Masked face recognition with convolutional neural networks and local binary patterns

Hoai Nam Vu[1] · Mai Huong Nguyen[2] · Cuong Pham[1]

Accepted: 26 July 2021 / Published online: 14 August 2021

Abstract

Face recognition is one of the most common biometric authentication methods as its feasibility while convenient use. Recently, the COVID-19 pandemic is dramatically spreading throughout the world, which seriously leads to negative impacts on people's health and economy. Wearing masks in public settings is an effective way to prevent viruses from spreading. However, masked face recognition is a highly challenging task due to the lack of facial feature information. In this paper, we propose a method that takes advantage of the combination of deep learning and Local Binary Pattern (LBP) features to recognize the masked face by utilizing RetinaFace, a joint extra-supervised and self-supervised multi-task learning face detector that can deal with various scales of faces, as a fast yet effective encoder. In addition, we extract local binary pattern features from masked face's eye, forehead and eyebow areas and combine them with features learnt from RetinaFace into a unified framework for recognizing masked faces. In addition, we collected a dataset named COMASK20 from 300 subjects at our institution. In the experiment, we compared our proposed system with several state of the art face recognition methods on the published Essex dataset and our self-collected dataset COMASK20. With the recognition results of 87% f1-score on the COMASK20 dataset and 98% f1-score on the Essex dataset, these demonstrated that our proposed system outperforms Dlib and InsightFace, which has shown the effectiveness and suitability of the proposed method. The COMASK20 dataset is available on https://github.com/tuminguyen/COMASK20 for research purposes.

Keywords Face recognition · Local binary pattern · Masked face recognition

1 Introduction

COVID-19 is changing the lives of millions of people around the world. Every day the number of deaths due to the pandemic gradually increases without any sign of reaching a peak [1]. Many countries are suffering from the third wave of the pandemic with a worse level than the previous periods in both numbers of people infected and deaths. The healthcare systems of many countries are becoming increasingly overloaded in the unprecedented pandemic of the human history. As known, the COVID-19 is a dangerous disease because it spreads quickly within the community through direct human-to-human contact. There are many proposed solutions to fight against COVID-19 viruses such as social distancing, vaccine preparation, and contactless operations in public space. Among these solutions, contactless operations are encouraged in many countries, especially in public areas like airports, supermarkets, and subway stations. Wearing masks in these public spaces is necessary to prevent the spread of this horrible viruses. Specifically, in some countries, like Vietnam, China, India, a person will be fined for not wearing a mask in public. At the beginning of the pandemic, wearing masks was controversial in many countries. Thereafter, it can be known that wearing masks is one of the lowest-cost yet effective ways to prevent the

This article belongs to the Topical Collection: *Artificial Intelligence Applications for COVID-19, Detection, Control, Prediction, and Diagnosis*

✉ Cuong Pham
cuongpv@ptit.edu.vn

Hoai Nam Vu
namvh@ptit.edu.vn

Mai Huong Nguyen
huongnm.ptit@gmail.com

[1] Department of Computer Science, Posts and Telecommunications Institute of Technology, Hanoi, 12110, Vietnam

[2] Department of Computer Vision, Aimesoft., JSC, Hanoi, 11310, Vietnam

people from infection. However, the performance of the face recognition system, a means to support contactless operation, might significantly decrease because masks cover a large area of the face including lips and nose, resulting in a large number of features that cannot be extracted. Therefore, it is necessary to investigate to improve the performance of the masked face recognition system during the pandemic.

Masked face recognition is a branch of occluded face recognition with prior knowledge about the targeted face's occluded area. Occluded face recognition is an active research scenario that attracted the computer vision research community. Previously, occluded face recognition systems have focused on detecting and recognizing an individual's face in the wild where the occluded area of the face is in random shape and position. Meanwhile, a masked face is often obscured the nose, mouth and cheeks area. The remaining unobstructed regions might be the eyes, eyebrows, and forehead. Therefore, a masked face recognition system might effectively focus on the analyses of features that can be extracted from the areas including the eyes, eyebrows and forehead, which are uncovered by the mask, of the subject.

With the rapid development of deep learning methods in the computer vision domain, face recognition has been considered the solved problem in the reasonable condition with the recognition accuracy approaches to approximate 100%. However, methods of analyzing textual and shape features used for facial recognition are still in progress investigated for proposing approaches to face recognition in specific domains. These feature types show their strengths in their processing time without requiring a large amount of training data. In this paper, we propose a deep ensemble model for masked face recognition that combines deep learning feature extraction at the first stage and Linear Binary Patterns (LBP) textual feature extraction at the second stage. The LPB textual feature extraction utilize local features on the eyes, eyebrows, and forehead areas of the subject. In addition, we verify our proposed method on two datasets, one is published the Essex dataset and the other is the our self-collected COMASK20 dataset. The remainder of this paper is organized as follows. Section 2 reviews the related works. Section 3 describes the proposed system. Section 4 reports on experimental results. Future research directions and a discussion are provided in Section 5.

2 Related works

Face recognition is one of the most critical problems of computer vision area as it has a wide range of application real-world. In addition, face recognition can usually be used as biometric identification and verification. Many studies have been proposed to solve the problem of face recognition. However, in some specific environmental conditions, the existing face recognition system's performance is far from expectation. The main challenging issue ranges from pose changes, illumination, aging, expression, and especially occlusion. Compared to the above problems, facial occlusion is the one that most commonly occurs in practice due to environmental conditions, camera angles, and subject activity. Besides, occluded face recognition remains an unsolved problem because a part of essential features is disappeared, leading to higher inter-class similarity and intra-class variation.

2.1 Non-occluded face recognition

Non-occluded face recognition is described as face recognition in common conditions where the subject's faces are fully visible in the images. A typical face recognition consists of four main steps: face detection, facial landmark detection, facial features extraction, facial features classification. Face detection is a process utilized to estimate the bounding box of the face in a given image or video frame. If there is one more face in the images, all of them should be detected. Face detection is one of the most important steps which stacked at first place in the face recognition pipeline. Therefore, to ensure the good performance of face recognition, the face detection algorithms need to be robust to pose, illumination changes, and scale differences. Besides, face detection should eliminate background noise as much as possible [2]. Many face detection algorithms have been proposed in the literature. The Viola-Jones face detection [3] was proposed to use Haar-like features for frontal faces in real-time. However, the viola-jones face detection is not robust to pose, illumination changes, and especially occlusion. Color features have also been implemented to detect the face in the images [4–6].

Recently, deep learning has boosted up the object detection performance which leads to successful deep learning-based face detector [2, 7–10]. MTCNN (Multitask cascaded convolutional networks) [62] face detector is a framework widely used in many practical applications. MTCNN leverages a cascaded architecture with three stages of deep convolutional networks to predict landmark location in the coarse-to-fine domains. MTCNN outperformed classical face detector by a large margin in various benchmarks. [11] proposed a method based on cascaded convolutional networks to address the problem of fixed-size input images and weak generalization ability of the existing methods. They achieved competitive accuracy to the state-of-the-art architectures while being able to recognize the human face in real-time. One of the crucial steps in the face recognition pipeline is facial landmark detection, which tries to extract important points on the face, such as corners of the eyes, eyebrows, mouth, etc.). These points are then the input of the face alignment. Aligning the face to the fixed view has shown the advantages for the face recognition accuracy. Other methods were proposed to perform face alignment procedures [12–14]. In order to evaluate the facial landmark detection algorithm, the ground-truth based localization error was implemented by [15]. Due to the development of deep learning techniques, the performance of facial landmark detection methods has been improved significantly [16–18]. The third step in the face recognition pipeline is facial features extraction. Prior to deep learning, facial features were extracted by geometric, texture, and shape properties of the subject's face. In [19], the author extracted local binary pattern features in order to recognize facial expression from videos. In [20], the author evaluated the efficiency of geometric shaped facial features for face recognition. They designed a model for identifying the exact person by finding the center and corners of the eye using an eye detection module. After the facial feature extraction stage, some classifier algorithms are applied to obtain the final recognition results [21].

Deep feature extraction refers to a group of methods that takes advantages of deep learning models to extract important facial features instead of handcrafted features. One of the most important tasks to design a deep network for feature extraction is the network architecture and loss function. For the face recognition, the softmax loss is not sufficient for separating the facial features. Therefore, other kinds of loss function for facial feature extraction were proposed such as Euclidean-distance based loss [22, 23], triplet loss [24], and variation of softmax loss [25, 26]. The deepface works used Alexnet as the base network with the softmax function achieved 97.35% on the Facebook dataset in 2014. The author of DeepID2 used alexNet and contrastive loss to achieve 99.15% accuracy on the Celeb Faces. The method proposed by [27] reached 99.83% verification performance on LFW with the ResNet-100 architecture. Their work investigated new ideas of arcface loss function for significantly improving the performance of existing methods.

2.2 Occluded face recognition

Occluded face recognition refers to a branch of methods in which the system has to identify the individual whose face is occluded. Facial occlusion is one of the most challenging problems because of their random positions of occluded parts as the the occluded parts on the subject's face can be arbitrary in position, size and shape. Therefore occluded face recognition often need large datasets to alleviate underfitting. However, it is not feasible to simulate all realistic scenarios of facial occlusions. Therefore, the problem of occluded face recognition remains uncompletely solved in their respective practical applications. The most critical issue caused by occlusion is that the facial appearance changes substantially, which leads to decreased accuracy of facial recognition systems. In many cases, occluded face recognition is based on an occlusion-free face database for training and an occluded face data for testing. The occluded face dataset consists of a collection of real occlusions or synthetic occlusions. The work [28] categorized face recognition techniques under occlusion into three different groups: occlusion robust feature extraction, occlusion aware face recognition, and occlusion recovery based face recognition.

The occlusion robust feature extraction methods can be applied if the extracted features are relatively robust. This group of approach aims to extract features that are less affected by occlusions while remaining the discriminative ability. Local Binary Pattern (LBP) is utilized to represent face image efficiently. LBP and their respective variants are applied successfully to face recognition [29, 30]. The Scale Invariant Feature Transform (SIFT) descriptor is also a useful feature extractor for face recognition [31]. Histograms of Oriented Gradient (HOG) descriptor [32] were proposed to cope with face recognition. The idea of HOG is to characterize the outer shape of the subject's face. The author of [33] proposed a system to incorporate Gabor feature [34] for face recognition problems. The author of [35] uses a graph to represent a face, each point of the graph corresponding to Gabor descriptor extracted from the face's landmarks. The random sampling patch-based method [36] has been proposed to divide the face image into patches equally. After that, an SVM classifier is applied to the selected patch accordingly. The final results were obtained by fusing of SVM modules. In the paper [37], an occluded face recognition method is proposed using SIFT feature descriptors' statistical learning. The estimated probability density is implemented to measure the similarity between two images in the testing data to classify the feature vector. Recently, deep features extractor outperform state of the art feature extractor by a large margin. Face verification has been improved significantly due to the development of the deep CNN model [38–40]. According to the CNN model's deep architecture, we can obtain expected results if a vast training dataset is available to cover all occlusion cases. An alternative solution to the lack of training data is to implement data augmentation. Data augmentation techniques guarantee the features are extracted equally among different classes.

The occlusion aware face recognition is a branch of method that takes occlusion information into account to recognize the face. In [41, 42], a binary classifier is used to search for the occluded area. They first divided the facial area into a non-overlapping sub-area and then trained an SVM classifier to identify whether sub-area is an occluded area or not. The author of [43] proposed a method to use compressed sensing to detect occluded facial area. For recognition, the detected occlusion is excluded to obtain expected results. The author of [44] incorporates deep learning features to deal with the distortion of occlusion caused. They proposed four region-specific tasks to identify whether the left eye, right eye, nose, and mouth are occluded or not. Some methods assume the prior knowledge of occlusion. For example, the eye region is chosen for feature extraction when subjects are wearing masks. The nose and mouth region are chosen for feature extraction when subjects are wearing glasses. The author of [45] proposed a method to extends non-negative matrix factorization to obtain occlusion estimation. The method does not require the position of occlusion parts. In [46], the author proposed a model that adds MaskNet at the traditional CNN model's middle layer. The goal is to represent an image feature with high confidence and to eliminate one distorted by occlusions. The author of [47] proposed a method incorporating the CNN model to learn the correspondence between the occluded area and corrupted features. Their results show promising performance on both synthesized and realistic datasets.

Occlusion recovery based face recognition methods tries to recover full face from the occluded face, allowing the application of a conventional face recognition approach. However, face recognition performance depends on the recovery process heavily. The occlusion recovery based method can be considered as a pre-processing stage for occluded face recognition problem. The face reconstruction processes have been implemented by linear reconstruction, sparse representation classifier, and deep learning. The author of [48] used PCA reconstruction to remove eye occlusions, which appears when people wear glasses. The author of [49] used variants of PCA to detect occlusions and reconstruct occluded face region. The sparse representation classifier is one of the most commonly used tools to deal with occluded face reconstruction. The author of [50] used a linear combination of training samples to represent an occluded face. At the same time, the author of [51] incorporated the prior knowledge of pixel distribution to improve the sparse representation. In [52], the authors proposed a method to take advantage of robust sparse representation to model adjective block occlusion based on tailored score and error images. The error image is defined as the difference between the occluded face and occlusion free face in the training set. Recently, in order to address the problem of occlusion recovery, deep learning has attracted significant attention from the research community. In the paper [53], the author proposed a method which consists of LSTM and autoencoder to model occluded face from both spatial and temporal characteristics. Besides, generative adversarial network (GAN) is considered a powerful tool for blind reconstruction [54, 55]. Especially, the work [56] only used a corrupted image to reconstruct the original facial image. Occlusion aware GAN [57] is proposed to tackle with occlusion problem. The corresponding occluded regions are recovered using a pre-train GAN, which is trained on original non-occluded faces.

3 Proposed method

As depicted in the Fig. 1, the pipeline of our proposed method consists of three main stages: face detection, face embedding and face recognition with LBP features, which are detailed as the followings.

3.1 Face detection

In this work, we take an advantage of RetinaFace [58], an end-to-end system to solve the scaled-image problem in real-time detection. Regarding the trade-off between inference time and accuracy, herein, we utilized MobileNet [59] to reduce parameters while trying to achieve a competitive accuracy. The use of MobileNet in a single-stage detector (RetinaFace) promises a solid lightweight and real-time face detection system.

Dealing with hassle and low-resolution images, RetinaFace utilizes the Feature Pyramidal Networks (FPN) [60] to produce a rich feature representation of the image. More specifically, rich semantic features at all levels are built quickly from a single input image scale with little or not sacrificing representational power, speed, or memory. This approach has significant performance improvements for faces with various scales. In comparison to other methods, which only use one different between the prediction result and the label, there are 4 keys of face localisation that RetinaFace puts focus on: detection, alignment, pixel-wise parsing and 3D dense correspondence regression. Inspired by work by Deng et al. [58], combined 4 factors in a new proposed loss function which is a multi-task loss to obtain a significant improvement in training and evaluation the detection model. Multi-task loss is a sum of 4 sub-losses, which need to be minimized. More specifically, at the first element: face classification, model will be penalized for each false prediction. At the second one: face box regression, it calculates distance between the bounding box coordinates of the predicted face and the ground-truth associated with the positive anchor. Facial landmark regression is similar to the box regression loss. Instead of bounding box, it finds the distance between the predicted five facial landmarks and the labeled ones. Lastly, the dense face regression loss is generated from the difference between the original face and the reconstruction from a mesh decoder. The multi-task loss function is mathematically formulated

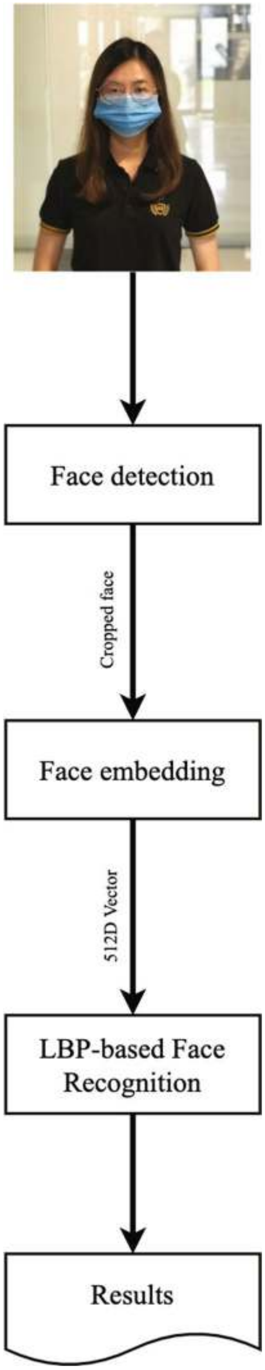


Fig. 1 Proposed method pipeline

as:

$$L = L_{clf}(p_{clf_i}, p_{clf_i^*}) + \lambda_1 p_{clf_i^*} L_{box}(p_{box_i}, l_{box_i}) \\ + \lambda_2 p_{clf_i^*} L_{pts}(p_{pts_i}, l_{pts_i}) + \lambda_3 p_{clf_i^*} L_{pix} \tag{1}$$

where p_{clf_i} is the predicted probability of anchor i being a face, $p_{clf_i^*}$ is 1 for the positive anchor and 0 for the negative anchor, p_{box_i} is the predicted bounding box, l_{box_i} is the labeled box, p_{pts_i} is the set of 5 landmark points given from model's prediction and l_{pts_i} is the annotated ones. λ_1, λ_2, λ_3 are all loss-balancing parameters, which are particularly set to 0.25, 0.1 and 0.01, respectively.

By default, beside the location of face, RetinaFace also outputs a set of 5 landmark points and a dense 3D mapping of points for further reconstruction process. With its flexibility and efficiency, RetinaFace becomes the state-of-the-art of face detection system to work with different image resolutions and conditions. In particular, RectinaFace can handle more tiny or occluded faces with higher scores. Besides, though MTCNN has a good detection performance, it requires GPUs for the prediction speed-up to be appropriate for real-time processing. The MTCNN's frame rate is relatively low (approximately 16 FPS) when it runs on a CPU [61], whereas, Retina can run real-time with up to 60 FPS on a single CPU core for VGA-resolution images by using a lightweight backbone network [58]. Therefore, as a great utility, RetinaFace can boost any face recognition algorithms by replacing MTCNN [62] in alignment and detection tasks. In this study, we utilize original RetinaFace for face detection stage.

In the next section, we introduce state-of-the-art approaches to face recognition.

3.2 Face embedding

As we aim to build a robust system, which can discriminate up to millions different people, so the 128-d or 256-d feature vectors or less might lead to the lack of features to enhance the discrimination of people's faces. Larger dimensional feature vectors would contain more information for distinguishing the faces. In practical, due to the expensive computational cost, we need to trade off between high dimensional and time processing. Therefore, in this study, we choose the 512-d feature vectors from Arcface to fulfill real-time processing while being able to achieve good performance. The reason to select 512-d feature vectors through a pilot experiment (performance and speed comparison between 512-d vector and 1024-d vectors) is demonstrated in experimental evaluation Section 3.

Due to dramatically increasing the number of users, which leads to a significant challenge to face recognition. Face features are then expected to contain information of high-level characteristics that can differentiate between individuals and others. Herein, we choose Insightface, which is an implement of ArcFace [27], for the face recognition task with an anticipate that the system could overcome the problem of large-scale identities by giving a 512 dimensional output vector (512d-vector) instead of 128d as originally proposed FaceNet [24] or Dlib [63]. 512d vectors are intuitively illustrated via analysing the angle statistics. In particular, ArcFace was aimed to make predictions only depend on the angle between features and weights. From that, the ground-true class weight W_j in ArcFace is not just the closest weight to the embedding feature x_i but also the closest weight while adding the angle m. By using this stricter condition of Arcface in training, we can make a strong boundaries between neighbours and has a better performance while testing.

In addition, once working with traditional softmax loss (2), we always encounter a noticeable ambiguity in decision boundaries as the feature embedding is not optimized to enforce higher similarity for intraclass samples and diversity for inter-class samples.

$$-\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{W_{y_i}^T x_i+b_{y_i}}}{\sum_{j=1}^{n}e^{W_j^T x_i+b_j}} \tag{2}$$

In order to overcome this limitation, ArcFace creates a more evident gap between the nearest classes by particularly set $b_j = 0$, transformed $W_j^T x_i = \| W_j \| \| x_i \| \cos\theta_j$ while θ_j is the angle between weight W_j and feature x_i, used l2 norm for $\| W_j \|$ and $\| x_i \|$ then re-scaled $\| x_i \|$ to s. After getting angle between feature x_i and the ground truth weight W_{y_i}, an angular margin penalty m is added on the ground truth angle θ_{y_i}. The following step is then processed by multiplying all logits $(cos(\theta_{y_i} + m)$ by scale s before getting through softmax. The final loss function is defined as:

$$-\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))}+\sum_{j=1,j\neq y_i}^{n}e^{s\cos\theta_j}} \tag{3}$$

Precisely, face is detected from image, cropped and resized to 112x112 pixels before the embedding process. This step brings an effort in decreasing the computational cost and improving network efficiency. Since ArcFace is an angular margin loss and the embedded vector contains geometric features, cosine distance should be used to compute distances between the probe image vs all registered faces. However, to boost up the system speed, we made a slight change in the distance function as further depicted in the nearest neighbors calculation in 3.3. At the end of this stage, calculated distances are then sorted and the top 5 smallest distances are taken as inputs for the adjacent model.

3.3 Improved face recognition

We propose a combination of K-nearest neighbors (k-NN) and Local Binary Patterns (LBP) into one single classifier, a simple yet effective classifier that is robust to marked faces in our face recognition system. By utilizing the robust 512d-vectors for accurate prediction, k-NN is chosen as it's simple while ensuring no data being lost during the learning process. In addition, k-NN model can be updated new knowledge without training all data from scratch. We can also set the specific value to K for prediction and use those K points in further assurance process if needed, which might be more complicated for classifiers such as an artificial neural network. Therefore, despite of the heavy computational cost when the data 's size grows, k-NN is simple for parameter's tuned.

K-nearest neighbors: The face image after being embedded into a 512d-vector can be inputted k-NN for training and give out k closest names in the prediction stage afterwards. At this stage, whenever an input vector queries, each instance of known data is popped out to measure the similarity with the queried one. All distance values are sorted in ascending order and the first k entries should be the expected results. KNN model calculates distance between 2 vectors based on any functions that we define. In this proposed method, instead of directly using cosine formula to get the similarity, we calculate the dot product of the pair of vectors which also contains the angular information as shown in (5). Additionally, our proposed distance function is compatible to most of clustering or nearest neighbors searching algorithms by reasoning the fact that all the classifiers categorizing an input based on the minimum distance between classes. Moreover, to use the cosin function, we need to do classification based on the maximum $\cos\theta$ value. Since the $\cos\theta$ is inversely proportional to the angle θ , which represents the different between two vectors (Fig. 2).

This slight change in the distance function saves computational cost and gains higher evaluation scores in some cases (see Table 6). After calculating the dot product of two vectors, we then subtract the result from 1 to

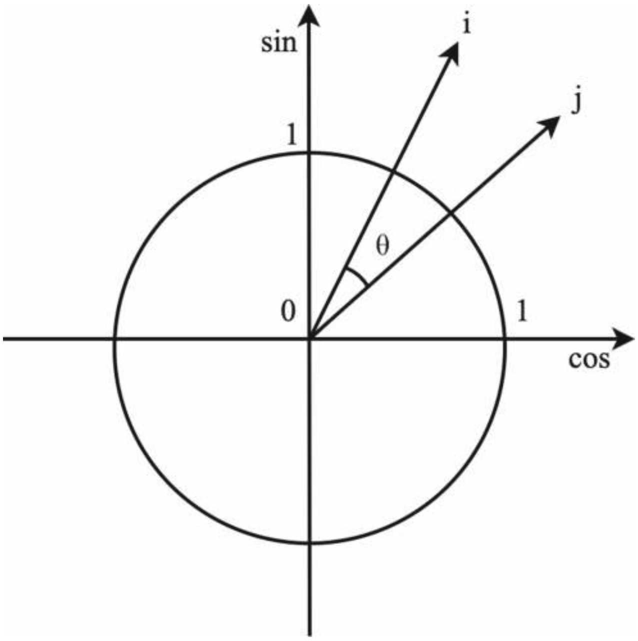


Fig. 2 Angle between two vectors indicating the difference

get the distance value. The proposed function is formally formulated as (4):

$$d(i, j) = 1 - i \bullet j \tag{4}$$

$$\cos\theta = \frac{i \bullet j}{\|i\| \|j\|} \tag{5}$$

in which, i is the queried vector, j is an instance in the known list, $i \bullet j$ is the dot product of 2 vectors, $\|i\|$ and $\|j\|$ are respectively the magnitudes of vector i and j and θ is the angle between 2 vectors. After this step, the top k suspicious names are given out for further identification process.

LBP-based voting: This addition is the key to the success of this work. The reason for choosing LBP can be explained as follows. Let consider Gabor [34], for example, can be efficient for a small number of texture classes, then for three or more, it becomes worse. Besides, according to [74], the original Gabor only works well for macro textures and its application on micro ones is under-performed. Therefore, to cluster or classify an input between hundreds and thousands of individuals's eyebrows, Gabor has been shown ineffective.Other characterization methods such as Eigenface [75] and Fisherface [76], which both have PCA in the process, the dimension reduction may cause some loss of discriminant information useful in the further process [76] and lead to the lower recognition rate in different datasets [77]. Finally, the computational complexity of all 3 mentioned methods has prevented their use in practice. We illustrates the comparison between Gabor and LBP-voting for the assurance process in experimental evaluation section.

The concept in this part is instead of picking the 1st ranked element from top 5 given entries as labeled, we exactly attached a ‘voting layer’ to get the final decision. In particular, we focus on eyebrows area [64, 65] based on the target is to recognize people even wearing mask. Though eyes are considered an identifiable feature [66], they are still excluded from our interested regions since it is literally ‘unstable’ and directly impacts on prediction results. As mentioned, features extracted from eyes are not always the same to an individual in many cases, for example: flinching or sunglasses covered. By defining our specific ROI (region of interest), we manually retrain an optimal shape predictor to localize points in the interested area, using dlib [63]. The results are depicted in Fig. 3.

For each input image, left and right eyebrows are extracted as two image patches from corresponding landmark points' coordinates [67], then we apply median filter [68] for image processing before finding texture patterns. This step is particularly useful for images with noise around edges and lines. After that, histogram equalization is applied on each patch to map the original histogram distribution to a new distribution (wider and more uniform). This will result in high-level contrast images. Moreover, we utilize Local Binary Pattern (LBP) [69, 70], an effective method for texture classification and even face recognition, as a voting model. Based on LBP, intensively robust hand-crafted features might not be a must, but it is expected to be balanced between the effectiveness and lightness with an input of simple representative vectors.

Algorithm 1 Double assurance

```
Input  : Query image $I$
Output: Element with the highest similarity

Calculate embedded vector (I) $\rightarrow V_I$
for $V_k$ in KNN Model do
    distance($V_k$, $V_I$)
end
Sort distances in order
return Top 5 nearest vectors with distances $\rightarrow$
$v_1, v_2, v_3, v_4, v_5$ and $d_1, d_2, d_3, d_4, d_5$
Let $d_1$ be the smallest distance
if $d_1 < D_a$ then
    return corresponding person with distance $d_1$
else if $d_1 > D_n$ then
    return Unknown person
else
    histogram(I) $\rightarrow h_I$
    for $V_k$ in Top 5 nearest vectors do
        mapping($V_k$) $\rightarrow u_1, u_2.u_3, u_4, u_5$
    end
    Initialize count dictionary $\rightarrow$ C =
    $"u_1" : 0, "u_2" : 0, "u_3" : 0, "u_4" : 0, "u_5" : 0$
    repeat $T$ times
        for $u_k$ in list(u1, u2, u3, u4, u5) do
            histogram($u_k$) $\rightarrow h_1, h_2, h_3, h_4, h_5$
        end
        for $h_k$ in list($h_1, h_2, h_3, h_4, h_5$) do
            distance($h_k$, $h_I$)
        end
        Sort distances in order
        Find user with smallest distance $\rightarrow u_k$
        C[$u_k$] += 1
    end
    Do voting in the count dictionary
    return person with the highest vote
```

In addition, an operator is applied on every pixel of image using a concept of sliding window, size defaulted to 3x3 but can be changed depending on different parameter values of radius R and number of neighbours P. As shown in the (6), The value of center pixel is set as a threshold and compared to eight neighbors' pixels. If a neighbor pixel has a higher (or an equal) gray level than the center pixel then 1 is assigned to that neighbor pixel, otherwise it will be 0

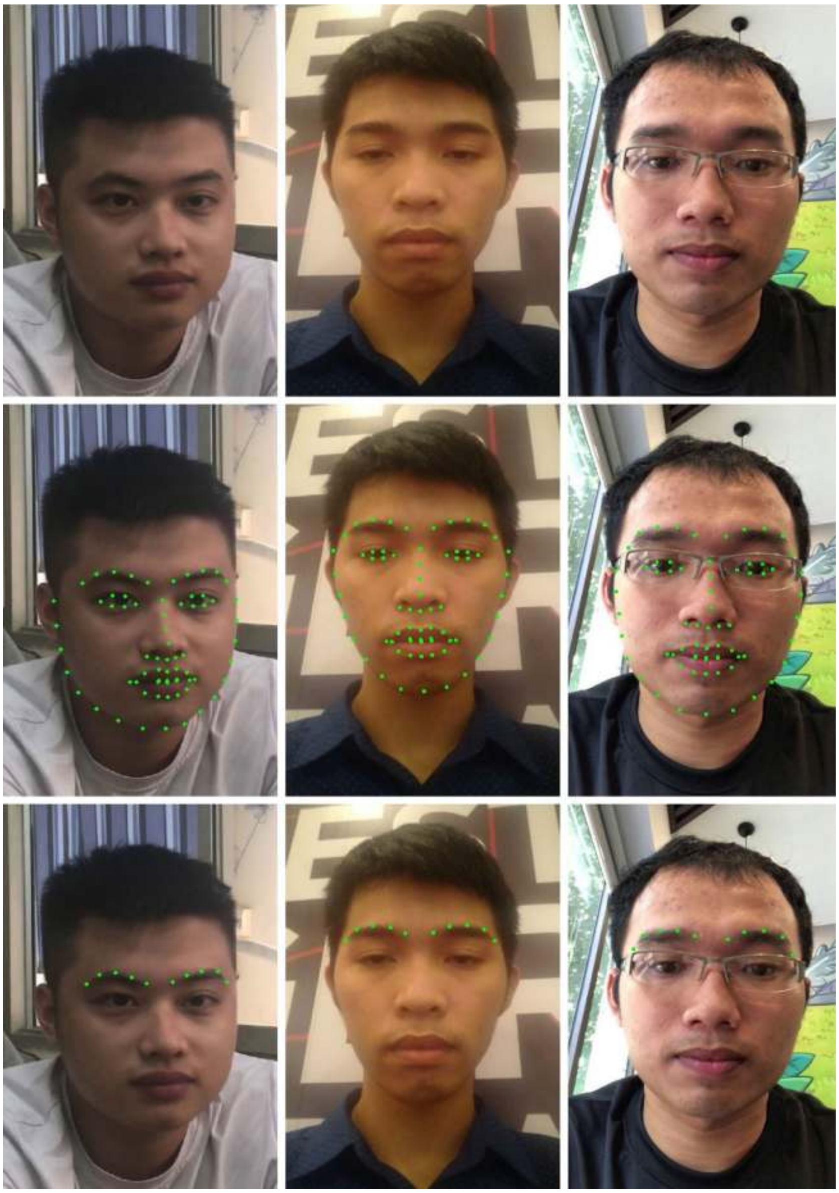
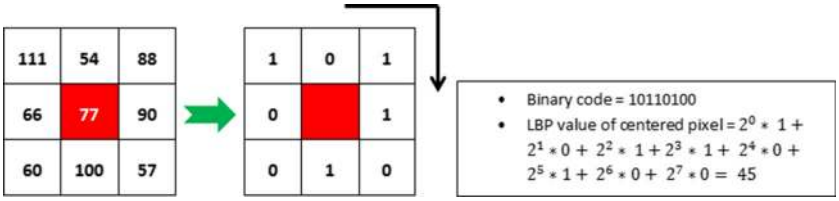


Fig. 3 Localization of landmark points for input images (top) on the whole face (middle) and on specific ROIs (bottom)

(6). The LBP code for the center pixel is then produced by concatenating the eight ones or zeros to a binary code then convert to decimal value (example in Fig. 4). Once the LBP for every pixel is calculated, the feature vector of the image can be constructed. In particular, we can capture fine-grained details in the image by calculating the frequency of occurrence of each code over a region as histogram features.

$$P_i = \begin{cases} 0 & \text{if } g_i < g_c \\ 1 & \text{otherwise} \end{cases} \tag{6}$$

In Fig. 5, the image is divided into K smaller regions that we then apply LBP on each to construct the histograms.



111	54	88
66	77	90
60	100	57

1	0	1
0		1
0	1	0

- Binary code = 10110100
- LBP value of centered pixel = $2^0 * 1 + 2^1 * 0 + 2^2 * 1 + 2^3 * 1 + 2^4 * 0 + 2^5 * 1 + 2^6 * 0 + 2^7 * 0 = 45$

Fig. 4 Binary code and value are made for a pixel

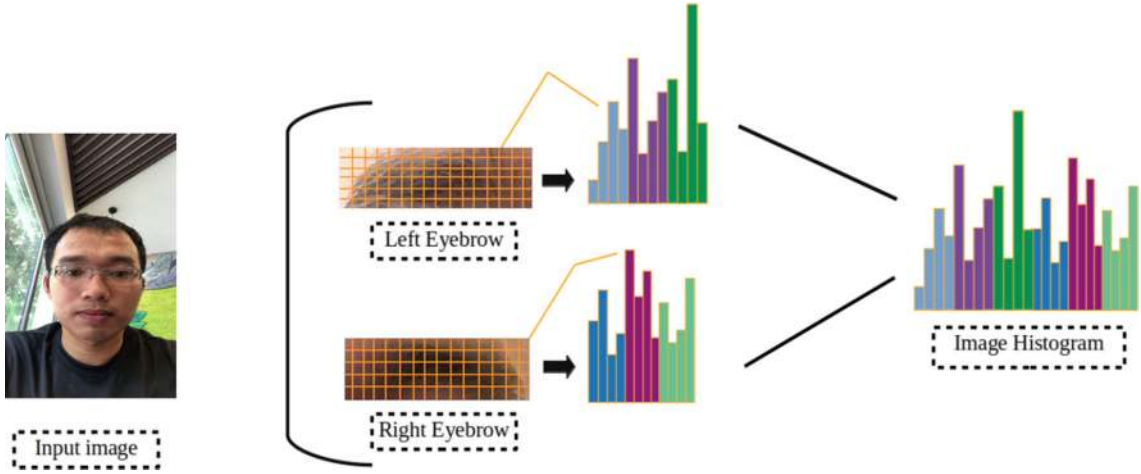


Fig. 5 Feature histogram created by the LBP pipeline

The feature vector of an image is created by concatenating all the calculated regional histograms to the bigger one. For an individual's image, the final vector is concatenated by two histograms from left and right eyebrows images. As increasing the value of P resulted in proportionate increases in the number of dimensionalities of the histogram vector, in our proposed method, R=3 and P=24 are configured to account for the variable neighborhood sizes and to capture the more details of image.

Finally, we build 2 modules to solve the problem: a KNN model to look up the top K closest neighbors and a histogram database for loop mapping to find the final identity of the input.

3.4 Datasets

3.4.1 Essex face recognition data

The dataset was collected by Libor Spacek with the major participation of first year undergraduate students at the University of Essex. It contains 4 subsets: faces94, faces95, faces96 and grimace. In which, the variation of backgrounds and head scales, as well as the variation of expressions is applied on the last 2 folders (faces96, grimace). People in Essex's face recognition data come from various racial origin with a wide range of ages and different appearances (wearing glasses or having beards) [71]. Manual check was done to assure no duplicated or mixed folder, which contains multiple individuals in one place. The detail of the dataset is given in Table 1.

3.4.2 COMASK20 dataset

In addition, we collect a dataset from our institution. Each subject was asked to represent by a 5-10-seconds video, which then frames were then split every 0.5 seconds and stored in separate folders. We manually eliminated all obscured images in folders to make sure the quality of figures were well-observed. In particular, image is considered as unqualified if it contains face under occlusion, for example, hair or dazzling light covered. Besides, blurred facial images also need to be ejected. As regards the previous step, it explicitly led to an unbalance in the number of images between individuals. For instance, some folders only contain 3-5 images. In order to simulate possible cases of real life registration and prediction, videos were made in variation of backgrounds, head scales and light conditions. We requested individuals to go to different places even in the same spot to record, which results in different quality of video as there is a notable change in light intensity between angles of a same spot. More precisely, individual was recorded at home, cafeteria and classes. We implicitly avoid areas, where the light or sunlight shine down directly on the subject. At each place, we took a 3-5s video then finally concatenated all of records as one for each person. The split frames in our dataset also have different sizes as videos were collected in both vertical and horizontal modes. In the end, we obtain 2754 facial images labeled for 300 different identities. Figure 6 represents example images from our dataset.

4 Experimental evaluation

4.1 Experimental settings

All implemented code was executed on Ubuntu 18.04 64-bit, 16GB RAM, Intel®Core^{TM} i7-8700 CPU @ 3.20GHz × 12 and Intel®UHD Graphics 630 (Coffeelake 3x8 GT2). Two datasets were used to evaluate the proposed method: Essex's Face Recognition Data and self-collected data. Due to the lack of images of wearing-mask people, we used

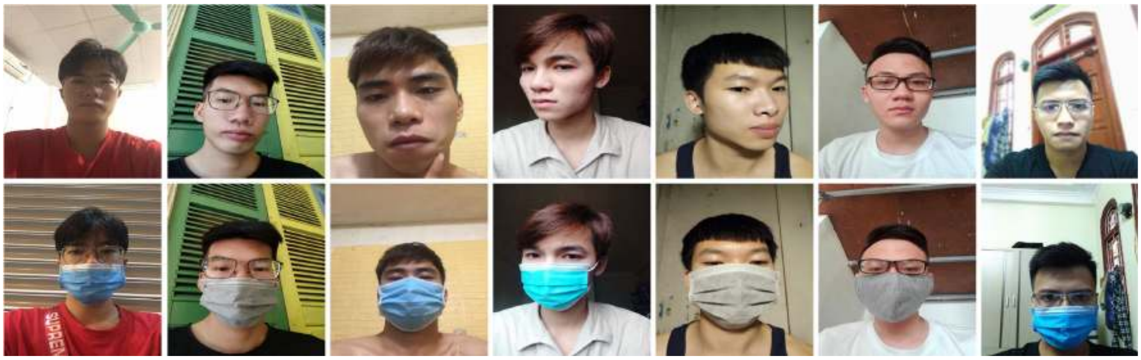


Fig. 6 Original face images (first row) and Masked face images (second row) in COMASK20 dataset

dlib 68-points landmark as an alternative for generating and wrapping an artificial mask on face. After this step, each image on testing dataset should have another version with mask, as shown in Figs. 6 and 7.

In training stage, only non-masked images were used for training the model. We split the data of each individual into 85% for training and 15% for testing. This ratio is reasonable as some subjects might have a small number of images (i.e. less than 5). As illustrated in the flowchart (Fig. 1), a face is detected and then got embedded by Insightface, which are encoded into the 512d-vector. Next, the vector of 512 elements is attached to an identity tag. Both will then be added to two separate lists: one for feature vectors and the other for names. Loop the process till we get the last image in dataset. Finally two lists are fed-up to k-NN, and the name list is also saved for further index mapping. Histogram database is also created by the flow described in *LBP-based voting* in Section 3.3.

In the prediction stage, a 512d-vector is extracted from the input image for the input of the k-NN model to query about 5 suspicious names (K=5). In addition to the original k-NN, herein, we set a threshold D_a to 0.35 as the maximum distance that can be accepted to be considered neighbors. Accordingly, model will return none if no element in the known list holds the condition. Otherwise, the one with the smallest distance in the satisfied set will be returned a long with its label. In this case, another threshold called D_n is defined for identifying the subject. As regards this, we only do the "assuring comparison" process when there is an element, which distance is between D_a and D_n (D_n = 0.7). From the top 5 smallest distances given out by KNN, we trace back to 5 corresponding names by mapping the returned indexes in the saved name list. In the next step, we calculate distances between the histogram vector of the input image $input_h$ vs the histogram vector from each user $user_h$ in the mapped names list. Finally, to finish one assuring process, the element with smallest distance will take one increment on its count variable. After T times loop of assuring (T=100), the voting process finds out the final identity, as described in the Algorithm 1 (Table 1).

In the embedding process, our system has two main steps which are Arcface and LBP respectively. In particular, the whole masked face was embedded firstly by Arcface to find the person with the most similarity of distance. This comparison will return the similarity distance, which we will use afterward to decide whether to continue the second step (LBP) further or stop. If the distance is within the range of D_a and D_n, the result is returned and the algorithm

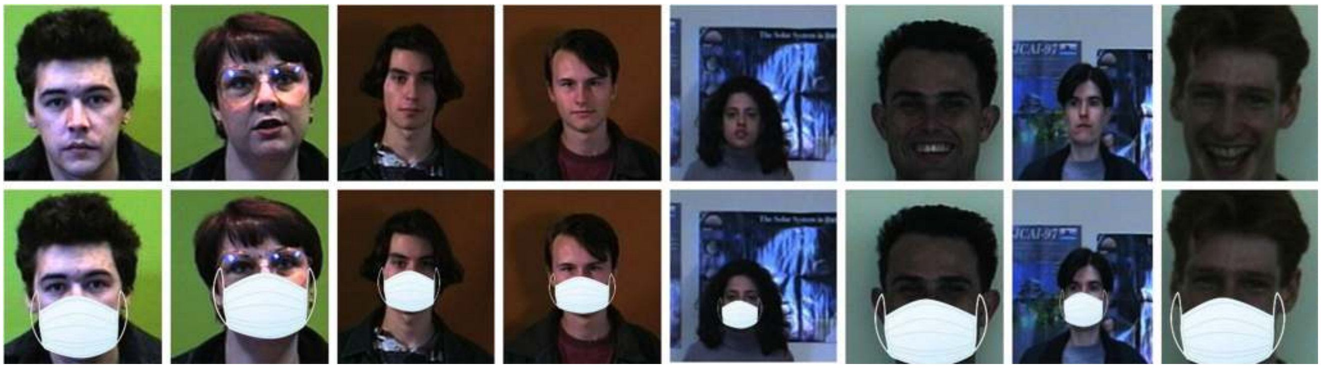


Fig. 7 Original face images (first row) and Masked face images (second row) in Essex dataset

Table 1 Essex's face recognition dataset summary

	Resolution	Initial reported	After manual check
Faces94	180 x 200 pixels	153	152
Faces95	180 x 200 pixels	72	72
Faces96	196 x 196 pixels	152	152
Grimace	180 x 200 pixels	18	18

terminates. Otherwise, the eyebrows area was extracted for calculating the distance if it is not in the range. In this case, we plot the histogram from eyebrows version and then doing the comparison from top nearest 5 subjects from the first step to derive the decision.

4.2 Result analysis

One of the main contribution of this paper is the proposed distance function. As can be seen in Table 2, the slight change from cosine distance can make more effective on both model evaluation and computational time. On thê COMASK 20 dataset, our proposed function has achieved an F1-score of (87%), which quite significant compared to Euclidean and Cosine. Beyond the accurate results, our proposed distance also saves up 8% and 10.7% computational cost compared to Euclidean and Cosine respectively.

To demonstrate the effectiveness of the 512D vector choice as output of face embedding stage, we conducted a pilot study for performance comparison and computational speed comparison between 512-d and 1024-d vectors. To identify an unknown face, the vector extracted from the face must be compared with all the faces vectors in the entire dataset. It is evident that the 1024-d vector would be much slower than the 512-d vector as shown in Table 3. In addition, the 512-d vectors have been shown to be enough to contain all useful discriminative information. Hence, the performance of 512-d and 1024-d vector are almost the same which are 0.93 and 0.94 with our COMASK dataset.

The proposed model is rigorously evaluated on the Essex and COMASK20 datasets. In addition, the model is compared to InsightFace and Dlib, state-of-the art face recognition methods. We use precision, recall, and f1-score for performance metrics.

Table 2 Comparison table between our purposed distance functions vs Euclidean and Cosine on COMASK 20

Distance function	Precision	Recall	F1-score	Inference time
Euclidean	0.85	0.69	0.76	1.11337
Cosine	0.93	0.82	0.87	1.14166
Our proposal	0.87	0.87	0.87	1.01947

Table 3 The face recognition performance for 512-D vs 1024-D feature vectors

	Precision	Recall	F1-core	Inference time
512-D	0.87	0.87	0.87	0.96046
1024-D	0.89	0.87	0.88	1.29443

As can be seen in Tables 4 and 5, our method has outperformed InsightFace and Dlib with the average precision and recall are more than 96% over the 2 datasets. In addition, our work has achieved 99% and 87% in f1-score on the Essex's dataset and on ours respectively, which are highly promising. In the same context, InsightFace only achieved an averaged f1-score of 57% on masked face datasets. This situation can be explained that Insightface has been used a lot of prominent features on the face to categorize, particularly 512-d feature vectors. The vectors are embedded from these features thoroughly depicting individual facial characteristics. However, the wearing mask on the face implicitly obscures the face areas where prominent features located, which significantly leads to be mis-classified for masked faces, therefore produced low results.

Similarly, the Dlib model has underperformed with the low scores over the Essex and COMASK20 datasets. Especially on the COMASK20 dataset, Dlib particularly gained a modest number of f1-score when running by original Dlib (12%) and 21% by being integrated with our module, as depicted in Table 5. Since Dlib uses face-embeddings as 128d vectors, which might be omitted some facial features. As result, Dlib struggles with the similarity search results and ends up with inaccurate top K nearest names before fed in LBP. In order to improve the accuracies, we investigate the distance function the number of neighbors (K). Particularly, euclidean, cosine and our proposed distance are used to calculate the similarity between faces to an input, whereas 5 different values of K: 1, 3, 5, 6, 10 are respectively set as the number of nearest neighbors in the comparison. In order to compare the effectiveness of LBP and Gabor, we replaced the LBP with Gabor filter and the results are shown in Table 5 which shows that the combination of Insightface and Gabor filter (1 kernel and 80 kernels) have underperformed our final proposed with F1-score are 0.84 and 0.85 respectively. In addition, the Gabor filter has greater complexity than LPB that proven by the prediction time. Our final proposal is able to run at approximately one second, while the Gabor based methods run at longer time which are 1.3 second and 3.2 seconds for 1 and 80 kernels respectively.

As illustrated in Table 6, varying the K-value impacts on the performance of our proposed model on the COMASK20 dataset. This can be explainded, as no constraints imposed

Table 4 Comparison of our proposed method vs. other methods on the Essex dataset (K=5)

		Precision	Recall	F1-score
faces94	Dlib	0.52	0.31	0.39
	Dlib + LBP-based voting	0.78	0.56	0.65
	InsightFace	0.72	0.59	0.65
	Insightface (euclidean) + LBP	0.99	0.99	0.99
	Insightface (cosine) + LBP	0.99	0.99	0.99
	Our proposed method	0.99	0.99	0.99
faces95	Dlib	0.67	0.39	0.49
	Dlib + LBP-based voting	0.86	0.57	0.69
	InsightFace	0.76	0.55	0.64
	Insightface (euclidean) + LBP	0.99	0.98	0.98
	Insightface (cosine) + LBP	0.99	0.99	0.99
	Our proposed method	0.99	0.99	0.99
faces96	Dlib	0.59	0.27	0.37
	Dlib + LBP-based voting	0.82	0.41	0.55
	InsightFace	0.64	0.34	0.44
	Insightface (euclidean) + LBP	0.99	0.88	0.93
	Insightface (cosine) + LBP	0.99	0.97	0.98
	Our proposed method	0.99	0.97	0.98
grimace	Dlib	0.79	0.49	0.6
	Dlib + LBP-based voting	0.79	0.51	0.62
	InsightFace	0.79	0.47	0.59
	Insightface (euclidean) + LBP	0.95	0.94	0.94
	Insightface (cosine) + LBP	1	1	1
	Our proposed method	1	1	1

on the subject for capturing their faces, our dataset has significant variations of background, head size and light intensities meanwhile there are the same environmental conditions in each subset of Essex's dataset. This also could be seen in Table 7 that average scores of models on 4 sub-folders of the Essex dataset are nearly the same when running with any specific K values. Precisely, the model running with cosine distance and the one running with our final proposed function produced exactly the same results on the whole of data, especially on Essex's face recognition dataset. In particular, for our model, with K=3, the model achieves 89% recall and 93% f1-score running for the euclidean distance. In contrast, LBP-voting can achieves up to 94% in both recall and f1-score, which also higher than cosine distance, 92% recall and 94% f1-core. Besides, we also experienced a fluctuation of scores when running with euclidean distance and different number of neighbors.

As detailed in Table 6, K=1 is a best value for our proposed method. According to the experimental results in [72, 73], with different chosen distance function, it can significantly impact on the performance of KNN classifier. Therefore, it can be understandable that the change of

Table 5 Comparison of our proposed method vs other methods on COMASK20 (K=5)

	Precision	Recall	F1-score	Prediction time
Dlib	0.12	0.13	0.12	1.54875
Dlib + LBP-based voting	0.22	0.20	0.21	1.57038
InsightFace	0.58	0.46	0.51	0.99812
Insightface (euclidean) + LBP	0.85	0.69	0.76	1.11337
Insightface (cosine) + LBP	0.93	0.82	0.87	1.14166
Insightface + Gabor (1 kernel)	0.87	0.82	0.84	1.34166
Insightface + Gabor (80 kernels)	0.88	0.82	0.85	3.24166
Our proposed method	0.87	0.87	0.87	1.01947

Table 6 The proposed model's performance for different parameter values of K and distance functions on COMASK20

		Euclidean			Cosine			LBP-voting		
		Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
	faces94	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
	faces95	0.99	0.98	0.98	0.99	0.99	0.99	0.99	0.99	0.99
	faces96	0.99	0.89	0.94	0.99	0.97	0.98	0.99	0.97	0.98
	grimace	0.95	0.94	0.94	1	1	1	1	1	1
k=1	COMASK20	0.97	0.91	0.94	0.97	0.94	0.95	0.96	0.95	0.95
	faces94	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
	faces95	0.99	0.98	0.98	0.99	0.99	0.99	0.99	0.99	0.99
	faces96	0.99	0.89	0.94	0.99	0.97	0.98	0.99	0.97	0.98
	grimace	0.95	0.94	0.94	1	1	1	1	1	1
k=3	COMASK20	0.97	0.89	0.93	0.97	0.92	0.94	0.95	0.94	0.94
	faces94	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
	faces95	0.99	0.98	0.98	0.99	0.99	0.99	0.99	0.99	0.99
	faces96	0.99	0.88	0.93	0.99	0.97	0.98	0.99	0.97	0.98
	grimace	0.95	0.94	0.94	1	1	1	1	1	1
k=5	COMASK20	0.85	0.69	0.76	0.93	0.82	0.87	0.87	0.87	0.87
	faces94	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
	faces95	0.99	0.98	0.98	0.99	0.99	0.99	0.99	0.99	0.99
	faces96	0.99	0.88	0.93	0.99	0.97	0.98	0.99	0.97	0.98
	grimace	0.95	0.94	0.94	1	1	1	1	1	1
k=6	COMASK20	0.83	0.66	0.74	0.92	0.80	0.86	0.86	0.86	0.86
	faces94	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
	faces95	0.99	0.98	0.98	0.99	0.99	0.99	0.99	0.99	0.99
	faces96	0.99	0.89	0.94	0.99	0.97	0.98	0.99	0.97	0.98
	grimace	0.95	0.94	0.94	1	1	1	1	1	1
k=10	COMASK20	0.75	0.54	0.63	0.88	0.74	0.80	0.81	0.81	0.81

the distance function might save computational cost and improve the performance for masked face datasets. Finally, comparing to the original cosine distance, KNN could be theoretically fitting the data and converge model faster by using our customized function.

In Fig. 8, we also plot ROC curve for COMASK20 dataset, which is the alternative way to compare our method with state-of-the-art methods. ROC curves are formed by plotting the true positive rate and the false positive rate at various threshold settings. In order to analyze the curve, the areas under the curve (AUC) are taken into consideration. The higher the area is, the better the method is. As can be seen from the ROC curves, our proposed method achieves an AUC of 0.9 larger than other AUCs by a large margin.

Table 7 Results of K values and distance methods on the Essex dataset

		Distance function								
		Euclidean			Cosine			LBP voting		
		Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
Neighbors	k=1	0.98	0.92	0.95	0.99	0.98	0.98	0.99	0.98	0.98
	k=3	0.98	0.92	0.95	0.99	0.97	0.98	0.99	0.97	0.98
	k=5	0.98	0.92	0.95	0.99	0.97	0.98	0.99	0.97	0.98
	k=6	0.96	0.91	0.93	0.99	0.97	0.98	0.99	0.97	0.98
	k=10	0.96	0.9	0.93	0.99	0.97	0.98	0.99	0.97	0.98

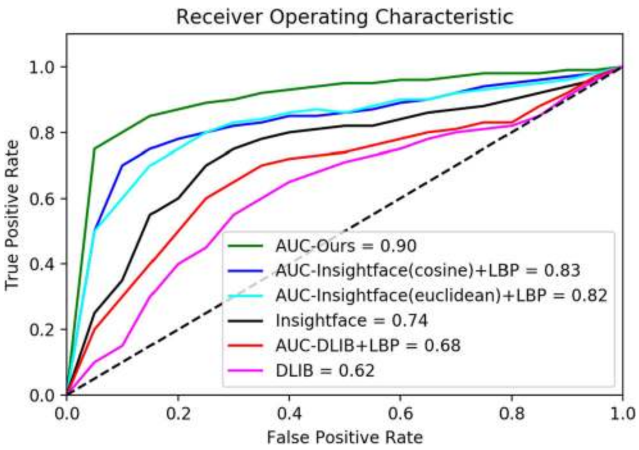


Fig. 8 The ROC curves of ours vs other methods

The DLIB method only obtains an AUC of 0.62. The competitive AUC score proves the discriminative ability of our proposed method.

5 Conclusion

We have presented a method that combinates the deep learning models and Local Binary Pattern (LBP) features into a unified framework to recognize the masked face. The proposed method utilizes deep models for face detection; and extract face features combined with LBP features extracted from eyes and eyebrows. An empirical experiment is conducted for verifying the proposed method. Evaluation results have demonstrated that the proposed method is significantly outperformed several state of the art face recognition methods including Dlib and InsightFace on the published Essex dataset and our self-collected dataset COMASK20, with the results of 87% f1-score on the COMASK20 dataset and 98% f1-score on the Essex dataset. This has indicated the effectiveness and suitability of the proposed method. For future works, we have planned to optimize the proposed model including the reduction of parameters and energy consumption for effectively employing the model on the portable and mobile devices, which can be deployed for checking in the classroom at educational institutions.

Acknowledgements Hoai Nam Vu was funded by Vingroup Joint Stock Company and supported by the Domestic Master/ PhD Scholarship Programme of Vingroup Innovation Foundation (VINIF), Vingroup Big Data Institute (VINBIGDATA), code VNIF.2020.TS.103. In addition, we would like to say many thanks to the volunteers who have participated in this study by providing their facial and masked facial images as well as being agreed to publish the COMASK20 dataset for research purposes.

References

1. https://www.worldometers.info/coronavirus
2. Ranjan R, Sankaranarayanan S, Bansal A, Bodla N, Chen J-C, Patel VM, Castillo CD, Chellappa R (2018) Deep learning for understanding faces: Machines may be just as good, or better, than humans. IEEE Signal Processing Magazine 35(1):66–83
3. Viola P, Jones MJ (2004) Robust real-time face detection. Int J Comput Vis 57(2):137–154
4. Chen W, Wang K, Jiang H, Li M (2016) Skin color modeling for face detection and segmentation: a review and a new approach. Multimedia Tools and Applications 75:839–862
5. Tayal Y, Lamba R, Padhee S (2012) Automatic face detection using color based segmentation. Int J Sci Res Public 2(6):1–7
6. Erdem CE, Ulukaya S, Karaali A, Tanju Erdem A (2011) Combining Haar feature and skin color based classifiers for face detection. In: 2011 IEEE International conference on acoustics, speech and signal processing (ICASSP), IEEE, pp 1497–1500
7. Li Y, Sun B, Wu T, Wang Y (2016) Face detection with end-to-end integration of a convnet and a 3d model. In: European conference on computer vision. Springer, Cham, pp 420–436
8. Hu P, Ramanan D (2017) Finding tiny faces. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 951–959
9. Jiang H, Learned-Miller E (2017) Face detection with the faster r-CNN. In: 2017 12Th IEEE international conference on automatic face and gesture recognition (FG 2017), IEEE, pp 650–657
10. Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: Towards real-time object detection with region proposal networks. arXiv:1506.01497
11. Qi R, Jia R-S, Mao Q-C, Sun H-M, Zuo L-Q (2019) Face detection method based on cascaded convolutional networks. IEEE Access 7:110740–110748
12. Xiong X, De la Torre F (2013) Supervised descent method and its applications to face alignment. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 532–539
13. Zhang J, Shan S, Kan M, Chen X (2014) Coarse-to-fine auto-encoder networks (cfan) for real-time face alignment. In: European conference on computer vision. Springer, Cham, pp 1–16
14. Ren S, Cao X, Wei Y, Sun J (2014) Face alignment at 3000 fps via regressing local binary features. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 1685–1692
15. Rathod D, Vinay A, Shylaja S, Natarajan S (2014) Facial landmark localization-a literature survey. Int J Current Eng Technol 4(3):1901–1907
16. Yang S, Xiong Y, Loy CC, Tang X (2017) Face detection through scale-friendly deep convolutional networks. arXiv:1706.02863
17. Wu Y, Hassner T, Kim KG, Medioni G, Natarajan P (2017) Facial landmark detection with tweaked convolutional neural networks. IEEE Trans Pattern Anal Mach Intell 40(12):3067–3074
18. Bodini M (2019) A review of facial landmark extraction in 2d images and videos using deep learning. Big Data and Cognitive Computing 3(1):14
19. Ding Y, Zhao Q, Li B, Yuan X (2017) Facial expression recognition from image sequence based on LBP and Taylor expansion. IEEE Access 5:19409–19419
20. Benedict SR, Satheesh Kumar J (2016) Geometric shaped facial feature extraction for face recognition. In: 2016 IEEE International conference on advances in computer applications (ICACA), IEEE, pp 275–278

21. Gumus E, Kilic N, Sertbas A, Ucan ON (2010) Evaluation of face recognition techniques using PCA, wavelets and SVM. Expert Systems with Applications 37(9):6404–6408
22. Xing EP, Ng AY, Jordan MI, Russell S (2002) Distance metric learning with application to clustering with side-information. NIPS 15(505–512):12
23. Weinberger KQ, Saul LK (2009) Distance metric learning for large margin nearest neighbor classification. J Mach Learn Res 10(2):207–244
24. Schroff F, Kalenichenko D, Philbin J (2015) Facenet: a unified embedding for face recognition and clustering. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 815–823
25. Liu W, Wen Y, Yu Z, Li M, Raj B, Le S (2017) Sphereface: Deep hypersphere embedding for face recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 212–220
26. Liu W, Zhang Y-M, Li X, Yu Z, Dai B, Zhao T, Song L (2017) Deep hyperspherical learning. arXiv:1711.03189
27. Deng J, Guo J, Xue N, Zafeiriou S (2019) Arcface: Additive angular margin loss for deep face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 4690–4699
28. Zeng D, Veldhuis R, Spreeuwers L (2020) A survey of face recognition techniques under occlusion. arXiv:2006.11366
29. Liao S, Zhu X, Lei Z, Zhang L, Li SZ (2007) Learning multi-scale block local binary patterns for face recognition. In: International conference on biometrics. Springer, Berlin, pp 828–837
30. Zhao G, Pietikainen M (2007) Dynamic texture recognition using local binary patterns with an application to facial expressions. IEEE Trans Pattern Anal Mach Intell 29(6):915–928
31. Lowe DG (1999) Object recognition from local scale-invariant features. In: Proceedings of the seventh IEEE international conference on computer vision, vol 2, Ieee, pp 1150–1157
32. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: 2005 IEEE Computer society conference on computer vision and pattern recognition (CVPR'05), vol 1, Ieee, pp 886–893
33. Zhang B, Shan S, Chen X, Gao W (2006) Histogram of gabor phase patterns (hgpp): A novel object representation approach for face recognition. IEEE Trans Image process 16(1):57–68
34. Zou J, Ji Q, Nagy G (2007) A comparative study of local matching approach for face recognition. IEEE Trans Image Process 16(10):2617–2628
35. Wiskott L, Krüger N, Kuiger N, Von Der Malsburg C (1997) Face recognition by elastic bunch graph matching. IEEE Transactions on pattern analysis and machine intelligence 19(7):775–779
36. Cheheb I, Al-Maadeed N, Al-Madeed S, Bouridane A, Jiang R (2017) Random sampling for patch-based face recognition. In: 2017 5Th international workshop on biometrics and forensics (IWBF), IEEE, pp 1–5
37. Seo J, Park H (2011) A robust face recognition through statistical learning of local features. In: International conference on neural information processing. Springer, Berlin, pp 335–341
38. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
39. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems 25:1097–1105
40. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556
41. Min R, Hadid A, Dugelay Jean-Luc (2011) Improving the recognition of faces occluded by facial accessories. In: Face and gesture 2011, IEEE, pp 442–447
42. Chen Z, Xu T, Han Z (2011) Occluded face recognition based on the improved SVM and block weighted LBP. In: 2011 International conference on image analysis and signal processing, IEEE, pp 118–122
43. Andrés AM, Padovani S, Tepper M, Jacobo-Berlles J (2014) Face recognition on partially occluded images using compressed sensing. Pattern Recognition Letters 36:235–242
44. Xie J, Xu L, Chen E (2012) Image denoising and inpainting with deep neural networks. Advances in Neural Information Processing Systems 25:341–349
45. Ou W, Luan X, Gou J, Zhou Q, Xiao W, Xiong X, Zeng W (2018) Robust discriminative nonnegative dictionary learning for occluded face recognition. Pattern Recognition Letters 107:41–49
46. Wan W, Chen J (2017) Occlusion robust face recognition based on mask learning. In: 2017 IEEE International conference on image processing (ICIP), IEEE, pp 3795–3799
47. Song L, Gong D, Li Z, Liu C, Liu W (2019) Occlusion robust face recognition based on mask learning with pairwise differential siamese network. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 773-782
48. Park J-S, Oh YH, Ahn SC, Lee S-W (2005) Glasses removal from facial image using recursive error compensation. IEEE Trans Pattern Anal Mach Intell 27(5):805–811
49. De La Torre F, Black MJ (2003) A framework for robust subspace learning. Int J Comput Vis 54(1):117–142
50. Wright J, Yang AY, Ganesh A, Shankar Sastry S, Ma Y (2008) Robust face recognition via sparse representation. IEEE Trans Pattern Anal Mach Intell 31(2):210–227
51. Zhou Z, Wagner A, Mobahi H, Wright J, Yi Ma (2009) Face recognition with contiguous occlusion using markov random fields. In: 2009 IEEE 12Th international conference on computer vision, IEEE, pp 1050–1057
52. Iliadis M, Wang H, Molina R, Katsaggelos AK (2017) Robust and low-rank representation for fast face identification with occlusions. IEEE Trans Image Process 26(5):2203–2218
53. Zhao F, Feng J, Zhao J, Yang W, Yan S (2017) Robust LSTM-autoencoders for face de-occlusion in the wild. IEEE Trans Image Process 27(2):778–790
54. Oord V, Aaron NK, Kavukcuoglu K (2016) Pixel recurrent neural networks. In: International conference on machine learning, PMLR, pp 1747–1756
55. Kingma DP, Welling M (2013) Auto-encoding variational bayes. arXiv:1312.6114
56. Ulyanov D, Vedaldi A, Lempitsky V (2018) Deep image prior. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 9446–9454
57. Chen Y-A, Chen W-C, Wei C-P, Wang Y-CF (2017) Occlusion-aware face inpainting via generative adversarial networks. In: 2017 IEEE International conference on image processing (ICIP), IEEE, pp 1202–1206
58. Deng J, Guo J, Zhou Y, Yu J, Kotsia I, Zafeiriou S (2019) Retinaface: Single-stage dense face localisation in the wild. arXiv:1905.00641
59. Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H (2017) Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv:1704.04861
60. Lin T-Y, Dollár P, Girshick R, He K, Hariharan B, Belongie S (2017) Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2117–2125
61. Zhao Xu, Liang X, Zhao C, Tang M, Wang J (2019) Real-time multi-scale face detector on embedded devices. Sensors 19(9):2158

62. Zhang K, Zhang Z, Li Z, Qiao Y (2016) Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Process Lett 23(10):1499–1503
63. King DE (2009) Dlib-ml: A machine learning toolkit. J Mach Learn Res 10:1755–1758
64. Sadr J, Jarudi I, Sinha P (2003) The role of eyebrows in face recognition. Perception 32(3):285–293
65. Turkoglu MO, Arican T (2017) Texture-based eyebrow recognition. In: 2017 International conference of the biometrics special interest group (BIOSIG), IEEE, pp 1–6
66. Radji N, Cherifi D, Azrar A (2015) Importance of eyes and eyebrows for face recognition system. In: 2015 3Rd international conference on control, engineering and information technology (CEIT), IEEE, pp 1–6
67. Bah SM, Ming F (2020) An improved face recognition algorithm and its application in attendance management system. Array 5:100014
68. Huang T, Yang GJTGY, Tang G (1979) A fast two-dimensional median filtering algorithm. IEEE Trans Acoustics Speech Signal Process 27(1):13–18
69. Ojala T, Pietikainen M, Harwood D (1994) Performance evaluation of texture measures with classification based on Kullback discrimination of distributions. In: Proceedings of 12th international conference on pattern recognition, vol 1, IEEE, pp 582–585
70. Ahonen T, Hadid A, Pietikainen M (2006) Face description with local binary patterns: Application to face recognition. IEEE Trans Pattern Anal Mach Intell 28(12):2037–2041
71. Spacek L (2008) Description of the collection of facial images. Online] http://cswww.essex.ac.uk/mv/allfaces/index.html
72. Alfeilat A, Arafat H, Hassanat ABA, Lasassmeh O, Tarawneh AS, Alhasanat MB, Salman HSE, Surya Prasath VB (2019) Effects of distance measure choice on k-nearest neighbor classifier performance: a review. Big Data 7(4):221–248
73. Hu L-Y, Huang M-W, Ke S-W, Tsai C-F (2016) The distance function effect on k-nearest neighbor classification for medical datasets. SpringerPlus 5(1):1–9
74. Li M, Staunton RC (2008) Optimum Gabor filter design and local binary patterns for texture segmentation. Pattern Recognition Letters 29(5):664–672
75. Turk MA, Pentland AP (1991) Face recognition using eigenfaces. In: Proceedings. 1991 IEEE computer society conference on computer vision and pattern recognition, IEEE Computer Society, pp 586–587
76. Belhumeur PN, Hespanha JP, Kriegman DJ (1997) Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. IEEE Trans Pattern Anal Mach Intell 19(7):711–720
77. Sultana M, Gavrilova M, Yanushkevich S (2014) Multi-resolution fusion of DTCWT and DCT for shift invariant face recognition. In: 2014 IEEE International conference on systems, man, and cybernetics (SMC), IEEE, pp 80–85

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Hoai Nam Vu was born in Ha Noi, Viet Nam in 1990. He received the B.E. degrees in Electronic and Telecommunication engineering from the Ha Noi University of Science and Technology, Ha Noi, Viet Nam in 2013 and the M.S. degree in Electronic and Computer engineering from Chonnam National University, Gwangju, South Korea, in 2015. He is currently pursuing the Ph.D. degree in Computer Science at Posts and Telecommunications Institute of Technology, Ha Noi. Since 2016, he has been a lecturer with Computer Science Department, Posts and Telecommunications Institute of Technology, Viet Nam. His research interests include drone-based image processing, machine learning, and deep learning.

Mai Huong Nguyen was born in Ha Noi, Vietnam in 1995. She is currently in her second year of Master's program in Computer Science and Engineering at University of Oulu, Oulu, Finland. Before moving to Finland, she worked as a Python developer at Aimesoft JSC, a leading Multimodel AI company in Hanoi, Vietnam. She got a B.S in Computer Science at Posts and Telecommunications Institute of Technology (PTIT), Ha Noi, Vietnam in 2018. Her main research interests are robot vision, cloud/edge computing and machine learning in general.

Cuong Pham is an Associate Professor of Computer Science at Posts and Telecommunications Institute of Technology (PTIT). Prior to joining PTIT, he was a Marie Curie Research Fellow at Philips Research, Eindhoven, the Netherlands. He got a B.S in Computer Science at Vietnam National University in 1998, an MS in Computer Science at New Mexico State University, USA in 2005, and a PhD in Computer Science at Newcastle University, United Kingdoms in 2012. His main research interests are ubiquitous computing, wearable computing, human activity recognition, and machine learning/deep learning.