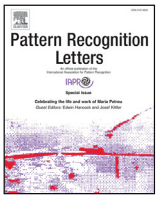
Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec



Low-resolution face recognition in resource-constrained environments☆

Mozhdeh Rouhsedaghat[a,*], Yifan Wang[a], Shuowen Hu[b], Suya You[b], C.-C. Jay Kuo[a]

[a] University of Southern California, Los Angeles, California, USA
[b] DEVCOM Army Research Laboratory, Adelphi, Maryland, USA

ARTICLE INFO

Article history:
Received 25 November 2020
Revised 19 April 2021
Accepted 12 May 2021
Available online 17 June 2021

Edited by: Jiwen Lu

Keywords:
Low-resolution face recognition
Small model
Small data
SSL
Pixelhop++

ABSTRACT

Although Deep Neural Networks (DNNs) have achieved tremendous success in the face recognition task, utilizing them in resource-constrained environments with limited networking and computing is challenging. Such environments often demand a small model capable of being effectively trained on a small number of labeled training data, with low training complexity, and low-resolution input images. To address these challenges, we adopt an emerging machine learning methodology called Successive Subspace Learning (SSL) to propose LRFRHop, a high-performance data-efficient low-resolution face recognition model for resource-constrained environments. SSL offers an explainable non-parametric feature extraction submodel that flexibly trades the model size for the verification performance. Its training complexity is significantly lower than DNN-based models since it is trained in a one-pass feedforward manner without backpropagation. Furthermore, active learning can be conveniently incorporated to reduce the labeling cost. We demonstrate the effectiveness of LRFRHop by conducting experiments on the LFW and the CMU Multi-PIE datasets.

1. Introduction

Deep learning based face recognition has reached maturity in recent years. DNN models consisting of millions of model parameters have been developed and made significant progress. However, their promising performance mainly relies on several factors: higher input image resolutions, using an extremely large number of training images, and abundant computational/memory resources. For example, DeepFace [32] was trained on a collection of photos from Facebook that contains 4.4M images. FaceNet [28] was trained on the Google dataset that contains 500M images. SphereFace [23] was trained on the CASIA-WebFace dataset [37] that contains 0.49M images. Along this direction, many activities are centered on face image collection and the setup of the required computing and communication environment.

We may face an opposite situation in some real-world applications, i.e., edge or mobile computing in resource-constrained environments with poor computing and communication infrastructure, as is often the case in the field and in operational settings. Such environments demand a smaller model size, fewer labeled images for training, lower training and inference complexity, and lower input image resolution, partly due to the need to identify individuals at farther standoff distances. Due to these stringent requirements, DNNs may not be suitable. The goal of this work is to address these challenges by developing a transparent non-parametric model that allows graceful performance tradeoff between resources and performance and is capable of being easily integrated with active learning to minimize the training sample number while achieving relatively high accuracy. We adopt an emerging machine learning system called PixelHop++ [5] to achieve these objectives. PixelHop++ is designed based on the SSL principle and has several unique characteristics that fit our objectives well.

1. PixelHop++ is a lightweight non-parametric model whose size can be flexibly adjusted for graceful performance tradeoff. It is trained in a feedforward one-pass manner and the training complexity is significantly lower than DNNs.
2. SSL adopts a statistics-centric principle. It is a mathematically transparent approach which exploits pixel-to-pixel correlations for dimension reduction to derive image features. It also analyzes statistics between features and labels to identify discriminant features.
3. We will incorporate active learning in SSL to select the most “informative” samples of a dataset for labeling and reduce the

☆ Handle by Associate Editor Li Liu.
* Corresponding author.
E-mail addresses: rouhseda@usc.edu (M. Rouhsedaghat), wang608@usc.edu (Y. Wang), shuowen.hu.civ@mail.mil (S. Hu), suya.you.civ@mail.mil (S. You), jckuo@usc.edu (C.-C.J. Kuo).

https://doi.org/10.1016/j.patrec.2021.05.009
0167-8655/© 2021 Elsevier B.V. All rights reserved.

labeling cost. PixelHop++ is a lightweight model, and it can easily be integrated with active learning.

The main contribution of our work lies in the assembly of two effective tools to address the challenge of face recognition in resource-constrained environments. Both PixelHop++ and active learning are existing tools. Yet, to the best of our knowledge, this is the first time that they are jointly applied to a face biometric problem. We will demonstrate the power of the integrated solution in the context of face recognition with extensive experiments. As the second contribution, we propose a pairwise feature generation module to extract effective joint features from the PixelHop++ output channels for each pair of face images.

The rest of the paper is organized as follows. Related prior work is reviewed in Section 2. The proposed face recognition method and its integration with active learning are presented in Section 3 and Section 4, respectively. Experimental results are provided in Section 5. Finally, concluding remarks and future work are given in Section 6.

2. Related work

Face Recognition. Face recognition has made significant progress in recent years. Most successful face recognition models use deep learning technique [16,23,28,32] and offer high accuracy on the benchmarking datasets. Although these models are powerful for high-resolution face recognition, they usually contain tens of millions of model parameters and require a large amount of training data and computation resources. Recently several lightweight CNN models are proposed [4,22,34] which are significantly smaller than regular CNN networks, but they still have a very large number of model parameters.

Low-Resolution Face Recognition. In comparison with high-resolution face recognition, less attention has been drawn to low-resolution face recognition. Generally, there are two different settings for this problem: high-resolution to low-resolution, and low-resolution to low-resolution. In the first setting, the low-resolution probe images are compared against high-resolution gallery images [2,24,25] while in the second setting both probe and gallery images are low-resolution face images [6,10–12]. It may also be possible for a proposed model to be evaluated under both settings [17]. We evaluate the effectiveness of LRFRHop under the second setting.

Successive Subspace Learning (SSL). The main technique in subspace learning is to project a high-dimension input space to a low-dimensional output subspace, which serves as an approximation to the original one. When these dimension reduction operations are performed sequentially, it leads to successive subspace learning (SSL). PixelHop++ [5] is the latest SSL-based model proposed for unsupervised representation learning on images by applying the channel-wise (c/w) Saab transform to them. The Saab transform [20] is a multi-stage variant of Principal Component Analysis (PCA) conducted on images. In each stage, it applies PCA to pixel blocks and also uses a bias term to avoid the sign-confusion problem [19]. The performance of the Saab transform can be further enhanced by removing the spatial correlation between Saab transform outputs in the current stage so that the Saab transform can be applied to each output channel separately in the next stage. PixelHop++ with the c/w Saab transform is demonstrated to offer an effective multi-stage representation. For example, FaceHop [27] is a recently proposed model for gender classification which leverages PixelHop++ for feature learning on gray-scale face images. In this paper, in addition to incorporating chrominance channels of face images, we add a new module for effective pair-wise feature generation to tackle the face recognition problem.

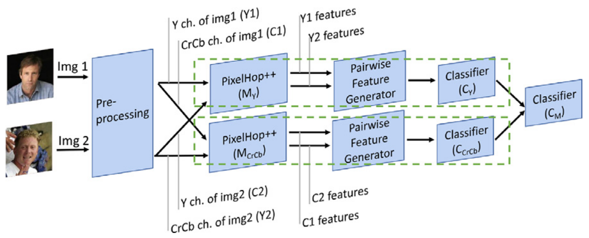


Fig. 1. The block diagram of the proposed face recognition model.

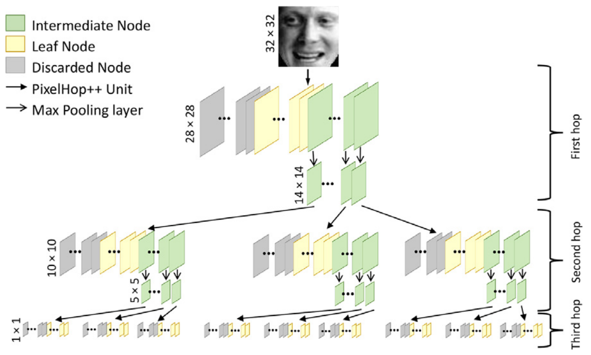


Fig. 2. Illustration of data flow in the three-level c/w Saab transform in PixelHop++, which provides a sequence of successive subspace approximations (SSAs) to the input image.

Active Learning. Active learning is used to select the most informative unlabeled data for labeling so that almost the same accuracy can be reached with the smallest amount of labeled data. An active learning method begins with a small amount of labeled data to train a machine learning model. The model queries the labels of some unlabeled data based on a query strategy, and the model is retrained by all labeled data. This process is repeated until we reach the labeled sample budget. Common query strategies include the entropy method [31], the query by committee method [30] and the core-set method [29]. Active learning was incorporated in DNNs by Wang et al. [33]. An active annotation and learning framework is introduced in [36] for the face recognition task.

3. Proposed face recognition method

The block diagram of LRFRHop is depicted in Fig. 1. As shown in the figure, the proposed model is an ensemble of two submodels, each of which consists of three components: PixelHop++, pairwise feature generation, and a classifier. The Y channels of each face pair are fed into one designated submodel while the CrCb channels are fed into another one. Each submodel generates a probability score and finally a meta classifier ensembles their predictions. We will elaborate on each component of the block diagram below.

3.1. PixelHop++

In each submodel, we use a three-level PixelHop++ system, similar to the system shown in Fig. 2. The input to each of the PixelHop++ systems is a whole face image of size $32{\times}32{\times}K_0$, where $K_0 = 1$ and 2 for the Y channel and CrCb channels, respectively. Each level of a PixelHop++ system has one “PixelHop++ unit” followed by $(2{\times}2)$-to-$(1{\times}1)$ max-pooling layer. In LRFRHop, the PixelHop++ unit of each level operates on blocks of $5{\times}5$pixels with a stride of one. In the training phase of each PixelHop++ unit, we collect sample blocks from each input channel to derive Saab kernels for that channel separately. Then, we project each block on the derived kernels and generate a set of responses for the central pixel in the block. Since the responses can be positive or negative, we add a constant bias to all responses to ensure that they are all non-negative, which explains the name of the "successive approximation with adjusted bias (Saab) transform" [20].

The first Saab kernel is the unit-length constant-element vector that computes the local mean of each block which is called the DC component. After removing the DC component, we apply PCA on residuals. Since each block has 25 dimensions, we get one DC component plus 24 AC components, whose kernels are eigenvectors of the covariance matrix of the collected blocks, for each channel. In each level, the components generated by each kernel for the blocks of each channel form an output channel (shown by a node in the tree in Fig. 2), e.g. the DC component of blocks extracted from an input channel form one output channel/node. We divide these nodes into three groups:

- Intermediate nodes: The DC and several leading low-frequency AC channels will be forwarded to the next level for further energy compaction.
- Leaf nodes: The nodes which are kept at the current level.
- Discarded nodes: AC components with very small eigenvalues are discarded.

As we mentioned, in each PixelHop++ unit we apply channel-wise (c/w) transform to pixel blocks; in other words, for pixel blocks extracted from each individual input channel, an individual Saab transform is applied. The reason we can process channels individually is that all AC channels are orthogonal to the DC channel and all AC responses are uncorrelated due to PCA. Note that the first PixelHop++ unit of M_{CrCb} is an exception (Cr and Cb channels are not uncorrelated), so we apply Saab transform on blocks of $5\times5\times2$ pixels at this level and obtain 1 DC component and 49 AC components for each pixel block. We should emphasize that *channel separability* is powerful in reducing our model size and computational complexity. Unlike DNNs, PixelHop++ does not transform one large 3D (i.e., 2D spatial plus 1D spectral) tensor but multiple 2D spatial tensors.

Each level of PixelHop++ provides an approximation to the input with different spatial/frequency tradeoffs. The input is a pure spatial representation. The output of level-3 is a pure frequency representation, and the outputs of level-1 and level-2 are hybrid spatial/frequency representations. A square in Fig. 2 indicates a channel which is the union of all corresponding spatial locations. Before the max-pooling layer, the dimensions of the output channels of level-1 and level-2 are 28×28 and 10×10, respectively. Clearly, level-1 has more spatial detail than level-2. The spatial dimension of level-3 is 1×1. Each intermediate/leaf node at a level indicates a frequency channel at the corresponding level.

For channels at each level, we need two hyper-parameters to partition them into the mentioned three groups. We use the energy level of a channel as the criterion. If its energy is less than a cutoff energy denoted by E_C, the channel is discarded. If its energy is higher than the forwarded energy threshold denoted by E_F, the channel is forwarded to the next level. We normalize the energy of the root node (i.e., the input image) to 100%. The energy level of each intermediate/leaf node is computed as follows.

- Step 1: Initial DC and AC energy computation for each PixelHop++ unit.
  Each eigenvalue of the covariance matrix indicates the energy of its corresponding node. We define the initial energy (E_{init}) of each output node as the ratio of its corresponding eigenvalue to the sum of all other eigenvalues related to that PixelHop++ unit. At this step, the sum of the DC and total AC energy values for each PixelHop++ unit is 100%.
- Step 2: Normalized energy at each node.
  Based on the first step, we have the energy of an intermediate/leaf node against its siblings. By traversing the tree from the root node to a leaf node, the path includes intermediate nodes. The normalized energy of a leaf node against the root node is the product of E_{init} values of all visited nodes (including itself).

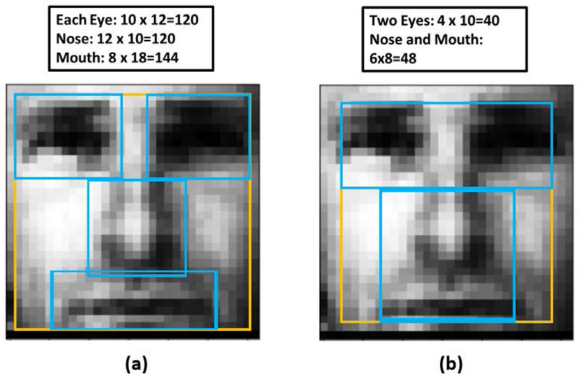


Fig. 3. Illustration of selected spatial regions of interest (ROIs) with respect to the input image for frequency channels at (a) level-1 and (b) level-2.

Note that by lowering E_C and E_F, we can obtain a better approximation at the cost of higher model complexity.

3.2. Pairwise feature generation

To compare whether two face images are similar or not, we can examine their similarities at different spatial regions, channels, and levels. This is feasible because of the rich representations offered by PixelHop++. Note that although the content of an intermediate node will be mainly forwarded to the next level, it does have different spatial/spectral representations at two adjacent levels. Thus, for feature extraction, we do not differentiate intermediate/leaf nodes at each level.

There are K_1, K_2 and K_3 nodes at level-1, level-2 and level-3, respectively, as shown in Fig. 2. We can process them individually as detailed below.

- **Level-1.** It has the highest spatial resolution, i.e., 28×28. We can use it to zoom in on salient regions of the face such as eyes, nose, and mouth at each channel (see Fig. 3(a)) and flatten them to form vectors. Accordingly, we extract 4 feature vectors from each node at this level ($4\times K_1$ feature vectors).
- **Level-2.** It has a lower spatial resolution (10×10). We can still zoom in on unions of salient regions such as one horizontal stripe covering two eyes and one vertical stripe covering the nose and the mouth at each channel (see Fig. 3(b)) and form 2 vectors per node accordingly ($2\times K_2$ feature vectors).
- **Level-3.** It has no spatial resolution. Each leaf node offers a scalar description of the whole face. In our implementation, we concatenate all K_3 nodes into a long sequence and then group every 10 nodes as one vector. Consequently, we obtain P feature vectors, where P is the floor of K_3 divided by 10.

To compare the similarity between two face images, we collect corresponding vector pairs from the same spatial region of the same node (including intermediate and leaf nodes) at each level and compute two similarity measures for them - the cosine similarity (C_k) and the length ratio (R_k) for the k-th vector pair. We define the length ratio of two vectors as the ratio of the vector with smaller L2 norm to the vector with larger L2 norm. If two images are similar, their cosine similarity and length ratio should be close to unity. Otherwise, they should be farther away from (and less than) one. We observe experimentally that an individual R_k value is not as discriminant as C_k. Instead, the average length ratio for each spatial region in level-1 and -2 as shown in Fig. 3 and the

Table 1
Comparison of test accuracy of C_Y and C_{CrCb} and hyper-parameter settings of M_Y and M_{CrCb}, where K_1, K_2, and K_3 are numbers of intermediate and leaf nodes at level-1, level-2, and level-3, P is the number of vectors at level-3 and $N = 7 + 4K_1 + 2K_2 + P$ is the feature dimension.

Input ch.	E_C	K_1	K_2	K_3	P	N	Acc.(%)
Y	0.0005	18	119	233	23	340	83.47
CrCb	0.0004	19	73	124	12	241	75.89

average length ratio for P pairs in level-3 is more robust and discriminant.

To summarize, the ultimate feature vector to be fed to the binary classifier is the concatenation of: 1) 7 average length ratio values (4 extracted from level-1, 2 from level-2, and 1 from level-3), 2) $4K_1$ cosine similarity values from four spatial regions and K_1 channels, 3) $2K_2$ cosine similarity values from two spatial regions and K_2 channels, and 4) P cosine similarity values from groups of channels in level-3. The value of the feature dimension (N) for each submodel is given in Table 1.

3.3. Classifiers

As described in Section 3.2, we extract the feature vector from the Y channel and the CrCb channels of each pair separately. For each pairwise feature, we train a classifier which makes a soft decision, i.e., the probability for the pair to be match or mismatch. Then, we feed these two probabilities into a meta classifier for the final decision. We use the Logistic Regression (LR) classifier in our experiments to achieve a smaller model size although using larger binary classifiers we may achieve higher accuracy.

4. Integration with active learning

The feature generation process in LRFRHop is unsupervised, and labels are only needed for classifier training. As a motivation for active learning, a scenario of interest is training a model on a mobile agent when the model has access to unlabeled training samples locally but has to fetch the label of a limited number of samples from the server through unreliable low-bandwidth channels. To overcome the communication constraint, active learning can be used to retrieve labels of most informative samples. We consider three active learning methods as explained below.

1. *Entropy method*: In each iteration, the entropy of each sample in the pool of unlabeled data D^u is computed and the samples with higher entropy are picked and added to the labeled training data. Higher entropy means the model is more uncertain about those samples. The entropy for sample X can be computed using

$$entropy(x) = -\sum_{j=1}^{J} p(y_i \mid x) \log p(y_i \mid x), \tag{1}$$

where $p(y_i \mid x)$ is the probability that sample x belongs to the class label y_i.
2. *Query By Committee (QBC)*: Instead of training one model, this method trains several models called a committee. In each iteration, the disagreement between the committee members for each sample in D^u is measured and the samples with the largest disagreement values are picked. One of the common disagreement measures is *Vote Entropy* which can be computed using

$$VE(x) = -\sum_{j=1}^{J} \frac{V(y_i)}{C} \log \frac{V(y_i)}{C}, \tag{2}$$

where C is the number of models, $V(y_i)$ is the number of votes which label y_i receives from committee members.
3. *Core-set method*: The core-set selection problem is choosing b sample points from the pool of unlabeled data that minimize the maximum distance between each data point remaining in the pool and its nearest data point in the selected subset. This problem is NP-Hard. A greedy approach for core-set selection is the k-Center-Greedy algorithm. In the i^{th} iteration, it selects the samples from D^u with the maximum distance from their closest sample in the labeled training data D^l_{i-1}.

5. Experiments

We evaluate the performance of the proposed method by conducting experiments on two well-known datasets: Labeled Faces in the Wild (LFW) [14] and CMU Multi-PIE [13]. In all experiments, we use low-resolution face images of size 32×32 unless otherwise specified.

The ***LFW dataset*** is a widely used dataset for face verification. It consists of 13,233 face images of 5749 individuals. For performance benchmarking of different face verification models, it provides 6000 face pairs in 10 splits. We follow the “Image-Restricted, Label-Free Outside Data” protocol. We choose this protocol as we use a tool for facial landmark localization in the preprocessing step. But for training the model, we only use face images in the LFW training set. A 3D aligned version of LFW [8] is used in experiments. For data augmentation, we add the pair of horizontally filliped images of each training pair to the training data.

The ***CMU Multi-PIE dataset*** contains more than 750,000 images of 337 people recorded in four sessions. For each identity, images are captured under 15 viewpoints and 19 illumination conditions with a few different facial expressions. In experiments, we select a subset of 01 session which contains frontal and slightly non-frontal face images (camera views 05_0, 05_1, and 14_0) with a neutral expression and under all existing illumination conditions.

Preprocessing. Several commonly used face processing techniques are adopted in the preprocessing step such as:

1. Applying a 2D face alignment algorithm to input face images to reduce the effect of pose variation;
2. Cropping face images properly to eliminate background;
3. Using histogram equalization (HE) to reduce the effect of different illumination conditions on the Y channel.

We use the dlib [18] toolkit for face detection and landmark localization. Face images are aligned/normalized so that the line connecting the eye centers is horizontal and all faces are centered and resized into a constant size of 32×32pixels. We convert the color space to YCrCb and, then feed the Y channel (the luminance component) to one designated submodel and the Cr and Cb channels (chrominance components) to another submodel.

5.1. Face verification on LFW

Experiment #1. We use the first 90% of the LFW training pairs for training the model and the rest of the pairs for testing. We set $E_C = E_F$ so that the number of leaf nodes in the first and second hop is zero, and study the effect of changing this hyper-parameter on the test accuracy of each submodel. As shown in Figs. 4 and 5 by increasing the energy threshold, the accuracy initially increases and then decreases gracefully while the model size decreases substantially, demonstrating that this architecture has a trade-off between accuracy and model size. For M_Y we select $E_C = E_F = 0.0004$ which gives the best accuracy while having a reasonable model size. For M_{CrCb}, although $E_C = E_F = 0.0001$ gives the best accuracy, we select $E_C = E_F = 0.0005$ because, compared with the operating

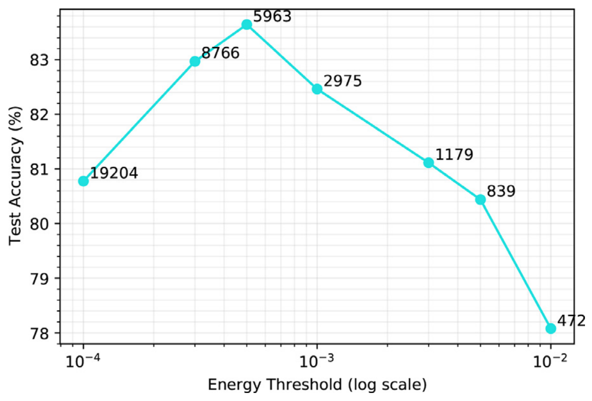


Fig. 4. The relation between the test accuracy (%) and the energy threshold in M_Y, where the number of model parameters in M_Y is shown at each operational point.

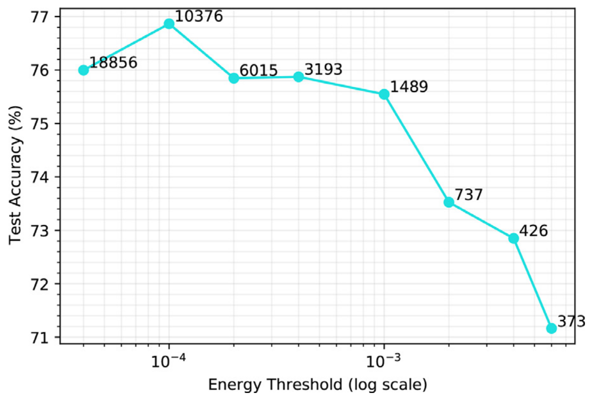


Fig. 5. The relation between the test accuracy (%) and the energy threshold in M_{CrCb}, where the number of model parameters in M_{CrCb} is shown at each operational point.

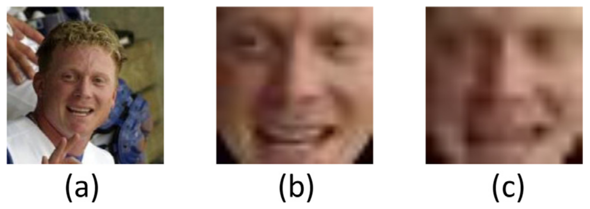


Fig. 6. Quality of the obtained 32×32 (b) and 16×16 (c) low-resolution face images compared with the original high-resolution face image (a).

point of the highest accuracy, it has a considerably smaller model size (3× smaller) and gives only slightly lower accuracy.

Using these hyper-parameters, the parameters of each PixelHop++ block in LRFRHop and the accuracy of the classifiers trained on their output features are given in Table 1. The test accuracy of the meta classifier (C_M) under this setting is 85.33%. We use the determined hyper-parameters in all experiments.

Experiment #2. To compare LRFRHop with the state-of-the-art models, we compute the 10-fold cross-validation accuracy for input resolutions of 32×32 and 16×16 pixels. To obtain the 16×16 face images, we down-sample images to 16×16 and then resize them back to 32×32. The quality of the obtained face images is compared in Fig. 6. To the best of our knowledge, there is no low-resolution to low-resolution face recognition model which has reported accuracy under the “Image-Restricted, Label-Free Outside Data” protocol on LFW. To this end, we compare our model with SKD (Selective Knowledge Distillation) [11] and BD (Bridge Distillation) [12] which are two state-of-the-art low-resolution face recognition models using extensive training data (under the “Unrestricted With Labeled Outside Data” protocol) and are evaluated for input resolutions of 32×32 and 16×16 pixels.

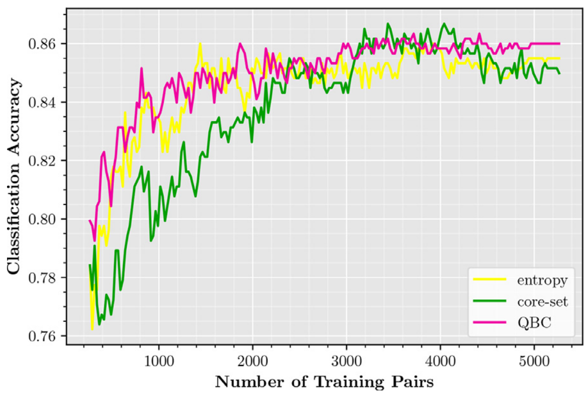


Fig. 7. Comparison of classification accuracy of three active learning methods as a function of the number of training pairs.

We compare LRFRHop with SKD without distillation, SKD, BD without distillation, and BD in terms of accuracy, the number of parameters, and the number of training images for the input resolution of 16×16 and 32×32 in Tables 2 and 3, respectively. According to Table 2, for the case of 16×16 resolution, LRFRHop achieves an accuracy of 82.16% which is only 3.71%, and 3.72% lower than SKD and BD, respectively, while its model size is about 79× smaller than SKD and 21× smaller than BD and uses only 5400 pair of images as the training set. On the other hand, a pre-trained model on the VGGFace dataset [26] with 2.6 M images is used as the teacher model of SKD and its student model is trained and fine-tuned on the UMDFaces dataset [1] with 367,888 images. BD uses an ensemble of two models as the teacher model. One of them is trained on the CASIA-WebFace dataset [37] with 0.49 M images and the other is trained on the VGGFace2 dataset [3] with 3.31 M images. The student model of BD is trained on UMDFaces. Also, note that LRFRHop outperforms SKD without distillation and BD without distillation. According to Table 3, for the case of 32×32 resolution, LRFRHop achieves an accuracy of 83.53% which is only 6.19%, and 5.69% lower than SKD and BD, respectively, while its model size is considerably smaller and uses a much smaller training set as mentioned before. Based on these results, LRFRHop does have a competitive performance for deployment in resource-constrained environments.

FSLR (Fewer-Shots and Lower-Resolutions) [10] is another state-of-the-art low-resolution face recognition model which is designed and optimized exclusively for 16×16 images. It has achieved state-of-the-art verification accuracy on 16×16 LFW images but its model size is 11 larger than LRFRHop. Regarding the training set, a pre-trained teacher model on a massive dataset containing 3.8 M face images is used to train the student of FSLR on UMDFaces.

Experiment #3. In this experiment, we use the three active learning methods which are introduced in Section 4 to obtain the minimum required number of labeled training pairs without significant loss in recognition accuracy. We use the first 90% of the LFW training pairs as the initial pool of unlabeled data (D_u) and the rest as the test data, then we randomly select 5% of the pool as D_0. The data augmentation step is removed. For the QBC algorithm, we use two different classifiers (LR and SVM) as the committee of classifiers. We apply each active learning algorithm to samples and compute the test accuracy versus the number of labeled training pairs used for training the model. The result is shown in Fig. 7.

By comparing the performance of different algorithms, we see that QBC is the most effective one for LRFRHop in the current experiment setting as it has the fastest convergence in test accuracy and also the most stable performance. For example, with this algo-

Table 2
Face verification results on LFW for 16×16 images.

Model	#Param.	Training Set (#Img.)	Acc.(%)
SKD without distillation	0.79 M	UMDFaces (367,888)	62.82
SKD	0.79 M	VGGFace (2.6 M), UMDFaces (367,888)	85.87
BD without distillation	0.21 M	UMDFaces (367,888)	81.26
BD	0.21 M	VGGFace2 (3.31 M), CASIA-WebFace (0.49M), UMDFaces (367,888)	85.88
LRFRHop	0.01 M	LFW (5400 pairs)	82.16

Table 3
Face verification results on LFW for 32×32 images.

Model	#Param.	Training Set (#Img.)	Acc.(%)
SKD without distillation	0.79 M	UMDFaces (367,888)	70.23
SKD	0.79 M	VGGFace (2.6 M), UMDFaces (367,888)	89.72
BD without distillation	0.21 M	UMDFaces (367,888)	85.8
BD	0.21 M	VGGFace2 (3.31 M), CASIA-WebFace (0.49M), UMDFaces (367,888)	89.22
LRFRHop	0.01 M	LFW (5400 pairs)	83.53

rithm, LRFRHop can achieve an accuracy of above 84% using only 1465 training pairs and a stable accuracy of about 86% with only 3000 pairs.

5.2. Face identification on CMU Mmulti-PIE

We evaluate the effectiveness of LRFRHop for the face identification task by conducting experiment on the Multi-PIE dataset. To the best of our knowledge, no low-resolution face recognition model has reported results on this dataset. Thus, we compare our model with high-resolution face recognition models that have used this dataset for performance benchmarking. We use a setting mentioned in [38] as Setting-1. According to this setting, the first 150 identities are used for training and the rest of the identities (151–250) from session 01 with neutral expression are used for testing so that there is no overlap between training and test identities. As the gallery image, one frontal face image with frontal illumination is used and the rest of the images in the test set are selected as probes. For model training, we generate 17,888 training pairs by pairing probe images with gallery images and 21,000 training pairs by randomly pairing images with each other. Overall, 38,888 training pairs are used to train LRFRHop.

Since our model is not proposed for handling extreme pose variation, we report results only for frontal and slightly non-frontal images (±15°) in the dataset. The rank-1 face identification accuracy is shown in Table. LRFRHop has a competitive and comparable performance although it uses low-resolution face images, a small training set, and a small model size. For example, Light CNN-29 has 12.637M parameters and uses 128×128 images while our model size is 1149× smaller and trained on 32×32 images. It is worthwhile to comment that TP-GAN in Table also uses Light CNN-29 for feature extraction from images so it is larger than Light CNN-29.

Table 4
Rank-1 identification rate (%) for frontal and slightly non-frontal face images (±15°) in Setting-1.

Method	Resolution	Acc.(%)
CPF [38]	6060	89.45
HPN [7]	256,220	84.23
c-CNN Forest [35]	-	96.97
Light CNN-29 [34]	128,128	99.78
TP-GAN [15]	128,128	99.78
LRFRHop	3232	89.48

Table 5
The number of parameters of each component in LRFRHop.

Subsystem	Num. of Param.
M_Y	451 + 2543 + 2969
Pairwise Feat. Gen. - Y	740
LR Classifier - C_Y	341
M_{CrCb}	476 + 1369 + 1348
Pairwise Feat. Gen. - CrCb	432
LR Classifier - C_{CrCb}	242
Meta Classifier	3
Total	10,914

5.3. Model size computation and time complexity

We compute the size of LRFRHop based on the information provided in Table 1. Each PixelHop++ system in LRFRHop has three levels and the c/w Saab transform is applied in each level. The first level of the M_Y system has 18 intermediate nodes and 7 discarded nodes (25-18). Thus, it has 25×18 variables for storing PCA components and one bias parameter (451 parameters in total). The second level has 119 intermediate nodes and 25×18-119 discarded nodes. For each of the 18 output channels of level one, the Saab transform is applied separately. Initially, 25×18 nodes are generated, and 119 of them are then selected as intermediate nodes based on the energy threshold. For each Saab transform, the DC kernel is known and constant so we have 18 repetitive kernels in the second level. As a result, we have 25×(119-18) parameters for storing PCA components and 18 bias parameters which is in total 2543 parameters. By the same token, there are 25×(233-119)+119 parameters in the third level of M_Y. In the Pairwise Feature Generator - Y unit, the mean and the standard deviation of each PixelHop++ output node are stored. Hence, the size of the unit is 2×(18+119+233) equal to 740. The LR classifier's size equals its input feature size plus one which is 341 for C_Y. Likewise, the size of all subsystems is computed and summarized in Table 5.

Training LRFRHop for experiment #1 takes about 20 minutes and its inference time is 103 milliseconds per image pair. The hardware platform for training and inference time measurement was Intel(R) Core(TM) i7-7660U CPU @ 2.50GHz.

6. Conclusion and future work

A lightweight data-efficient low-resolution face recognition model for resource-constrained environments, called LRFRHop, was proposed in this paper. We plan to develop a similar methodology for face ethnicity, age, and gender recognition in resource-constrained environments. Furthermore, occluded face recognition [9,21] and extreme pose variation in resource-constrained environments are two challenging problems, and the generalization of LRFRHop to these challenges is a valuable future research topic.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was sponsored by US Army Research Office (ARO) under contract number W911NF2020157.

References

[1] A. Bansal, A. Nanduri, C.D. Castillo, R. Ranjan, R. Chellappa, Umdfaces: An annotated face dataset for training deep networks, in: 2017 IEEE International Joint Conference on Biometrics (IJCB), IEEE, 2017, pp. 464–473.
[2] S. Biswas, K.W. Bowyer, P.J. Flynn, Multidimensional scaling for matching low-resolution face images, IEEE transactions on pattern analysis and machine intelligence 34 (10) (2011) 2019–2030.
[3] Q. Cao, L. Shen, W. Xie, O.M. Parkhi, A. Zisserman, Vggface2: A dataset for recognising faces across pose and age, in: 2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018), IEEE, 2018, pp. 67–74.
[4] S. Chen, Y. Liu, X. Gao, Z. Han, Mobilefacenets: Efficient cnns for accurate real-time face verification on mobile devices, in: Chinese Conference on Biometric Recognition, Springer, 2018, pp. 428–438.
[5] Y. Chen, M. Rouhsedaghat, S. You, R. Rao, C.-C.J. Kuo, Pixelhop++: A small successive-subspace-learning-based (ssl-based) model for image classification, in: 2020 IEEE International Conference on Image Processing (ICIP), IEEE, 2020, pp. 3294–3298.
[6] Z. Cheng, X. Zhu, S. Gong, Low-resolution face recognition, in: Asian Conference on Computer Vision, Springer, 2018, pp. 605–621.
[7] C. Ding, D. Tao, Pose-invariant face recognition with homography-based normalization, Pattern Recognition 66 (2017) 144–152.
[8] C. Ferrari, G. Lisanti, S. Berretti, A. Del Bimbo, Effective 3d based frontalization for unconstrained face recognition, in: 2016 23rd International Conference on Pattern Recognition (ICPR), IEEE, 2016, pp. 1047–1052.
[9] S. Ge, C. Li, S. Zhao, D. Zeng, Occluded face recognition in the wild by identity-diversity inpainting, IEEE Transactions on Circuits and Systems for Video Technology 30 (10) (2020) 3387–3397.
[10] S. Ge, S. Zhao, X. Gao, J. Li, Fewer-shots and lower-resolutions: Towards ultrafast face recognition in the wild, in: Proceedings of the 27th ACM International Conference on Multimedia, 2019, pp. 229–237.
[11] S. Ge, S. Zhao, C. Li, J. Li, Low-resolution face recognition in the wild via selective knowledge distillation, IEEE Transactions on Image Processing 28 (4) (2018) 2051–2062.
[12] S. Ge, S. Zhao, C. Li, Y. Zhang, J. Li, Efficient low-resolution face recognition via bridge distillation, IEEE Transactions on Image Processing 29 (2020) 6898–6908.
[13] R. Gross, I. Matthews, J. Cohn, T. Kanade, S. Baker, Multi-pie, Image and Vision Computing 28 (5) (2010) 807–813.
[14] G.B. Huang, M. Ramesh, T. Berg, E. Learned-Miller, Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments, Technical Report, University of Massachusetts, Amherst, 2007.
[15] R. Huang, S. Zhang, T. Li, R. He, Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2439–2448.
[16] Y. Huang, P. Shen, Y. Tai, S. Li, X. Liu, J. Li, F. Huang, R. Ji, Improving face recognition from hard samples via distribution distillation loss, in: European Conference on Computer Vision, Springer, 2020, pp. 138–154.
[17] S.S. Khalid, M. Awais, Z.-H. Feng, C.-H. Chan, A. Farooq, A. Akbari, J. Kittler, Resolution invariant face recognition using a distillation approach, IEEE Transactions on Biometrics, Behavior, and Identity Science 2 (4) (2020) 410–420.
[18] D.E. King, Dlib-ml: A machine learning toolkit, Journal of Machine Learning Research 10 (2009) 1755–1758.
[19] C.-C.J. Kuo, Understanding convolutional neural networks with a mathematical model, Journal of Visual Communication and Image Representation 41 (2016) 406–413.
[20] C.-C.J. Kuo, M. Zhang, S. Li, J. Duan, Y. Chen, Interpretable convolutional neural networks via feedforward design, Journal of Visual Communication and Image Representation 60 (2019) 346–359.
[21] C. Li, S. Ge, D. Zhang, J. Li, Look through masks: Towards masked face recognition with de-occlusion distillation, in: Proceedings of the 28th ACM International Conference on Multimedia, 2020, pp. 3016–3024.
[22] X. Li, F. Wang, Q. Hu, C. Leng, Airface: lightweight and efficient model for face recognition, in: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, 2019, p. 0.
[23] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, L. Song, Sphereface: Deep hypersphere embedding for face recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 212–220.
[24] Z. Lu, X. Jiang, A. Kot, Deep coupled resnet for low-resolution face recognition, IEEE Signal Processing Letters 25 (4) (2018) 526–530.
[25] S.P. Mudunuri, S. Biswas, Low resolution face recognition across variations in pose and illumination, IEEE transactions on pattern analysis and machine intelligence 38 (5) (2015) 1034–1040.
[26] Parkhi, O. M., Vedaldi, A., Zisserman, A., 2015. Deep face recognition.
[27] M. Rouhsedaghat, Y. Wang, X. Ge, S. Hu, S. You, C.-C.J. Kuo, Facehop: A light-weight low-resolution face gender classification method, ICPR Workshops, 2020.
[28] F. Schroff, D. Kalenichenko, J. Philbin, Facenet: A unified embedding for face recognition and clustering, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 815–823.
[29] O. Sener, S. Savarese, Active learning for convolutional neural networks: Acore-set approach, arXiv preprint arXiv:1708.00489 (2017).
[30] H.S. Seung, M. Opper, H. Sompolinsky, Query by committee, in: Proceedings of the fifth annual workshop on Computational learning theory, 1992, pp. 287–294.
[31] C.E. Shannon, A mathematical theory of communication, Bell system technical journal 27 (3) (1948) 379–423.
[32] Y. Taigman, M. Yang, M. Ranzato, L. Wolf, Deepface: Closing the gap to human-level performance in face verification, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 1701–1708.
[33] K. Wang, D. Zhang, Y. Li, R. Zhang, L. Lin, Cost-effective active learning for deep image classification, IEEE Transactions on Circuits and Systems for Video Technology 27 (12) (2016) 2591–2600.
[34] X. Wu, R. He, Z. Sun, T. Tan, A light cnn for deep face representation with noisy labels, IEEE Transactions on Information Forensics and Security 13 (11) (2018) 2884–2896.
[35] C. Xiong, X. Zhao, D. Tang, K. Jayashree, S. Yan, T.-K. Kim, Conditional convolutional neural network for modality-aware face recognition, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 3667–3675.
[36] H. Ye, W. Shao, H. Wang, J. Ma, L. Wang, Y. Zheng, X. Xue, Face recognition via active annotation and learning, in: Proceedings of the 24th ACM international conference on Multimedia, 2016, pp. 1058–1062.
[37] D. Yi, Z. Lei, S. Liao, S.Z. Li, Learning face representation from scratch, arXiv preprint arXiv:1411.7923 (2014).
[38] J. Yim, H. Jung, B. Yoo, C. Choi, D. Park, J. Kim, Rotating your face using multitask deep neural network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 676–684.