tour guide to confirm whether all tour members have arrived at each checkpoint. The checking is repeated again and again during traveling. Another example is calculating the number of people entering a recreation area, a museum, et al. Calculating the number of students attending class is repeated by teachers. If the number of people is large, the calculation process is time-consuming. Accordingly, providing an efficient method to calculate people's numbers brings great benefits for many applications. Additionally, face detection and recognition can be regarded as an efficient method for calculating the number of people.

There are two main methods for face detection. The first type is pattern comparison, which is also the most commonly used method. Many standard face images should be provided and stored in a database. When a test image is asked for comparison, it is compared with the database's images for similarity checking to define whether it is an actual human face. Nevertheless, this method requires an extensive database and much comparison time. The second type is the rule-based detection method. The rules should be rich enough for obtaining good performance. However, this method is hard to define a robust threshold for each rule in face detection. If the rule's threshold is too strict, it causes this method to not be able to recognize many human faces. In contrast, segmented blocks without the human face will cause false alarms in face area detection when the threshold is too loose.

Many face detection applications have been developed [1–13]. In the cases of statistical models, Tuncer et al. [1] proposed using the support vector machine and K-nearest neighbors methods for face recognition, where perceptual hash was used for feature extraction. Lin et al. [2] proposed using nonlinear least squares to facial landmarks for estimating the poses. Face recognition accuracy is improved by recognizing each image set's frontal image instead of using a fixed 3D model. Chakraborty et al. [3] proposed using a hand-crafted cascaded asymmetric local pattern to retrieve and recognize facial images. The method encodes the relationship among the neighboring pixels in horizontal and vertical directions, enabling the encoding method to obtain optimum feature length. Thus, the performance of face recognition is improved. Hssayni and Ettaouil [4] proposed using a co-occurrence matrix to extract the face information in an image, where several coefficients can represent the face information. Hence, a Bayesian neural network is employed to recognize the face. In the cases of machine learning [5,10,14–16], Viola and Jones [5] used three critical methods for face detection. This study proposed using a machine learning approach to visual object detection. An integral image is utilized as a feature. Because the Viola and Jones (VJ) algorithm brings many advantages to visual objection detection, many studies have been proposed to improve the accuracy of the Viola and Jones method. However, these methods are sensitive to parameter settings, enabling the detection performance to vary significantly. Artificial intelligence (AI) technologies are progressively applied in various aspects. Most face detection methods work well for a few face images, but the performance may be greatly degraded for a large number of face images. Sapijaszko and Mikhael [16] proposed using the 2D discrete wavelet transform and 2D discrete cosine transform to obtain texture features. A multilayer-sigmoid neural network recognizes the faces according to these features.

Recently, convolutional neural networks (CNNs) have been widely used in image recognition [7–23]. Wu et al. [7] proposed a multi-task CNN for face detection and head pose estimation by extracting more representative features. Zhang et al. [8] proposed deep cascaded multi-task convolutional neural networks that exploited the inherent correlation between detection and alignment to boost face detection performance. An online hard sample mining strategy was also applied. Li et al. [9] combined the CNN, a blur-aware bi-channel network, and a self-learning mechanism for exploiting video contexts continuously and face detection. Yang et al. [11] used a CNN for face detection by scoring facial parts responses according to the spatial structure and arrangement. The scoring mechanism is data-driven. Their method can detect faces under severe occlusion and unconstrained pose variations. Ranjan et al. [13] proposed an algorithm for simultaneous face detection, landmarks localization, pose estimation, and gender recognition using CNN. They fused the intermediate layers of a deep CNN using a separate CNN followed by a multi-task learning algorithm that operates on the mixed features, enabling the performance to be boosted. Ramya et al. [17] proposed using canonical correlation analysis to fuse the local face pattern extracted by the AlexNet and a shallow CNN. A multi-support vector machine then classifies the emotion category, where the accuracy rate can reach 87.69%.

Based on the above discussions, the methods in [8,13,17,24] use a deep cascaded multi-task framework and complex network for multi-function target recognition. The computational duty is high, while the size of the trained net is large. Loading the net for recognition is time-consuming. In this study, we attempt to develop a light app for quickly calculating the number of people, while loading the trained net requires little time. Therefore, the app is lightweight and practical. The VJ algorithm can effectively segment faces using machine learning in a photo. Additionally, CNN can effectively recognize the faces in a photo. In this paper, we propose an automatic and simple app to calculate the number of people in a photo by integrating the VJ algorithm and a face CNN. Initially, the VJ algorithm is employed to segment the faces as blocks in a photo. In this stage, all face-like blocks should be segmented out by the VJ algorithm, no matter how high the false detection rate is, so the miss detection of the face blocks is shallow. These segmented blocks are utilized for training a CNN for face confirmation. The face blocks detected by the VJ

algorithm are refined by the face CNN, enabling the segmented blocks without the human face to be excluded; meanwhile, the actual face blocks are preserved. Accordingly, the accuracy of the detected face blocks is much increased. Finally, the number of detected face blocks is summed up to obtain the number of humans in a photo. The experimental results reveal that the proposed method can effectively calculate the number of people in a photo with the average recall rate exceeding 100% in most conditions. It should be mentioned that this work aims to propose a practical app for quick calculating the number of people in a photo rather than develop an approach for face recognition. The proposed app is practical and can be applied in real environments. In response to the current expansion of the COVID-19 epidemic, if there are tour groups, out-of-school teaching activities, etc., the proposed app can help quickly calculate the number of people, avoid crowd gathering, and cause the risk of group infections. In particular, the help in calculating a large number of people is significant in real environments.

The rest of this paper is organized as follows: Section 2 describes the proposed app for automatically calculating people's numbers. Section 3 shows the experimental results. Finally, Section 4 concludes this study.

2. Proposed App for Automatic Calculating the Number of People

The proposed app firstly utilizes the VJ algorithm [5] to segment the face areas as blocks. Each segmented block is confirmed whether it belongs to the human face or not by a face CNN. Only the human face blocks are retained, while the segmented blocks without the human face are removed. The number of people can be obtained by summing up the retained human face blocks.

The graphic user interface (GUI) of the proposed app to calculate the number of people is shown in Figure 1. Firstly, a user must press the "Photo" button to take a snapshot from a webcam. If the user confirms this photo is available to include all members, the "Calculate" button should be pressed to perform face detection. Hence, detected rectangular blocks mark the face, and the number of people in the photo is displayed.

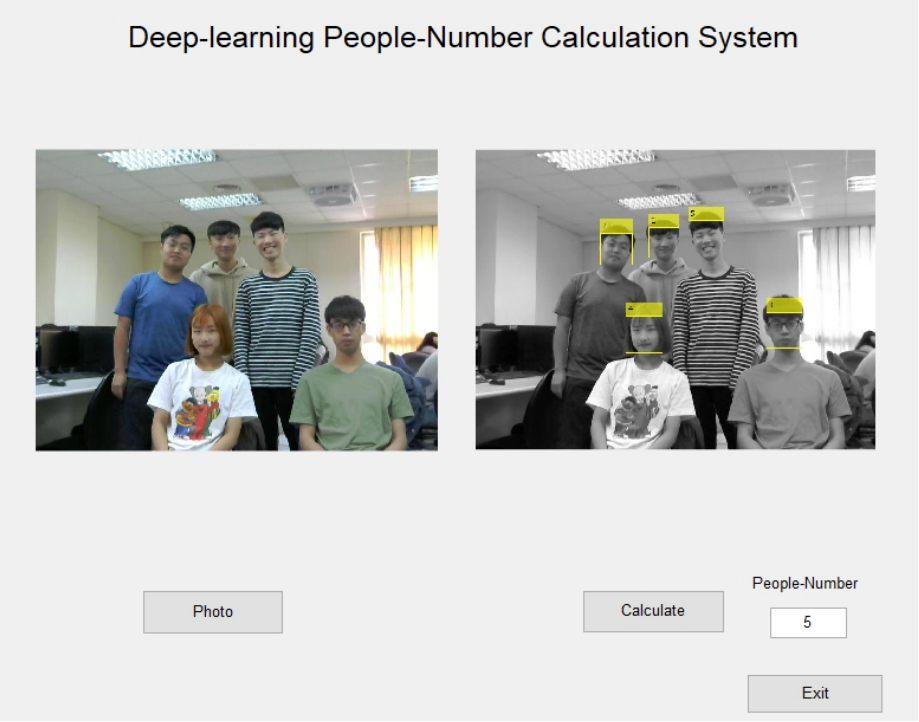

Figure 1. The GUI of the proposed calculator for calculating the number of people in a photo.

2.1. Human Face Detection

The block diagram of the proposed human face detection method is shown in Figure 2. Initially, we utilized the VJ algorithm [5] to segment the candidates of face blocks. The segmented blocks are classified into two categories, i.e., the face block and the segmented blocks without the human face. Each segmented block is manually labeled whether it contains the human face. Hence, the labeled segmented blocks are fed into a CNN to learn the face features, whereas the trained CNN is named the face CNN.

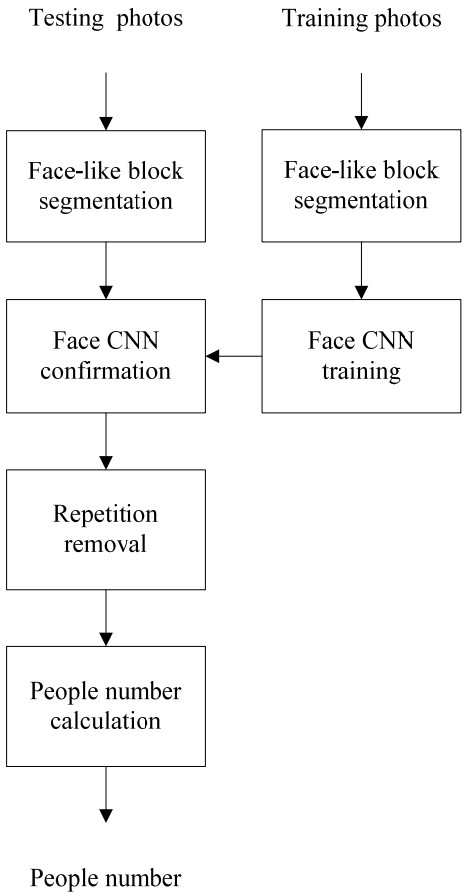

Figure 2. Block diagram of proposed calculator for calculating the number of people in a photo.

In the application phase, the face-like areas are also detected and marked as face-like blocks using the VJ algorithm. The face-like blocks are fed into the face CNN to determine whether they belong to the face category. The detected blocks without the human face are excluded; meanwhile, the blocks with the face inside are preserved.

Multiple detected results for the same face may exist in various sizes. These results are called repetition errors for face detection. Because the numerous detected results are all real face regions, they cannot be removed by the face CNN. A face at the same location with a different size should be refined to reduce the repetition error. Only the most significant extent at the exact center location is retained, while the others are removed. The repetition error is then reduced. Finally, the number of detected faces is summed up to obtain the number of humans in a photo.

The VJ algorithm [5] can effectively detect the face and provide flexible tuning parameters. This algorithm can also detect the upper body and contains some functions for detecting the eyes, eyebrows, nose, and mouth. The algorithm is robust and is used to perform initial face-range segmentation in this study. Figure 3 shows the detected face blocks with various parameters, while the minimum and maximum sizes of detectable objects are set to 20 × 20 and 85 × 85, respectively. By observing Figure 3a, many segmented blocks

without the human face are classified into face blocks when we use a loose condition in the VJ algorithm [5], i.e., the merging threshold is set to unity. The merging threshold plays a major role in face detection. If the VJ algorithm's segmentation condition is set to be strict, i.e., the merging threshold is set to three, all of the segmented blocks without the human face can be excluded, as shown in Figure 3c. Additionally, many face blocks cannot be successfully detected. After carefully tuning the value of the merging threshold, it is better to be set as two. The segmented result is shown in Figure 3b. Only two face regions cannot be correctly detected, and only one segmented block without the human face is falsely detected.

Figure 3 shows an example of face detection results using the VJ algorithm [5]. As shown in Figure 3a, all human faces can be thoroughly detected if the VJ algorithm's merging threshold is set to a low value (merging threshold = 1). However, many segmented blocks without the human face are also marked as face blocks, causing the false alarm to increase. The merging threshold value should be increased to reduce the false alarm rate, where the detected results are shown in Figure 3b,c. It can be found that most segmented blocks without the human face disappear, but some face areas cannot be successfully detected. Accordingly, the merging threshold should be carefully selected to obtain an acceptable result. Here, we employ a face CNN to make the segmented blocks without the human face being excluded, enabling the VJ algorithm's false alarm rate to be significantly reduced. Thanks to the face CNN's help, the merging threshold of the VJ algorithm only needs a loose setting, i.e., the merging threshold can be fixed to be unity, enabling all face areas to be successfully detected.

(**a**)

(**b**)

Figure 3. *Cont.*

(**c**)

Figure 3. An example of face detection results using the VJ algorithm with different merging thresholds: (**a**) merging threshold = 1; (**b**) merging threshold = 2; (**c**) merging threshold = 3.

2.2. Face CNN Training and Confirmation

As shown in Figure 3, most face blocks can be captured using a loose segmentation parameter in the VJ algorithm, i.e., the merging threshold is set to unity. Although many segmented blocks without the human face are classified as face blocks, they can be excluded by using the face CNN; meanwhile, the segmented blocks without the human face are effectively removed. Thus, the detection accuracy of the human faces is significantly improved.

The face-like blocks captured by the VJ algorithm are fed into a CNN to train a face CNN. The block size is 75 × 75. In turn, eight 2D convolutional filters were employed to capture facial features, where the size is 3 × 3. The batch normalization is also performed to ensure training accuracy. The maximum polling operation is then performed with the window size 2 × 2, where the stride is set to two, where the row and column resolutions are down-sampled by the factor two. The convolutional layer, batch normalization, and maximum polling layers are repeated five times to capture facial features effectively. Therefore, the accuracy rate for face detection is improved. Next, the 2D features are flattened and fed into a fully-connected layer with two outputs, which correspond to the human face and non-human, respectively.

A softmax function is used as the activation function at the output of the fully connected layer. Finally, one of the two outputs with a larger score corresponding to the recognized result of either representing a human face or a non-human one is selected as the face of CNN's output. The recognized result of the fully connected layer *i** can be expressed by

$$i^* = \underset{i}{\operatorname{argmax}}(p_i, i = 0,\ 1) \tag{1}$$

where p_1 and p_0 represent the probabilities of the human face and non-human one, respectively.

In (1), only the segmented blocks with larger value of p_1 will be preserved, i.e., $i^* = 1$. These blocks correspond to the existence of the face confirmed by the face CNN. The other segmented blocks, which are recognized as $i^* = 0$ by the face CNN, are removed, i.e., the VJ algorithm's detected results corresponding to the segmented blocks without the human face being removed. Therefore, the face's detection accuracy is significantly improved for the VJ algorithm using the face CNN. The detailed structure of the face CNN is shown in Table 1.

Table 1. Detailed layers of the face CNN.

Layer Number	Layer Name	Parameters
1	Image input layer	Image size = 75 × 75 × 1
2	Convolutional layer	Window size = 3 × 3, filter number = 8, zero padding = 1
3	Batch normalization layer	
4	Relu layer	
5	Max pooling layer	Window size = 2 × 2, stride = 2
6	Convolutional layer	Window size = 3 × 3, filter number = 16, zero padding = 1
7	Batch normalization layer	
8	Relu layer	
9	Max pooling layer	Window size = 2 × 2, stride = 2
10	Convolutional layer	Window size = 3 × 3, filter number = 32, zero padding = 1
11	Batch normalization layer	
12	Relu layer	
13	Max pooling layer	Window size = 2 × 2, stride = 2
14	Convolutional layer	Window size = 3 × 3, filter number = 64, zero padding = 1
15	Batch normalization layer	
16	Relu layer	
17	Max pooling layer	Window size = 2 × 2, stride = 2
18	Convolutional layer	Window size = 3 × 3, filter number = 128, zero padding = 1
19	Batch normalization layer	
20	Relu layer	
21	Fully connected layer	Class number = 2
22	Softmax layer	
23	Classification layer	

The face CNN can effectively extract face features and correctly classify the face and non-human face categories. Figure 4 shows the relationship between the accuracy rate and the number of various convolution layers of a face CNN. As the number of the convolutional layers increases, the accuracy rate increases. If the number of convolutional layers increases to five or more, the accuracy rate can reach 100%. Accordingly, the number of convolutional layers is selected to five in the experiments.

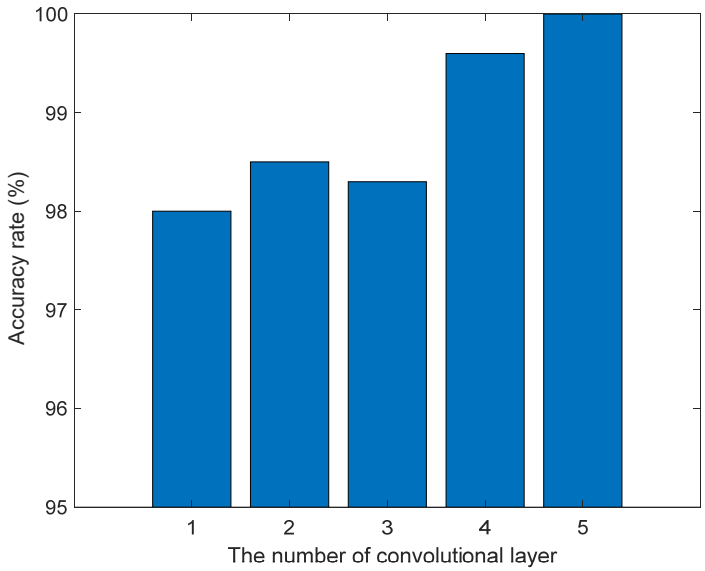

Figure 4. The relationship between the accuracy rate and the number of various convolution layers.

2.3. Repetition Removal

The face CNN can effectively exclude the segmented blocks without the human face. However, the face CNN cannot cope with repetition errors. These repeated detected face blocks are real human faces and correspond to identical faces; they cannot be removed by the face CNN. Two examples are shown in Figure 5a,c. It can be found that some human faces have the problem of repeated segmentation; that is, the same face is repeatedly marked twice at the neighbor location. To remove the repeated segmented blocks with various sizes for the same face, we only preserve the block with the smallest size, while the others are removed.

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

Figure 5. Examples of repetition removal for human face detection: (**a**) detection with repetition error for photo 1; (**b**) removal of repetition error for photo 1; (**c**) detection with repetition error for photo 2; (**d**) removal of repetition error for photo 2.

The coordinate position of each detected block's center point is calculated first. Hence, the distance between the center points of a pair of neighbor blocks is calculated. The distance between two detected blocks is calculated by

$$d_{i,j} = \sqrt{(x_{c_i} - x_{c_j})^2 + (y_{c_i} - y_{c_j})^2} \tag{2}$$

where (x_{c_i}, y_{c_i}) represents the center coordinate of the segmented blocks c_i and c_j. i and j denote the indices of detected blocks, respectively.

If there are many center points of the detected blocks too close, these blocks will correspond to the same face. Only the block with the largest size is preserved; the others are removed, given as

$$c_i = \begin{cases} \phi \text{ , if } d_{i,j} \le \delta \text{ and } A_i < A_j, j = 1 - N \text{ and } j \ne i \\ c_i \text{ , otherwise} \end{cases} \tag{3}$$

where δ (δ = 5) represents the distance threshold of repeated blocks for a face in an analysis photo. A_i denotes the area of the segmented block c_i. ϕ represents the removal for a segmented block, i.e., the segmented block is regarded as a repeated block and should be removed.

In (3), only one central point of the segmented blocks with the smallest area is retained, while the other segmented blocks in the neighborhood are removed. This enables the repeated segmented blocks for the same face to be removed. Two examples of repetition removal for face detection are shown in Figure 5. The detected results using the VJ algorithm and the face CNN suffered from repetition error for some faces are shown in Figure 5a,c. The refinement using (3) can effectively remove each detected face's repeated blocks, enabling the detection accuracy to be much improved.

3. Experimental Results

In the experiments, one hundred photos were taken by our mobile device to evaluate the proposed app's performance in calculating the number of people in each image. The state-of-the-art YOLOv4 (You Only Look Once version 4) network [24] and VJ algorithm are conducted for performance comparisons. The recall rate, precision rate, and *F*-measure are utilized to evaluate the calculated number of people's correctness.

3.1. Performance Comparisons

The precision rate of the recognized results can be computed by [25]

$$P\% = \frac{\text{The number of correctly detected people}}{\text{The number of detected people}} \cdot 100\% \tag{4}$$

In (4), the precision rate $P\%$ gets a high value when the number of correctly detected people increases. The higher the precision rate value indicates, the more accurate the detection number of people in a photo. The recall rate can reflect the successfully detected number of people. The recall rate can be computed by [25]

$$R\% = \frac{\text{The number of correctly detected people}}{\text{The number of people in a photo}} \cdot 100\% \tag{5}$$

From (5), the higher the number of correctly detected people in a photo, the higher the value of the recall rate $R\%$, i.e., it denotes better detection performance. Additionally, the *F*-measurement is employed to assess the overall performance of the proposed calculator, given as [25]

$$F\text{-measure} = 2 \cdot \frac{R \cdot P}{R + P} \cdot 100\% \tag{6}$$

In (6), the score of the *F*-measure is high when the scores of precision rate $P\%$ in (4) and recall rate $R\%$ in (5) are both high. These results indicate that the overall performance of calculating the number of people is satisfactory. The *F*-measure score decreases if the precision rate $P\%$ or recall rate $R\%$ score is reduced. Thus, the *F*-measure score can be utilized to evaluate the whole performance of calculating the number of people. The performance comparisons for the testing set are shown in Table 2, where the images in the testing set are not used in training the face CNN. The YOLOv4 network significantly outperforms the VJ algorithm and slightly beats the proposed app in the precision rate. The VJ algorithm's merging threshold is set to unity, enabling most face areas to be successfully detected. The recall rate is 99.53%. However, many segmented blocks without the human face are falsely detected, significantly reducing the precision rate. The face CNN can thoroughly remove the falsely detected face blocks, considerably improving the average precision rate from 36.34% to 98.61%. The recall rates obtained by the YOLOv4 network and the proposed approach are comparable. The overall performance in terms of the *F*-measure for the YOLOv4 network and proposed app can reach 98.61 and 98.27%, respectively. Accordingly, the detection accuracy of the YOLOv4 is the best, while that of the proposed app is also acceptable.

Table 2. Comparisons of detection results.

Measure	Score		
	VJ	**YOLOv4**	**Proposed**
Precision rate	36.34%	99.30%	98.61%
Recall rate	99.53%	97.93%	97.93%
F-measure	53.24%	98.61%	98.276%

The mobile device is an Intel Core i5 CPU and 8GB DDR4 memory without the help of a GPU. The performance comparison in terms of average time consumption is shown in Table 3. The YOLO v4 needs much more time (102.78 s) than the proposed app (0.09 s) in loading trained networks for face detection. It is attributed to the layer of the YOLOv4 network being very deep. On the contrary, the proposed app is relatively simple and effective. It takes only a short time to load the trained network. The YOLOv4 network needs more time (4.43 s/image) to calculate the people than the proposed app (2.55 s/image) by observing the recognition time. The YOLOv4 network obtains slightly higher recognition performance, as shown in Table 2. However, the time consumption is higher than the proposed method, as shown in Table 3, particularly the network loading time. Although the architecture of the proposed method is simple, the time consumption time for loading the network is short, while the recognition accuracy is acceptable. It is beneficial in implementing a mobile device for quickly calculating people in an image. Accordingly, the proposed approach is practical in realistic environments.

Table 3. Comparisons of time consumption.

Task	Time Consumption (s)	
	YOLOv4	Proposed
Network loading	102.78	0.09
Recognition	4.43	2.55

Figure 6 shows another example of detected face areas with various merging thresholds, while the minimum and maximum sizes of the detectable object are 20 × 20 and 85 × 85, respectively. By observing Figure 6a, all face areas can be successfully detected when the merging threshold is set to unity. However, many segmented blocks without the human face have also been classified into face blocks in the VJ algorithm [5]. If the merging threshold is selected to two, many non-human blocks can be excluded, as shown in Figure 6b. Only one face cannot be successfully detected. However, a few segmented blocks without the human face are still identified as face blocks. If the merging threshold is set to five, the detected results are shown in Figure 6c. The red bounding box highlights the differences among Figure 3a–c. One can find that all of the segmented blocks without the human face are excluded. However, the VJ algorithm cannot successfully detect one face block on the left-hand side of the photo cannot be successfully detected. This result is due to the value of the merging threshold being set too high.

3.2. Discussions

By observing Figures 3 and 6, one can find that the merging threshold should be varied with various photos to obtain acceptable performance. The definition of a robust merging threshold for the VJ algorithm is complicated. Here, we propose using a small merging threshold, set to be unity, in the VJ algorithm as the first stage, so all face blocks can be thoroughly detected, despite the high false alarm rate. Hence, the face CNN is performed to exclude the segmented blocks without the human face, reducing the false alarm rate significantly.

Compared with the state-of-the-art YOLOv4 network, the performances are comparable in accuracy rate, as shown in Table 2 and Figures 7 and 8. The red bounding box highlights the differences in Figures 7 and 8. However, our method is easy to train and implement. In addition, training the YOLOv4 network is time-consuming. On the contrary, our approach is easy to train the model. The differences between the VJ algorithm, YOLOv4 network, and the proposed app are summarized and shown in Table 4. Because of the short loading time and high accuracy, as shown in Tables 2–4, users will intuitively enjoy using the proposed app.

(**a**)

(**b**)

(**c**)

Figure 6. An example of human face detection results using the VJ algorithm with various merging thresholds: (**a**) merging threshold = 1; (**b**) merging threshold = 2; (**c**) merging threshold = 5.

(**a**)

(**b**)

(**c**)

Figure 7. Comparisons of people calculating results: (**a**) using the VJ algorithm with merging threshold = 2; (**b**) using the YOLOv4 network; (**c**) using the proposed method.

(**a**)

(**b**)

(**c**)

Figure 8. Comparisons of people calculating results: (**a**) using the VJ algorithm with merging threshold = 5; (**b**) using the YOLOv4 network; (**c**) using the proposed method.

To demonstrate the application for this study, we made a video as an example, and the hyperlink is given as https://youtu.be/xUoRrcGgm_w (accessed on 20 May 2022). A snapshot of the proposed app for calculating the number of people is shown in Figure 9. Some students will enter the museum. The staff feels sad that students cannot buy tickets to enter the museum quickly. Utilizing the proposed app can help her to calculate the number of people rapidly.

Table 4. System comparisons among the VJ algorithm, YOLOv4 network, and the proposed app.

Measure	System Requirement and Performance		
	VJ	**YOLOv4**	**Proposed**
Parameter setting complexity	High	Not needed	Not needed
Precision rate	Low	High	High
Recall rate	High	High	High
F-measure	Middle	High	High
Model loading time	Short	Long	Short
Model training time	Middle	Long	Short

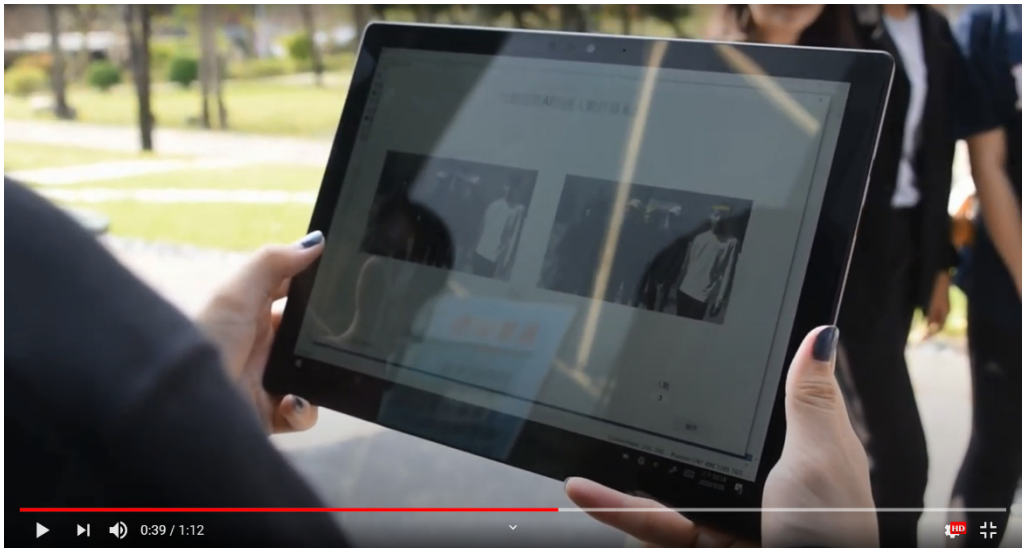

Figure 9. An example of using the proposed app for calculating the number of people.

4. Conclusions

This paper proposes a valuable app for calculating the number of people in a photo. This app applies a face CNN with a VJ algorithm to segment face regions in an image. Firstly, the VJ algorithm's loose condition is utilized to detect face-like blocks as much as possible, even though the false detection rate is high. In turn, the face blocks and non-human ones are utilized for training a face CNN, which is used to determine whether a detected block is an actual face. The trained face CNN removes the segmented blocks without the human face. The number of face blocks that remained is the number of people in the photo. Experimental results show that the proposed app can obtain the average accuracy rate, recall rate, and *F*-measure reaching 95%, 97%, and 96%, respectively. The proposed app is efficient for calculating the number of humans in a photo and can thus be used in practical work; for example, a tour guide can calculate how many tour members gather at a viewpoint. The limitation of this app is that the human face should face the camera, where slight angle movement is also available. In the future, we will collect the photos so that the human faces do not meet the camera as training data, enabling the proposed app to calculate the human faces not facing the camera accurately.

Author Contributions: Conceptualization, C.-T.L.; methodology, C.-T.L. and C.-J.O.; software, C.-J.O. and Y.-Y.L.; validation, C.-J.O. and Y.-Y.L.; formal analysis, C.-T.L., C.-J.O. and Y.-Y.L.; investigation, C.-T.L., C.-J.O. and Y.-Y.L.; resources, C.-J.O.; data curation, C.-J.O.; writing—original draft preparation, C.-T.L., C.-J.O. and Y.-Y.L.; writing—review and editing, C.-T.L.; visualization, C.-T.L., C.-J.O. and Y.-Y.L.; supervision, C.-T.L.; project administration, C.-T.L.; funding acquisition, C.-T.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Ministry of Science and Technology, Taiwan, grant number MOST 104-2221-E-468-007.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Acknowledgments: Our gratitude goes to the reviewers for their valuable comments, which have improved the quality of this paper.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Tuncer, T.; Dogan, S.; Abdar, M.; Pławiak, P. A novel facial image recognition method based on perceptual hash using quintet triple binary pattern. *Multimed. Tools Appl.* **2020**, *79*, 29573–29593. [CrossRef]
2. Lin, J.; Xiao, L.; Wu, T.; Bian, W. Image set-based face recognition using pose estimation with facial landmarks. *Multimed. Tools Appl.* **2020**, *79*, 19493–19507. [CrossRef]
3. Chakraborty, S.; Singh, S.K.; Chakraborty, P. Cascaded asymmetric local pattern: A novel descriptor for unconstrained facial image recognition and retrieval. *Multimed. Tools Appl.* **2019**, *78*, 25143–25162. [CrossRef]
4. Hssayni, E.H.; Ettaouil, M. New approach to face recognition using co-occurrence matrix and Bayesian neural networks. In Proceedings of the IEEE International Conference on Optimization and Applications 2020, Beni Mellal, Morocco, 20–21 April 2020; pp. 1–5.
5. Viola, P.; Jones, M. Rapid object detection using a boosted cascade of simple features. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2001, Kauai, HI, USA, 8–14 December 2001; pp. 511–518.
6. Wang, D.; Yu, H.; Wang, D.; Li, G. Face recognition system based on CNN. In Proceedings of the International Conference on Computer Information and Big Data Applications 2020, Guiyang, China, 17–19 April 2020; pp. 470–473.
7. Wu, H.; Zhang, K.; Tian, G. Simultaneous face detection and pose estimation using convolutional neural network cascade. *IEEE Access* **2018**, *6*, 49563–49575. [CrossRef]
8. Zhang, K.; Zhang, Z.; Li, Z.; Qiao, Y. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Process. Lett.* **2016**, *23*, 1499–1503. [CrossRef]
9. Li, J.; Liu, L.; Li, J.; Feng, J.; Yan, S.; Sim, T. Toward a comprehensive face detector in the wild. *IEEE Trans. Circuits Syst. Video Technol.* **2017**, *29*, 104–114. [CrossRef]
10. Yu, B.; Tao, D. Anchor cascade for efficient face detection. *IEEE Trans. Image Process.* **2019**, *28*, 2490–2501. [CrossRef]
11. Yang, S.; Luo, P.; Loy, C.C.; Tang, X. Faceness-net: Face detection through deep facial part responses. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 1845–1859. [CrossRef]
12. Bong, K.; Choi, S.; Kim, C.; Yoo, H.J. Low-power convolutional neural network processor for a face-recognition system. *IEEE Micro* **2017**, *37*, 30–38. [CrossRef]
13. Ranjan, R.; Patel, V.M.; Chellappa, R. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *41*, 121–135. [CrossRef]
14. Guo, C.; Liu, Y.; Jiao, X. Study on the influence of variable stride scale change on image recognition in CNN. *Multimed. Tools Appl.* **2019**, *78*, 30027–30037. [CrossRef]
15. Lu, C.T.; Wang, L.L.; Shen, J.H.; Lin, J.A. Image enhancement using deep-learning fully-connected-neural-network mean filter. *J. Supercomput.* **2021**, *77*, 3144–3164. [CrossRef]
16. Chernyshova, Y.S.; Sheshkus, A.V.; Arlazarov, V.V. Two-Step CNN Framework for text line recognition in camera-captured images. *IEEE Access* **2020**, *8*, 32587–32600. [CrossRef]
17. Sapijaszko, G.M.; Mikhael, W.B. Facial recognition system using mixed transform and multilayer sigmoid neural network classifier. *Circuits Syst. Signal Process.* **2020**, *39*, 6142–6161. [CrossRef]
18. Ramya, R.; Mala, K.; Nidhyananthan, S.S. 3D facial expression recognition using multi-channel deep learning framework. *Circuits Syst. Signal Process.* **2020**, *39*, 789–804. [CrossRef]
19. Lou, G.; Shi, H. Face image recognition based on convolutional neural network. *China Commun.* **2020**, *17*, 117–124. [CrossRef]
20. Masi, I.; Chang, F.; Choi, J.; Harel, S.; Kim, J.; Kim, K.; Leksut, J.; Rawls, S.; Wu, Y.; Hassner, T.; et al. Learning pose-aware models for pose-invariant face recognition in the wild. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *41*, 379–393. [CrossRef]
21. Low, C.Y.; Teoh, A.B.J.; Toh, K.A. Stacking PCANet +: An overly simplified ConvNets baseline for face recognition. *IEEE Signal Process. Lett.* **2017**, *24*, 1581–1585. [CrossRef]
22. Lu, C.-T.; Su, C.-W.; Jiang, H.-L.; Lu, Y.-Y. An interactive greeting system using convolutional neural networks for emotion recognition. *Entertain. Comput.* **2022**, *40*, 100452. [CrossRef]
23. Dumitrescu, F.; Boiangiu, C.-A.; Voncilă, M.-L. Fast and Robust People Detection in RGB Images. *Appl. Sci.* **2022**, *12*, 1225. [CrossRef]
24. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
25. Van Rijsbergen, C.J. *Information Retrieval*, 2nd ed.; Butterworths: London, UK, 1979.