Development and Optimization of Check-in System Based on Face Recognition Technology

Pingping Chen\textsuperscript{a}, Xiaoran Geng\textsuperscript{b}, Min Zou\textsuperscript{c}, Qingming Xu\textsuperscript{d} and Dingying Tan\textsuperscript{\ast}

School of Medical Information Engineering, Guangzhou University of Chinese Medicine, GuangZhou, China

\textsuperscript{\ast}Corresponding author e-mail: tandy@gzucm.edu.cn, \textsuperscript{\ast}chenpingping@gzucm.edu.cn, \textsuperscript{\ast}417195779@qq.com, \textsuperscript{\ast}562139648@qq.com, \textsuperscript{\ast}xu_qm@qq.com

Abstract. Face recognition technology is widely used. By comparing the speed and accuracy of Face++ and Baidu AI face recognition, the face sign-in service was collected in the mobile phone or pad APP, which realized the mobile check-in attendance that can be used for college students or training institutions. The system was characterized by accuracy, fast and anti-cheating ability. The experimental results show that Baidu AI's face detection and recognition performance is better, its speed is more than twice faster than Face++, the recognition score is higher than Face++, and includes live detection, which can effectively prevent static picture cheats. However, Baidu AI's face detection time (about 271.2ms) was much higher than the commercial face detection algorithm (about 25ms), so the Adaboost face detection algorithm was studied, and the Adaboost algorithm based on OpenCV was transplanted to Android. The platform enables real-time detection of faces. After the system optimization, the face detection time is about 10ms, which effectively improves the face detection speed.

1. Introduction

Face recognition technology has broad prospects, and its non-mandatory and non-contact characteristics have attracted the attention of domestic researchers. Today, face recognition technology is at a mature stage. More applications in major application markets are camera photography combined with face recognition technology, mobile payment and real-name authentication, but in education, face technology has not been widely used, especially face registration is not much, Baidu time and attendance punch card application is in the research and experimental stage. At present, there are a small number of documents mentioning face sign-in, such as the study on the automatic sign-in mode of training institutions based on face recognition \cite{2} mentions brushing attendance, but there is no detailed related implementation content.

This system explores the traditional handwriting, fingerprint or punch-in sign mode transfer face recognition sign-in mode, which is suitable for the mobile attendance and attendance system of students’ classrooms and conferences. It not only completes the sign-in process accurately and quickly, but also has strong anti-cheating ability and is convenient for teachers and students. The conference sign-in mode supports multiple attendance at the same time. The classroom mode will send the successful sign-on information to the parents’ mobile phone in real time through SMS, and push the
check-in report to the parents to solve the security problems that may be caused by the students not attending classes and skipping classes. Higher use value in training institutions or college students' classrooms.

2. Related work
Face recognition technology has been developed so far, and its algorithm research has many achievements, such as the traditional algorithm based on Adaboost face detection algorithm [3], Face recognition algorithm based on Gabor filter [4] and improved LBP face recognition algorithm [5]. In recent years, face recognition is based on the rapid development of neural networks and deep learning, such as GRNN neural network based on genetic optimization [6] and other algorithms to train a face recognition model with extremely high accuracy. The research on face recognition has been pushed to a new height and is widely used in finance, banking systems, public security, and access control.

Our life is inseparable from face recognition technology. Such as high-speed rail face brake, Alipay's face payment, today's headline real name authentication, Android phone face unlocking and Facebook began to launch face recognition for global users. However, there is very little education involved, and the related research literature is less and the content is vague. Therefore, the research proposes that the face-to-face attendance service built-in mobile phone or pad APP is applied to college students' classrooms or educational training institutions, and the system is optimized.

3. Sign-in system based on face recognition technology

3.1. Basic framework
The system is mainly divided into three major modules. As shown in Figure 1, collecting face pictures to be checked in to create a face library to be recognized, the face data is acquired on the spot, the detection and recognition are performed, and the face detection speed is optimized. The application provides two modes, conference mode and classroom mode, which are suitable for work and life scenarios. The conference mode is characterized by multi-person detection and recognition at the same time, which is characterized by high efficiency and speed. The classroom mode can only be checked by a single person. When the student completes the check-in, the successful sign-in information will be sent to the parent's mobile phone in real time through the mobile phone text message. At the same time, the check-in report will be generated, which can be shared with the parents to understand the student's class situation and solve possible safety issues caused by students not attending classes on time, skipping classes, etc.
3.2. Functional module design

3.2.1. Collecting face modules through Android device camera. The basic process of shooting is shown in Figure 2. First, check whether the current Android device has camera resources. If it exists, request access to the camera, otherwise it ends; check camera permissions. If the user does not have permission, the pop-up window prompts the user for authorization. If the authorization is successful, Open the camera and end otherwise; create a camera preview class, load the layout, and blend the preview image with the preview interface of the design. The camera callback listener realizes converting the photographed image into a bitmap, and then compressing and mirroring the bitmap, finally outputting it into a common image format file by streaming output. The camera needs to solve the following problems during image acquisition:

1. Camera preview
   The Android platform not only has serious system fragmentation, but also the hardware configuration provided by various brand mobile phone manufacturers is different, which leads to the endless emergence of Android adaptation problems. The camera is one of them. Google's official Android development documentation currently provides two camera APIs, the Camera API and the Camera 2 API, the latter only supporting Android API Level 21 or above, but individual phones still use the Camera API, based on system compatibility considerations, need to switch to call different versions of the API and SurfaceView to customize the camera preview interface according to the actual situation of the phone.

2. Focusing scheme
   Achieve auto focus, touch focus, and mobile focus. Autofocus is obtained by looking at the source code of zxing, and by continuously calling the Camera autofocus API with AsyncTask. Currently, the system uses a third autofocus scheme that can be adapted to the autofocus of most mobile phones. Touch focus is achieved by monitoring the gesture event callback auto focus. Mobile focus is based on the Android system itself with a lot of mobile sensors, most of the cameras now have auto-focus when taking pictures, so it is achieved by monitoring acceleration management based on motion sensors.

3. Camera adjustment
   The system has the front camera turned on, and the preview direction is set to be vertical through the Carema API, which causes the screen orientation does not match sensor direction. Therefore, the preview image seen is upside down and needs to be rotated 270 degrees to get a normal picture. It was
concluded during the testing process that if the camera's preview direction is changed, the rear camera needs to rotate the image 90 degrees, and the front camera needs to rotate the image 270 degrees to ensure that the image is normal.

(4) Preview image and take picture stretch deformation

By obtaining all the preview and shooting sizes supported by the camera hardware, and then filtering from all the previews and shooting sizes obtained, meet the condition is the same size of the SurfaceView aspect ratio. The current main size ratio is 4:3 or 16:9, update the settings via camera.setPrameters.

(5) Camera magnification adjustment

If the camera is closer to the face, the preview face will be larger, otherwise it will be smaller. If the picture is too large or small, it may cause face recognition errors. Although a fixed magnification can be set to control the face size, the magnification of most mobile phones is inconsistent, so it cannot be adapted to most mobile phones. The solution adopted by the system is to obtain the maximum magnification provided by the camera and select one Nth of the magnification value as the current magnification, which perfectly solves the adaptation problem of the mobile phone.

(6) Image compression

The size of the captured image is relatively large, and its loading will occupy many memory of the mobile phone, in addition, the volume of the network transmission will affect the face recognition rate and accuracy, the sampling rate compression and quality compression are adopted.

3.2.2. Collecting Face Modules Through Android Device Albums. The system supports multiple face registrations through multiple photos in the album. The Android system does not provide a UI for selecting multiple images, so need to customize the image selector ImagePickerLoader. The ImagePickerLoader is designed to load ContentProvider resources, including image and video media files, using the Android Loader mechanism to adapt to the Android Q system. The y-axis scrolling...
acceleration of the listen list. If the album list page is in the fast scrolling state and the image is not loaded, the list scrolling slowly and reloading will not cause memory shake, but the user's visual experience is slightly inferior. Matisse supports Glide and Picasso. The ImagePickerLoader extension interface for loading thumbnails, loading large images, and loading video playback. It is more scalable and has no restrictions on loading frames.

3.2.3. Face registration module. Face Detection: The face detection algorithm based on Face++ and Baidu AI realizes real-time detection. The system detects the user's face at the check-in site, calls the camera to take pictures and compress the static picture, then transmits the network to the server. The system processes the detection result returned by the server to complete the face detection process. The face registration is based on the face detection. If the face database is not built, the face database is created, and then the face is added, otherwise the face is directly added to the library.

3.2.4. Face Recognition Module. The core logic of face recognition is to obtain the facial feature values of the scene, and compare the feature values of each face one by one in the applied face database, and then return a face with the highest similarity. When the administrator opens the check-in mode, the face image of the scene is collected through the front camera of the device, and the face signing is completed in the system 1s. The specific process mainly includes two steps. The first is to take a bitmap from the camera and after high-quality compression, and then call the face recognition interface to obtain the basic facial features and logos returned by the third-party platform server, and finally match the face features to the face database, returns a face with the highest similarity in the face database. If the face matching threshold is greater than the set face, the face is registered, the check-in is successful, otherwise the check-in fails. The test equipment uses Xiaomi 2 Android5.0 system, vivoX21 Android8.0 system and Glory 10 Android9.0 system. According to the data comparison of Tables 1, 2 and 3, Baidu AI has better face detection and recognition performance, face detection and recognition rate is more than twice faster than Face++, and includes live detection to effectively prevent static picture cheats. The system completes detection and recognition at approximately 605.8ms, and the face recognition score is relatively high. However, Baidu AI's face detection takes about 271.2ms, which is much higher than the commercial face detection time (about 25ms). So we study OpenCV implements face detection based on Adaboost algorithm, and then this algorithm is implemented on the Android platform. The goal is to improve the speed of face detection.

| Table 1. Face++ and Baidu AI compare time detection of single face |
|--------------------------------|----------------|----------------|
| Detection time (ms)          | Face++  | Baidu AI |
|-------------------------------|---------|----------|
| Group one                     | 851     | 246      |
| Group two                     | 695     | 339      |
| Group three                   | 672     | 256      |
| Group four                    | 905     | 275      |
| Group five                    | 675     | 240      |
| Average value                 | 759.6   | 271.2    |

| Table 2. Face++ and Baidu AI compare time detection of three faces |
|--------------------------------|----------------|----------------|
| Detection time (ms)          | Face++  | Baidu AI |
|-------------------------------|---------|----------|
| Group one                     | 1531    | 475      |
| Group two                     | 1248    | 386      |
| Group three                   | 1345    | 467      |
| Group four                    | 1611    | 412      |
| Group five                    | 1123    | 378      |
| Average value                 | 1371.6  | 423.6    |
Table 3. Baidu AI and Face++ compare the recognition time of a single face

| Type         | Baidu AI                | Face++                  |
|--------------|-------------------------|-------------------------|
|              | Detection and identification time (ms) | Detection and identification time (ms) |
|              | Face score confidence   | Face score confidence   |
| Group one    | 1203                    | 93.671                  | 589                     | 98.623                  |
|              | Wearing glasses on the face | 91.671                  | Wearing glasses on the face |
| Group two    | 1125                    | No face glasses         | 647                     | No face glasses         | 94.586                  |
| Group three  | 1247                    | About 30 degrees on the right face | 82.503                   | About 30 degrees on the left face | 86.243                  |
| Group four   | 1004                    | About 30 degrees on the right face | 81.572                   | Looking up about 30 degrees | 85.748                  |
| Group five   | 1009                    | Looking up about 30 degrees | 84.262                   | Looking up about 30 degrees | 83.154                  |
| Average value| 1117.6                  | 86.7358                 | 605.8                   | 87.6708                 |

4. Face detection speed optimization

4.1. Adaboost face detection algorithm

Adaboost [7] (Adaptive Boost, Adaptive Boost) is a supervised machine learning algorithm based on the PAC (Probably Approximately Correct) model of Yoav Freund and Robert Schapire. Many face detection algorithms are based on the implementation of Adaboost [2]. The process of implementing face detection is shown in Figure 3. The feature selection and calculation are very important, which will directly affect the training speed of face detection. In addition, the classifier is the core part of the detection process and consists of a weak classifier and a strong classifier. Adaboost mainly focuses on the samples that are misclassified in the weak classifier training process, and then increases their weights for iterative training of the next weak classifier. These weak classifiers are combined into a strong classifier, and several strong classifiers form a cascade structure classifier. The strong classifier mainly focuses on the error sample. If the wrong sample is detected, the next strong classifier will not be called, and so on to increase the strict filtering conditions, the face detection speed is effectively improved.

Figure 3. Flow chart of Adaboost face detection algorithm
4.2. Haar-like features
Haar features (rectangular features) are based on grayscale images. As a very classic feature extraction algorithm, compared with the early use of RGB values of image pixels to calculate feature values, the feature extraction speed of the face is greatly accelerated.

A window is composed of two black and white rectangles. Its feature value is all pixels of the black rectangle and all pixels of the white rectangle are subtracted. The Haar feature can describe facial features, such as eyes, the white part of the feature template represents the eye area, the black is the non-eye area, and an image has many of these feature templates.

If it is a small number of feature templates, the amount of calculation for calculating rectangular features is very small, but in reality, a face image needs to be calculated at different positions in the image using different types of features and forms, resulting in a huge amount of calculation. For a 24 * 24 image, there are 115984 features in total, which requires a very large amount of calculation to calculate the pixel sum of each feature, and there are many repeated calculations. In order to reduce the number of calculations and speed up the calculation of the feature pixel sum, Paul Viola [8] proposed the use of integral image method.

4.3. Integral map
The integral map [9] is a matrix area pixel formed by defining an array to store the starting point of the image to each point. The area pixels and the array index can be used to avoid repeated calculations, thereby improving the calculation speed. The disadvantage is that the data stored in the array will be relatively large and occupy more memory space. The calculation result is only related to the integral figure of the end points of the rectangle, and has nothing to do with the coordinates of the image. The time complexity is reduced from O (mn) to O (1), where m and n respectively represent the number of rows and columns. Space is exchanged for time, thus speeding up the calculation of Haar features.

4.4. Implementation process
If the Android platform implements the Adaboost algorithm [10], it will obviously improve the face detection speed. The system is implemented based on the Android platform. Since OpenCV [11] implements the Adaboost face detection algorithm, the algorithm and face model are transplanted into the system. This algorithm is implemented by C/C++ language and can be run across platforms. Android is embedded needs cross compilation through NDK, compiles so library using cmake, and Java layer calls JNI and C/C++ layer communication to improve image processing speed. The algorithm migration implementation is mainly divided into the following steps (code is shown in Table 4):

1. Android Studio configures NDK and cmake environments to compile OpenCV libraries.
2. The Java layer defines the native methods related to face detection, such as setting the classifier, loading the face model, and compiling the generated header file.
3. Create a native-lib.cpp file and implement the native method in step 2.
4. Compile the native-lib so library by cmake.
5. The Java layer loads the native library of native-lib to implement static image detection and video detection.

| Table 4. Code to implement face detection |
|------------------------------------------|
| 1. private static native long
  nativeCreateCascadeClassifier(String cascadeName, int minFaceSize); |
| 2. private static native void
  nativeDetectFace(long arg, long inputImage, long faces); |
| 3. vector<Rect> rectFaces; |
| 4. ((DetectorAgregator *arg)->tracker->process(*((Mat *) imageGray)); |

   * analysis: Set up the classifier and initialization, face detection through this part of the code definition and implementation.*
4.5. Experimental results and analysis
This experiment divides the experiment into two groups to compare the face detection time before the system is not optimized. The single face detection and multiple face detection were used to observe the face detection time.

Experiment with 50 pictures. Before the system was optimized, the face detection test provided by Baidu AI took about 271.2ms. After the system was optimized, the face detection time was only about 11ms, and the detection speed was increased by 17 times. The system is slightly less robust to the face angle. If the side face angle is greater than about 15 degrees, the detection time is still very fast, and the corresponding detection time before optimization is increased to 889ms.

For multiple face detection, this experiment uses two-face images, a total of 50 tests. Before the system optimization, it takes about 423.6ms to detect multiple faces, and the number of faces increases with the increase of the number of faces. The face is tilted within 90 degrees, and the detection accuracy is 96%. After the system is optimized by Adaboost face detection algorithm, the accuracy is 94%, the detection time is about 10ms, and the face detection is completed quickly.

According to the results of Table 5, the system adopts the Adaboost algorithm to achieve the face detection speed of about 10ms, which can provide the face detection speed quickly. However, for the robustness of the face tilt angle, further research and improvement of the threshold of the classifier are needed to train a more comprehensive face database, thereby enhancing the robustness of the system's face detection.

| Number of faces | Optimized grayscale time (ms) | The number of faces detected | Before optimization face detection time (ms) | Optimized face detection time (ms) |
|-----------------|------------------------------|-------------------------------|---------------------------------------------|-----------------------------------|
| Single          | 1                            | 1                             | 271.2                                       | 11                                |
| Multiple        | 1                            | 2                             | 423.6                                       | 12                                |

5. Conclusion
In this study, by comparing the rate and accuracy of face detection, face recognition and face search services of Face++ and Baidu AI, the face sign-in service is collected in the APP of the mobile phone or pad. The experimental results show that Baidu AI has better face detection and recognition performance, and its face detection and recognition speed is more than twice faster than Face++. The recognition score is higher than Face++, and includes live detection, which can effectively prevent static pictures from cheating. However, Baidu AI's face detection time (about 271.2ms) is very slow compared to the commercial face detection algorithm (about 25ms). Therefore, the Adaboost face detection algorithm is studied, and the Adaboost algorithm based on OpenCV is transplanted to the system. The face detection time of the system optimization is about 10ms, which effectively improves the face detection speed.

Acknowledgments
This work was financially supported by Guangzhou University of Chinese Medicine, grant number: A3-0433-181-427-037, Innovation and entrepreneurship education project of Education Bureau of Guangzhou Municipal, grant number:2019KC103 and Research project of innovation and entrepreneurship education reform of undergraduate colleges and universities in Guangdong Province, grant number:2018A010920.

References
[1] Dave R, Vyas A, Desai N P. Face Recognition Techniques: A Survey [J]. 2018.
[2] Li Xinzhan. Face detection method based on improved Adaboost algorithm [J]. Technology and Economic Guide,2018 (18):26-26.
[3] Wang M , Deng W . Deep Face Recognition: A Survey [J]. 2018.
[4] Russell M, Fischaber S. OpenCV based road sign recognition on Zynq [C]// IEEE International Conference on Industrial Informatics. 2013.
[5] Jiang W, Guo G, Lai Z. An improved adaboost algorithm based on new Haar-like feature for face detection [J]. Journal of Shandong University, 2014, 44 (2):43-48.
[6] Russell M, Fischaber S. OpenCV based road sign recognition on Zynq [C]// IEEE International Conference on Industrial Informatics. 2013.
[7] Mliki H, Fendri E, Hammami M. Face Recognition Through Different Facial Expressions[J]. Journal of Signal Processing Systems, 2015, 81 (3):433-446.
[8] Xin X U , Xiao-Li H , Fang W . Long distance face detection based on Adaboost algorithm[J]. Computer Engineering & Design, 2015.
[9] A Survey of Face Recognition approach. Pandya J M, Rathod D, Jadav J J. International Journal of Engineering Research and Applications. 2013.
[10] WEI Yue-na, WU Xi-sheng. Face recognition algorithm based on Gabor feature fusion and block processing [J]. Computer Engineering & Design, 2017(3).
[11] LIU Bin, MI Qiang, XU Yan. Face recognition based on weighted fusion of LBP and MB-LBP features [J]. Computer Engineering & Design, 2018.