Proceedings of the 3rd International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/6/20230846

Masked face reconstruction and recognition through principle components analysis

Chenxin Cui

Department of Communication Engineering, Jilin University, Changchun, Jilin, China

cuicx2019@mails.jlu.edu.cn

Abstract. Due to the Covid-19 virus infecting humans since 2019, the traditional methods of face recognition are not suitable anymore, for the reason that lots of countries and regions request people to wear masks in public places, whereas the masks will add additional noise in the recognition process. To solve this problem, we suggest a new method named "masked face recognition", whose principle is first converting the masked face image to an unmasked reconstruction image by applying Principle Components Analysis. We call it the Reconstruction Process; then using Convolutional Neural Network to exert face recognition on unmasked reconstruction images, we call it the Recognition Process. During the Reconstruction Process, we first detect the unmasked area, then linearly fit the unmasked area with Eigenfaces generated from the PCA process and a set of coefficients, which are the objectives we optimize. After the optimization, we are able to obtain the reconstruction image by the linear combination of the Eigenface with its corresponding coefficient. Experimental results clearly show the high effectiveness of our method, and the accuracy of being recognized correctly is increased from about 80% if we use the original masked images to 96.36% if we use the unmasked reconstruction images.

Keywords: face recognition, face reconstruction, PCA.

1. Introduction

Face recognition is a biotechnology that aims to identify a person based on their facial features, which has already attracted a lot of attention and interest among researchers in recent years and received many significant achievements like Face-Net, etc. Face recognition is superior to other widely used biometric recognition methods like fingerprints in terms of hygiene, i.e., to avoid touching the scan table. However, with the Covid-19 virus infecting humans since 2019, people are required to wear masks nowadays in many countries. Therefore, lots of widely used face recognition methods are not suitable anymore since the prerequisite of using the whole face to extract the features is not feasible nowadays. To solve those problems, more and more research has been focused on using merely the upper part of the face (i.e., the forehead, eyes, eyebrows, and sometimes a part of the nose, etc.) to extract the features. We can name that research "masked face recognition," which is also the topic of this paper. However, it is still of great importance to clarify the difference between "masked face recognition" and "occluded face recognition", where the latter one, in fact, has already been researched for a period of time. The difference is subtle but easy to understand: "masked face recognition" means the occluded part of the face only occurs upon the mouth, whereas in "occluded

DOI: 10.54254/2755-2721/6/20230846

face recognition", the occluded part can occur anywhere on the face (i.e. not only masks upon the mouth, but also sunglasses on the eyes, etc.). Anyway, the fewer regions, the more difficult. Masked face recognition is a prominent topic, much more sophisticated than traditional face recognition.

To solve this tough task, we suggest the reconstruction method, which means using the unmasked area to reconstruct the masked area. The principle of the reconstruction method is first detecting the unmasked area, then utilizing some algorithms (like in this paper, we use Principle Components Analysis, or PCA) on the unmasked area to generate a series of parameters which are later used to generate a new image without a mask. This specific new image is a kind of semi-manufacture. After combining the original masked image with this new image, we can receive the final edition of reconstruction images, which are then used to classify who they represent through the Convolutional Neural Network, or CNN, for short. In summary, the reconstruction process can be viewed as removing the mask from the masked image and adding the simulated truly face. Thus far, we have succeeded in converting the masked image into an unmasked image, although the effect may not be satisfactory. Therefore, with the unmasked image, or the normal face image, we are able to use some top-of-the-line models like Face-Net to exert traditional face recognition, although it is not our primary achievement. Our study mainly focuses on the reconstruction process.

With the images produced in the reconstruction process, we are able to devise some experiments to prove this idea's effectiveness. In particular, we have to prove that the usage of reconstruction images as the objective to be classified via CNN is more accurate than using masked images themselves directly. Otherwise, this task is meaningless. It seems incongruous to waste time generating reconstruction images if the accuracy of classification on masked images is greater than that of reconstruction images. Fortunately, this bad ending never occurs in our experiments. We can receive the result with an accuracy of up to 96.36% if we use reconstruction images, whereas about 80% in the original masked images. Furthermore, we introduce in the Experiment section some additional metrics to evaluate our performance, such as the runtime of the reconstruction process.

The following paper is organized as follows: Section 2 introduces some relative works of masked face recognition; Section 3 illustrates the proposed algorithms used to fulfill this task; Section 4 shows the experiments we did to evaluate the effectiveness of our reconstruction method; Section 5 is the conclusion of the whole paper.

2. Related work

Nowadays, the mainstream of masked face recognition is first extracting the features of the unmasked area of the face, then applying some machine learning algorithms like Support Vector Machine (SVM) or Neural Network on them [1-6]. In [7], the authors used two pre-trained deep learning models to transform an input picture to a 128D embedding, which was then used to train the SVM classifier. [8] has the same idea as [7], except the authors in [8] chose FaceNet to extract the feature (that is, to create the 128D embedding). In [9], the authors used 2D Discrete Wavelet Transform (2D-DWT) to decompose the training samples so they could filter the high-frequency signals and just retain the low-frequency signals. In order to construct the low-dimensional signals, Principal Component Analysis (PCA) was used, followed by sparse decomposition utilizing the occlusion dictionary, and finally classification utilizing the occlusion dictionary [9]. Because the traditional PCA was too time-consuming for the authors of [10], they chose Kernel Principle Component Analysis (KPCA) as their feature extractor. They first divided an input face image into several different regions, and then used a new logarithmic transformed Gaussian error operator to calculate the weight of each area. Finally, they applied KPCA to each region to extract the features and classify the input faces. In [11], the authors suggested an attention-aware masked face recognition called AMaskNet, which incorporates a feature extractor and a contribution estimator. They used ArcFace34 as their main feature extractor and BN-Conv-BN-PReLU-Conv-BN as an aid, so they were able to receive 512D features of each picture. And their contribution estimator included (1) a spatial contribution estimator helping them to get greater weights in useful area (that is, the area not been masked), and ignore the region been occluded; and (2) a channel contribution estimator which stipulated the model to pay more attention on useful channels

DOI: 10.54254/2755-2721/6/20230846

of the feature map. In [12], the authors tried to recognize a masked face by extracting the 31-landmarks around the eyes and eyebrows. A recent study suggests that by fine-tuning existing state-of-the-art face models using masked images [13], better performance can be achieved than when using periocular-based models.

However, no matter what methods they used, the above papers both chose the original masked faces as the input. Nevertheless, in recent years, some researchers have been interested in face reconstruction [14], which is also the key point of this paper. The core idea of face reconstruction is using Eigenfaces to reconstruct the area that used to be occluded, and it can be done with the help of PCA. In [15-16], the authors used Fast Recursive PCA to reconstruct the face. They used the output (an image) of the previous round of PCA as the input for the next round of PCA. The iteration will be terminated if the distance between the input and the output in one round is less than a given value, and the distance meant the Euclid norm (l2 norm) value between those two pictures. In [17-19], the authors reconstructed the occluded part of the face by Gappy PCA. An intuitive paraphrase of Gappy PCA is applying PCA only on the non-occluded area of the original picture to create a new image, and then minimizing the distance (l2 norm value) between the new image and the non-occluded area of the original picture. Doing this could obtain a set of coefficients, which was used to create the occluded part by multiplying each coefficient with its corresponding Eigenface and then adding the mean face. However, due to the massive portion of masked area, Gappy PCA always has to spend plenty of time computing useless values in each iteration. The reason why these useless values form is that, during each iteration, Gappy PCA uses whole Eigenfaces (of course including the masked area) to create the new face and discards the value of the masked area when computing the distance. This strategy surely wastes a lot of time. To overcome this shortcoming, during each iteration of the reconstruction process, we merely reconstruct the unmasked area rather than the whole image. After the reconstruction, we can use the parameters that were made during the reconstruction to make the whole image.

3. Method

3.1. General thought

The tasks of this paper can be divided into two sections: first converting masked images to unmasked images (Reconstructing Process), and then using the unmasked image to identify who is that person (Identifying Process). Before the Reconstruction Process, we have to use Principle Components Analysis (PCA) to extract the features (Eigenfaces) from the training dataset. PCA is an efficient method not only in terms of reducing dimension, but it can also be used to extract features (e.g., Eigenfaces, in the field of face recognition) from the training dataset. Those Eigenfaces are key to the Reconstruction Process.

In the Reconstructing Process, we first convert the masked colored image to a grayscale image and resize its shape to a square. The reasons we use the grayscale image in this paper are: first, the grayscale image is convenient to compute the 'MEAN' in the 2nd line in algorithm 1. Besides, in this task, color information is unimportant and negligible. Therefore, to save time in both Reconstructing Process and Identifying Process, we ignore the color and use grayscale to represent them. Then, we detect the unmasked area, which is achieved by algorithm 1, and use the unmasked area to reconstruct the masked area, which is achieved by algorithm 2.

After the Reconstruction Process, we can obtain the unmasked reconstruction image of each masked image, so we are able to enter the Identifying Process. In this section, we select the Convolutional Neural Network as our identification algorithm. CNN is a widely used machine learning algorithm, especially in the field of image classification and recognition, because it is able to extract deep and hierarchical features from the input data. Therefore, by delivering the unmasked reconstruction images to CNN, we can receive the accuracy of being classified correctly.

DOI: 10.54254/2755-2721/6/20230846

3.2. Algorithm implementation

3.2.1. Reconstructing process. We propose two algorithms in this section: algorithm 1, whose pseudo code is written in Algorithm 1, aims to detect and extract the masked area from the masked image. With the masked area we detect, we can also obtain the unmasked area, which is then used as the objective we try to fit using Eigenfaces in algorithm 2 whose pseudo is written in Algorithm 2. We choose the Euclid Norm (l2 norm) as the criterion to evaluate the effect of the fitting process.

In algorithm 1, we first record the beginning time as t1, which is then used to compute the running time. We define 'MEAN' as the metric to identify where the masked area is. That is, first we identify the type (i.e., the color) of masked image (white or black), and then by comparing the value of each pixel of the masked image with MEAN, we can extract the masked area. In this paper, all masked images have the mask only occurring in the lower half part of those images, so we can solely apply algorithm 1 to the lower half part of each masked image. To represent the masked area, we use matrix A. A(i,j)=0 indicates that img(i,j) is not the masked area, and A(i,j)=1 indicates that img(i,j) is the masked area. After that, we can deliver the masked area, which is represented by matrix "A" in algorithm 1 to algorithm 2 for further processing.

Algorithm 1. Masked_Area_Detection (img)

Input: img
Output: A, t1
1.Recording current time as t1
2.We set MEAN equals to the mean value of the lower half part of img, where the lower half part of img means img[m/2:m,0:n]
3.Determining the type (i.e. color) of mask on the img based on MEAN
4.if img(i,j) is the masked area:
5. A(i,j)=1
6.else:
7. A(i,j)=0
8.return t1, A

To prove this way of detecting masked areas is simple but practicable, we pick up some examples and visualize them, as shown in figure 1:

Figure 1. Some examples of the output matrix 'A' representing the masked area in algorithm 1. (a) and (c) are two masked images whose mask' color are white and black, respectively. (b) and (d) are the output matrix 'A' given the input of algorithm 1 is (a) and (c), respectively. In (b)

From figure 1, we can see that main part of the mask can be detected and extracted correctly, while a small part of the mask may fail to be recognized successfully. This is because some pixels of the mask are not in line with other pixels of the mask. Like in (a) of figure 1, the pixel value of the pure white part is extremely high (which almost reaches to 1), so the 'MEAN' is high due to those pure

DOI: 10.54254/2755-2721/6/20230846

white parts, whereas the pixel value of the other deep white parts (the part covered by shadow) is lower, so they may not be detected as the masked part. Nevertheless, most parts of the mask can still be detected accurately. Besides, this method of detection is very quick (less than 1 second per image).

The accurate detection of the masked area lays down a solid foundation for further processing, i.e., the reconstruction operation, whose pseudo code is written in the table 2. In algorithm 2, we use N to represent the number of features we use during PCA, which also means after PCA, we can obtain N Eigenfaces from the training dataset, which consists of M images. Note that M must be greater than N. The most important part of algorithm 2, which is also the core idea and the main contribution of this paper, is using PCA to reconstruct the masked image. We first determine where the unmasked part is by matrix "A" and save it as img_vector, as shown in the 2nd line of algorithm 2, and then extract the counterpart of each Eigenfaces (i.e. maintain Eigenfaces (i, j) if A (i, j) =0, and discard Eigenfaces(i,j) if A(i,j)=1). We use Eigenfaces_vector to represent the Eigenfaces after being extracted in the 3rd line of algorithm 2. After that, we use Eigenfaces_vector to fit img_vector. The objective function is Euclid Norm and the optimization algorithm is gradient descend, where the termination condition of gradient descend can be altered due to different situations and requirements, and the learning rate could also be changed. After the iteration of gradient descend, we can receive a group of values which is represented by vector "a", which is updated in the 10th line for each round of iteration. Finally, we are able to reconstruct the masked image with vector "a" and Eigenfaces from the 11th line to the 15th line in algorithm 2. At last, we also compute the "time_costing", representing the total running time of algorithm 1 and algorithm 2, and the "distance", meaning the Euclid Norm Value between img_ori and img_reconstruction, which are both used to evaluate our method's effect in the subsequently Experiment Section.

Algorithm 2. Masked_Face_Reconstruction (img, img_ori, A, t1)

Input: img, img_ori, A, t1
Output: distance,time_costing,img_reconstruction
1.Applying PCA on M training images to generate N Eigenfaces, written as Eigenfaces
2.Saving the unmasked area of img as img_vector
3.Saving the unmasked area of Eigenfaces as Eigenfaces_vector
4.We use da to represent the derivative of objective function which can be found in 8th line
5.for (the max absolute value in da is greater than termination condition, like 1e-4 or 1e-5):
6. Generating img_fit with the linear combination of a group of parameters 'a' and Eigenfaces
7. img_diff=img-img_fit
8. loss= (Computing_Euclid_Norm_Value(img_diff, img_diff))^2
9. da=2*Computing_Euclid_Norm_Value(img_diff, Eigenfaces_vector)
10. a=a-learning_rate*da
10.end for
11.Generating img_reconstruction with the linear combination of a and Eigenfaces
12.for i,j:
13. if(A(i,j)==0):
14. img_reconstruction(i,j)=img(i,j)
15.end for
16.Recording current time as t2
17.time_costing=t2-t1
18.distance=Computing_Euclid_Norm_Value(img_ori, img_reconstruction)
19.return distance, time_costing, img_reconstruction

3.2.2. Identifying process. The previous Reconstruction Process ensures we obtain the reconstruction image (i.e., the unmasked image) for each masked image. Therefore, we can apply deep learning to help us identify the class of the reconstruction images, which is also the class of the original masked image. Deep learning is a very powerful and widely used machine learning method, especially with the rapid development of big data technology and the update of computer hardware configuration. We

DOI: 10.54254/2755-2721/6/20230846

are able to use deep learning to solve a variety of problems requiring high computer power in reality, like face recognition and natural language processing, etc. In this paper, we choose CNN, which stands for Convolutional Neural Network, as our classification algorithm.

To improve the performance and robustness of our model, we introduce two methods in this section: data augmentation and dynamic learning rate. Data augmentation is used to improve the classification accuracy and eliminate overfitting by exerting rotation, zoom in or zoom out, and other similar operations upon the training images randomly before each training epoch. Dynamic learning rate aims to adjust the learning rate automatically during the training process. To sum up, our strategy is randomly rotating 0 to 10 degrees clockwise or anticlockwise and zooming in (or out) 0.8 to 1.2 times for each training image before each epoch of the training process, and multiplying the learning rate by 0.8 if the historical highest accuracy of consecutive 10 epochs is not updated. The initial learning rate equals 1e-2.

4. Experiments and verification

4.1. Experiment settings

4.1.1. Computer configuration. The configuration of the computer which runs all the algorithms in this paper is listed in table 1 below:

Table 1. Computer configuration.

Operating System	Windows 10 64 bit
Processor	Intel Core i7-8750H @ 2.20GHz six cores
Main Board	ASUS FX505GM
Memory	8 GB
CPU	Nvidia GeForce GTX 1060 6 GB

4.1.2. Running platform. We use two different platforms to write and run Python codes: Jupyter Notebook and Kaggle Notebook.

Jupyter Notebook is a very useful and convenient tool in which you can write and run Python codes, and especially, you can run different parts of your codes accordingly. We use Jupyter Notebook to generate the reconstruction images from masked images.

Kaggle Notebook is suitable to run the program which includes a neural network since you can use Kaggle's GPU (NVIDIA TESLA P100) to accelerate that part of the program, and just as they say: "These GPUs are useful for training deep learning models, though they do not accelerate most other workflows (i.e., libraries like pandas and scikit-learn do not benefit from access to GPUs)". Therefore, we use Kaggle Notebook to train our CNN model, whose inputs are reconstruction images and output is the accuracy of being classified correctly.

4.2. Dataset description

Figure 2. Some examples of RMFRD. (a) and (c) are two people in RMFRD, (b) and (d) are the masked image of (a) and (c).

DOI: 10.54254/2755-2721/6/20230846

We select the Real World Masked Face Recognition Dataset (RMFRD) [20-21] as our primary dataset. This dataset contains 5,000 masked faces of 525 people and 90,000 normal faces. We use unmasked images as the training dataset and masked images as the validation dataset. Following figure 2, which shows some examples of RMFRD.

Besides the RMFRD, we also create some masked images manually as the complementation of our validation dataset. Creating masked images manually is not an innovative idea since [11,13,22] have already applied this method. We search for the photos of people who are present in our training dataset using the internet, then we add a mask to their faces using Photoshop. Figure 3 shows some examples of the photos which are collected from the internet and their masked counterparts. The color of masks in our final validation dataset are either black or white.

Figure 3. Some examples of manually generating masked images. (a) and (c) are two images we collected from internet, (b) and (d) are the masked images created manually via Photoshop.

4.3. Performances and evaluations

By delivering the masked image to algorithm 1 and algorithm 2 sequentially, we can receive its unmasked reconstruction counterpart as the output, which is the core idea and the main contribution of this paper. The effect of output is mainly determined by two parameters: n_components and the termination condition, which you can find in the 1st line and 5th line of algorithm 2, respectively. At the end of this section, we also try to find the relationship between the effect of the output and the shape of the input images. To illustrate the high efficiency of our method, we plot the following figures.

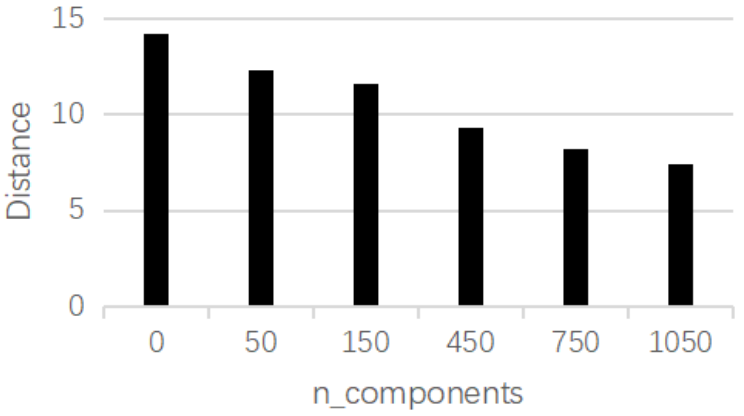

Figure 4. The relationship between distance and n_components. n_components=0 represents the distance between masked images and their original images.

DOI: 10.54254/2755-2721/6/20230846

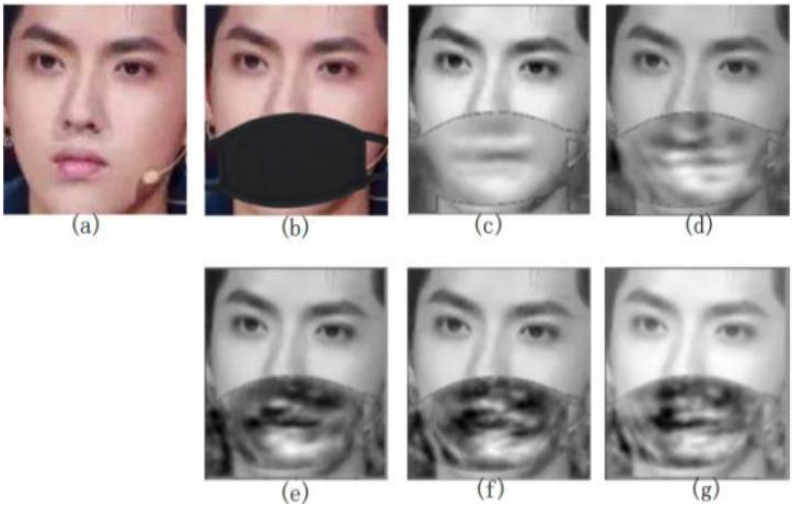

Figure 5. A group of examples (c-g) of being reconstructed through algorithm 1 and algorithm 2, given the input is (b), where (b) is the manually generating masked image of (a).

4.3.1. The effect of output given different value of n_components. Figure 4 shows the relationship between distance and n_components, where distance means the Euclid norm value of two images [4], which can be derived from the output of algorithm 2. (Note that here the distance on the longitudinal axis of figure 4 refers to the mean distance of all images, but the output of algorithm 2 is the distance of a single image. Therefore, to compute the mean distance, we can write down all the distances of different images and add them up, and then divide them by the number of images.), and n_components means the number of features we used to extract from the training images through PCA.

Table 2. The value of n_components we used to generate (b-g) in figure 5, and the distance between (a) in figure 5 and (b-g) in figure 5, respectively.

image (i) in figure 5	(b)	(c)	(d)	(e)	(f)	(g)
the value of n_components to generate (i)	0	50	150	450	750	1050
distance between (a) and (i)	15.1	13.38	11.75	7.75	7.14	6.71

In algorithm 2, we use N in the first line to represent n_components.

To intuitively illustrate the meaning of distance, we visualize them (more specifically, we want to see the difference between two images owing to the different value of distance). As shown in figure 5, we select a group of examples from our consequences, and the value of n_components we used and the distance are both listed in table 2. We also have to emphasize that before computing the distance, all images should be resized to 150*150 and converted to grayscale.

Again, n_components=0 means not applying PCA in masked images since we want to receive the distance between the masked image and its original image. In figure 5, this distance refers to the distance between (a) and (b). For those masked images which do not have original images, like the masked images in RMFRD, we randomly choose an unmasked image from the training dataset as the original image to compute the distance.

Although it is still difficult to identify the difference with our eyes, our CNN model can tell us that difference. By delivering (b-g) in figure 5 to the fine pre-trained CNN model, we can obtain the accuracy of being classified correctly as: 0.751; 0.799; 0.826; 0.882; 0.904; 0.932, which clearly shows the high efficiency of our method.

DOI: 10.54254/2755-2721/6/20230846

From the different values of n_components, we can obtain the different reconstructed images of the same masked image from the output of algorithm 2, like (c-g) in figure 5. By delivering those different reconstructed images to the CNN model, we can achieve the accuracy of being classified successfully, as shown in figure 6, and the configuration of the CNN model is listed in table 3.

Table 3. CNN configuration. The convolutional layer parameters are denoted as "conv<receptive field size>-<number of channels>". The ReLU activation function is not shown for brevity.

input (150*150 grayscale images)
conv3-256
maxpool
conv3-256
maxpool
conv3-512
maxpool
conv3-512
maxpool
flatten
dropout
FC-1024
FC-1024
FC-5
soft-max

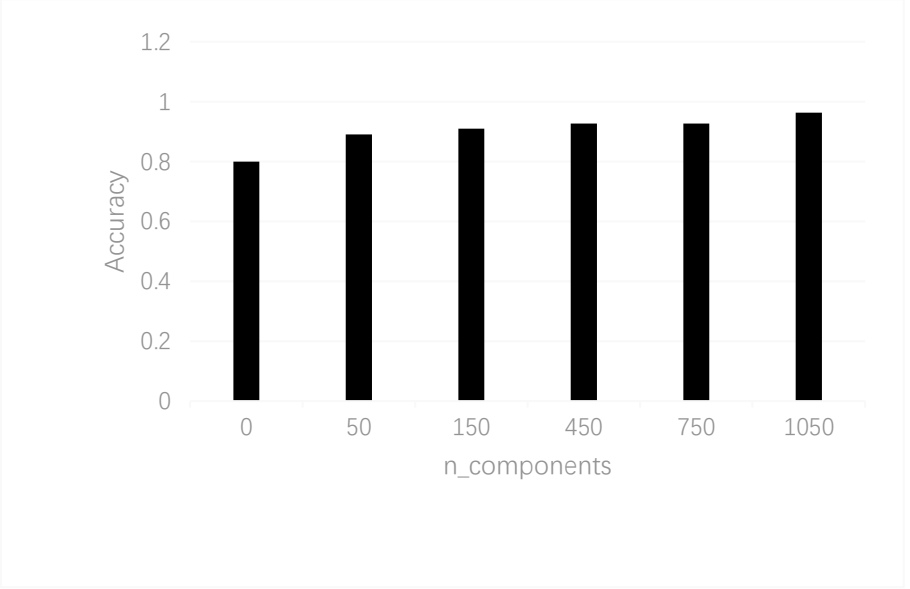

Figure 6. The relationship between accuracy and n_components. n_components=0 means using masked images as the input of CNN model.

From figure 6, we can see the highest accuracy equals 0.9636 when n_components equals 1050. Figure 7 is the training curve whose highest validation accuracy equals 0.9636:

DOI: 10.54254/2755-2721/6/20230846

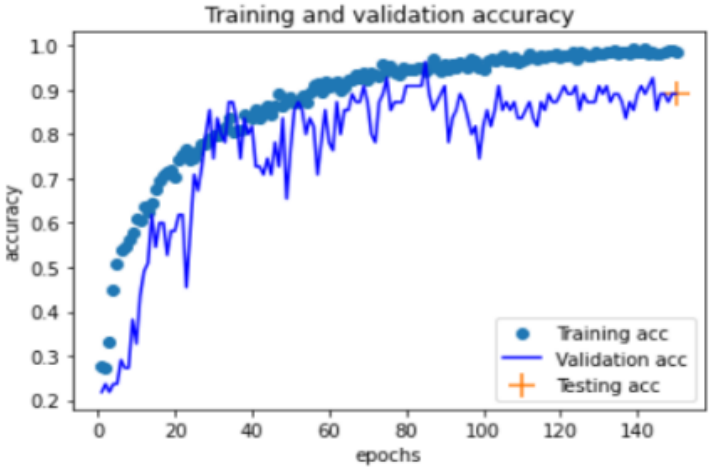

Figure 7. The training curve whose highest validation accuracy equals to 0.9636, which occurs in the 85th epoch.

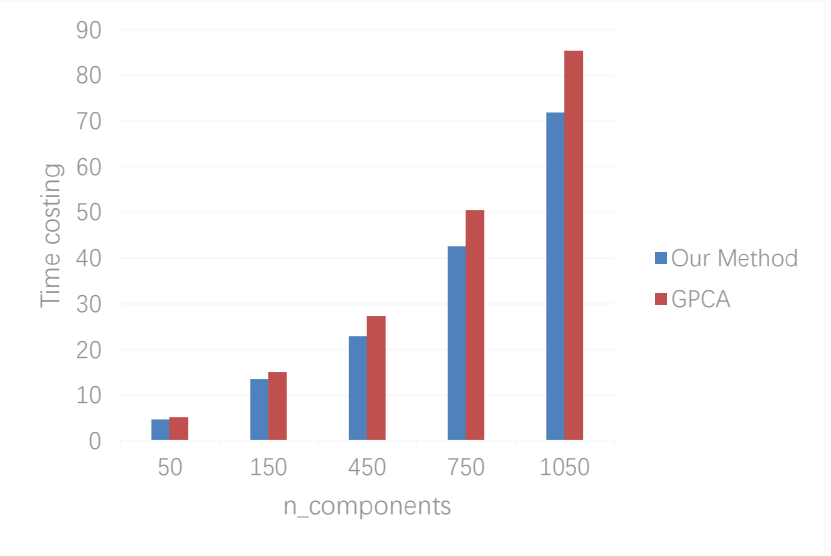

Figure 8. The relationship between time costing and n_components, and the difference between two different methods.

Besides the distance and accuracy, time cost is another important metric to evaluate our method. As I mentioned in the introduction section, GPCA is another welcome algorithm in the field of occluded image reconstruction. I said GPCA is time-consuming as well. To demonstrate this assertion, I compute the average running time of consecutive running algorithms 1 and algorithm 2 for all masked images as the input, given the different values of n_components. To compute the average running time, or the time cost, we can add the time_costing (the output of algorithm 2) for each image and divide it by the number of images. Then we can compute the time cost of our method. To obtain the time costing of GPCA, we could just change algorithm 2 with the thought of GPCA, which can be found in [2,22-23]. figure 8 shows the relationship between time costing and n_components, and the difference between two different methods. From figure 8, we can conclude the time costing of our method is less than GPCA, and with the increment of n_components, the difference becomes much bigger, which clearly shows our method is better than GPCA in terms of time costing. It is reasonable since during the gradient descent algorithm, in each round, GPCA will compute almost half of the useless values (the masked area), which surely wastes a lot of time.

DOI: 10.54254/2755-2721/6/20230846

4.3.2. The effect of output given different value of termination condition. Besides the n_components, another important parameter which determines the effect of reconstruction is the termination condition, which can be set up in the 5th line of algorithm 2. Here we use dx to refer to the termination condition, but don't confuse it with the da in the algorithm 2. The da in the algorithm2 means the derivative of the loss function, which you can find in the 9th line of algorithm 2, but the dx here we used as the horizontal axis to plot figure 9-10 means the termination condition is related to the derivative. Like figure 4, figure 6, and figure 8, we plot figure 9, figure 10, and figure 11, which indicate the relationship between distance and dx, accuracy and dx, and time cost and dx, respectively:

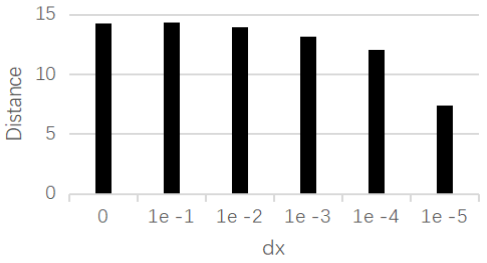

Figure 9. The relationship between distance and dx. dx=0 means not applying reconstruction.

Figure 10. The relationship between accuracy and dx. dx=0 means not applying reconstruction.

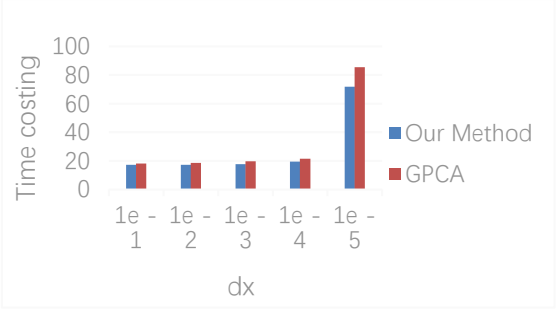

Figure 11. The relationship between time costing and dx.

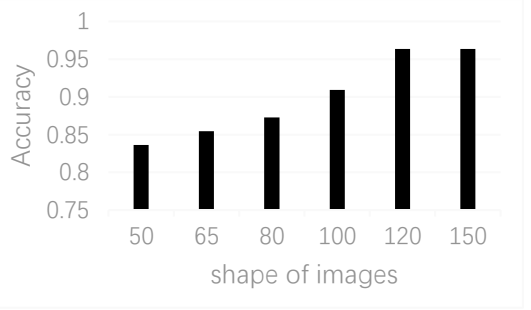

Figure 12. The relationship between accuracy and the shape of input images.

From figure 11, we can see that with the decrement of dx, the time costing will increase slightly at the beginning, but as dx continues to decrease, the time costing will drastically increase. As we know, the longer the time cost is, the better the effect of output we can get. Therefore, although the time costing is much higher when dx =1e-5, its accuracy and distance are much better than other conditions (e.g., dx =1e-4 or 1e-3), and that can be derived from figure 9 and figure 10. However, dx can not be decreased unboundedly, considering the machine power we have, and it is also unnecessary to keep dx decreasing since the effect of output is already good enough when dx is tiny enough, so it does not need to be optimized anymore. If we set dx =1e-6, then the time cost nearly approaches infinity, which means we will never receive the output.

4.3.3. The effect of output given different shape of input. Figure 12 illustrates the relationship between accuracy and the shape of images, where the shape of images refers to the shape of img (img is the input of algorithm 1 and algorithm 2). For instance, if the shape of images equals 120, then the shape of an img is 120*120, which means one img is represented by 14400 pixels. The shape of images (also the shape of img) can be changed manually before inputting img to the algorithm 1. The shape of the

DOI: 10.54254/2755-2721/6/20230846

img is supposed to be square. From figure 12, we can see in this paper, the shape of images equals 120 is enough.

5. Conclusion

In this paper, we propose a reconstruction method to accomplish the masked face recognition task by first converting the masked image to an unmasked image and then using a Convolutional Neural Network (CNN) to ensue the face recognition process. This reconstruction method can basically be divided into two sections: first detect the unmasked area of the masked image, then use the Eigenfaces acquired from Principle Components Analysis (or PCA) combined with a set of parameters to generate a new image to fit the unmasked area of the masked image. Experimental results clearly demonstrate the high effectiveness of our method. Compared to directly utilizing the original masked images to exert face recognition, our method can increase the accuracy of being classified correctly from about 80% to 96.36%, which is obviously great progress.

Although the accuracy seems pretty good, there are still some limitations in our method. First, we are not able to extract the fine edge of the masked area from the masked image. Besides, our dataset is not large enough to support more accurate reconstruction results. Finally, we didn't try to use any feature extractors (like Face-Net) to extract the features again before delivering the reconstruction images to the CNN model. Therefore, our further work will focus on solving the problems above.

References

[1] Ejaz M S, Islam M R, Sifatullah M, et al. 2019 Implementation of principal component analysis on masked and non-masked face recognition[C]//2019 1st international conference on advances in science, engineering and robotics technology (ICASERT). IEEE,pp 1-5.

[2] Liu Y, Zheng W. 2021 Masked Face Recognition based on Attention Mechanism and FaceX-Zoo[C]//2021 International Conference on Digital Society and Intelligent Systems (DSInS). IEEE, pp 107-110.

[3] Huang Y C, Tsao L H, Chen H H. 2022 Robust Masked Face Recognition via BalancedFeature Matching[C]//2022 IEEE International Conference on Consumer Electronics (ICCE). IEEE, pp 1-6.

[4] Mhadgut S. 2021 Masked Face Detection and Recognition System in Real Time usingYOLOv3 to combat COVID-19[C]//2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT). IEEE, pp 1-7.

[5] Hsu G S J, Wu H Y, Tsai C H, et al. 2022 Masked Face Recognition From Synthesis to Reality[J]. IEEE Access, 10: 37938-37952.

[6] Dai W, Wang J, Ren T, et al. 2022 Face Mask Recognition Based on YOLOv3-tiny[C]//2022 3rd International Conference on Electronic Communication and Artificial Intelligence (IWECAI). IEEE, pp 507-511.

[7] Poornima P D, Singh P N. 2021 Masked & Unmasked Face Recognition Using SupportVector Machine Classifier[C]//2021 IEEE International Conference on Mobile Networks and Wireless Communications (ICMNWC). IEEE,pp 1-4.

[8] Ejaz, Md & Islam, Md. 2019 Masked Face Recognition Using Convolutional Neural Network. Pp 1-6.

[9] Jianxin Z, Haoran L. 2020 Local occluded face recognition based on 2D-DWT and sparse representation[C]//2020 5th International Conference on Mechanical, Control and Computer Engineering (ICMCCE). IEEE,pp 2110-2114.

[10] Chen X, Wang S, Ruan X. 2016 Recognition of partially occluded face by error detection with logarithmic operator and KPCA[C]//2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI). IEEE,pp 460-464.

[11] Zhang M, Liu R, Deguchi D, et al. 2022 Masked Face Recognition With Mask Transfer and Self-Attention Under the COVID-19 Pandemic[J]. IEEE Access, 10: 20527-20538.

[12] Shahar M S M, Mazalan L. 2021 Face identity for face mask recognition system[C]//2021 IEEE

DOI: 10.54254/2755-2721/6/20230846

11th IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE). IEEE, pp 42-47.

[13] Martínez-Díaz Y, Méndez-Vázquez H, Luevano L S, et al. 2021 Towards Accurate and Lightweight Masked Face Recognition: An Experimental Evaluation[J]. IEEE Access, 10: 7341-7353.

[14] Malakar S, Chiracharit W, Chamnongthai K, et al. 2021 Masked face recognition using principal component analysis and deep learning[C]//2021 18th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON). IEEE, pp 785-788.

[15] Wang Z M, Tao J H. 2007 Reconstruction of partially occluded face by fast recursive PCA[C]//2007 International Conference on Computational Intelligence and Security Workshops (CISW 2007). IEEE, pp 304-307.

[16] Park J S, Oh Y H, Ahn S C, et al. 2005 Glasses removal from facial image using recursive error compensation[J]. IEEE transactions on pattern analysis and machine intelligence, 27(5): 805-811.

[17] Bagchi P, Bhattacharjee D, Nasipuri M. 2014 Robust 3d face recognition in presence of pose and partial occlusions or missing parts[J]. arXiv preprint arXiv:1408.3709.

[18] Colombo A, Cusano C, Schettini R. 2008 Recognizing faces in 3d images even in presence of occlusions[C]//2008 IEEE Second International Conference on Biometrics: Theory, Applications and Systems. IEEE,pp 1-6.

[19] Alyüz N, Gökberk B, Spreeuwers L, et al. 2012 Robust 3D face recognition in the presence of realistic occlusions[C]//2012 5th IAPR International Conference on Biometrics (ICB). IEEE, pp 111-118.

[20] Utomo Y, Kusuma G P. 2021 Masked Face Recognition: Progress, Dataset, and Dataset Generation[C]//2021 3rd International Conference on Cybernetics and Intelligent System (ICORIS). IEEE, pp 1-4.

[21] Huang B, Wang Z, Wang G, et al. 2021 Masked face recognition datasets and validation[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. pp 1487-1491.

[22] Hariri W. 2022 Efficient masked face recognition method during the covid-19 pandemic[J]. Signal, image and video processing, 16(3): 605-612.

[23] Schroff F, Kalenichenko D, Philbin J. 2015 Facenet: A unified embedding for face recognition and clustering[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. pp 815-823.