When Face Recognition Meets with Deep Learning: an Evaluation of Convolutional Neural Networks for Face Recognition

Guosheng Hu* ♡ ♣, Yongxin Yang* ◇, Dong Yi♠, Josef Kittler♣, William Christmas♣, Stan Z. Li♠, Timothy Hospedales◇
LEAR team, Inria Grenoble Rhone-Alpes, 38330 Montbonnot, France ♡
Centre for Vision, Speech and Signal Processing, University of Surrey, UK♣
Electronic Engineering and Computer Science, Queen Mary University of London, UK◇
Center for Biometrics and Security Research & National Laboratory of Pattern Recognition, Chinese Academy of Sciences, China♠
{g.hu,j.kittler,w.christmas}@surrey.ac.uk,{yongxin.yang, t.hospedales}@qmul.ac.uk, {szli,dyi}@cbsr.ia.ac.cn

Abstract

Deep learning, in particular Convolutional Neural Network (CNN), has achieved promising results in face recognition recently. However, it remains an open question: why CNNs work well and how to design a 'good' architecture. Existing studies tend to focus on reporting CNN architectures that work well for face recognition rather than investigate the reason. In this work, we conduct an extensive evaluation of CNN-based face recognition systems (CNN-FRS) on a common ground to make our work easily reproducible. Specifically, we use public database LFW (Labeled Faces in the Wild) to train CNNs, unlike most existing CNNs trained on private databases. We propose three CNN architectures which are the first reported architectures trained using LFW data. This paper quantitatively compares the architectures of CNNs and evaluates the effect of different implementation choices. We identify several useful properties of CNN-FRS. For instance, the dimensionality of the learned features can be significantly reduced without adverse effect on face recognition accuracy. In addition, a traditional metric learning method exploiting CNN-learned features is evaluated. Experiments show two crucial factors to good CNN-FRS performance are the fusion of multiple CNNs and metric learning. For reproducibility, our source code and models will be made publicly available.

The conventional face recognition pipeline consists of four stages: face detection, face alignment, feature extraction (or face representation) and classification. Perhaps the single most important stage is feature extraction. In constrained environments, hand-crafted features such as Local Binary Patterns (LBP) [1] and Local Phase Quantisation (LPQ) [2, 4] have achieved respectable face recognition performance. However, the performance using these features degrades dramatically in unconstrained environments where face images cover complex and large intra-personal variations such as pose, illumination, expression and occlusion. It remains an open problem to find an ideal facial feature which is robust for face recognition in unconstrained environments (FRUE). In the last three years, convolutional neural networks (CNN) rebranded as 'deep learning' have achieved very impressive results on FRUE. Unlike the traditional hand-crafted features, the CNN learned features are more robust to complex intra-personal variations. Notably, the top three face recognition rates reported on the FRUE benchmark database LFW (Labeled Faces in the Wild) [12] have been achieved by CNN methods [33, 25, 21]. The success of the latest CNNs on FRUE and more general object recognition task [14, 9, 13] stems from the following: (1) much larger labeled training sets are available; (2) GPU implementations greatly reduce the time to train a large CNN and (3) CNNs greatly improve the model generation capacity by introducing effective regularisation strategies, such as dropout [10].

Despite the promising performance achieved by CNNs, it remains unclear how to design a 'good' CNN architecture for a specific classification task due to the lack of theoretical guidance. However, some insights into CNN design can be gained by experimental comparisons of different CNN architectures. The work [5] made such comparisons and comprehensive analysis for the task of object recognition. However, face recognition is very different from object recognition. Specifically, faces are aligned via 2D similarity transformation or 3D pose correction to a fixed reference position in images before feature extraction while object recognition usually does not conduct such alignment, and therefore objects appear in arbitrary positions. As a result, the CNN architectures used for face recognition [24, 21, 25, 33, 28] are usually different from those for object recognition [14, 20, 26, 9]. For the task

* indicates equal contribution

of face recognition, there exists no systematic evaluation of the effect of different CNN design and implementation choices. In addition, published CNNs [24, 33, 28, 31] are trained on different face databases, most of which are not publicly available. The difference of training sets might result in unfair comparisons of CNN architectures. To avoid this, the comparison of different CNNs should be conducted on a common ground.

In this paper, we clarify the contributions of different components of CNN-based face recognition systems by conducting a systematic evaluation. To make our work reproducible, all the networks evaluated are trained on the publicly available LFW database. Specifically, our contributions are as follows:

- Different CNN architectures including number of filters and layers are compared. We also evaluate the face recognition performance using features from different layers: pooling, fully connected and softmax layers. We find the features from softmax layer perform slightly better than those from the most widely used fully connected layer. To our knowledge, this is the first work that compares the performance of these features for face recognition.

- Various implementation details, such as data augmentation, pixel value type (colour or grey) and similarity, are evaluated.

- We quantitatively analyse how downstream metric learning methods such as joint Bayesian [6] can boost the effectiveness of the CNN-learned features. In addition, we evaluate the impact of multiple network fusion introduced by [24].

- Finally, source code for our CNN architectures will be made publicly available (the training data is already public). This will accelerate future research, and also provides a competitive baseline for face recognition to the community.

1. Related Work

CNN methods have drawn considerable attention in the field of face recognition in recent years. In particular, CNNs have achieved impressive results on FRUE. In this section, we briefly review these CNNs.

The researchers in Facebook AI group trained an 8-layer CNN named DeepFace [28]. The first three layers are conventional convolution-pooling-convolution layers. The subsequent three layers are locally connected, followed by 2 fully connected layers. Pooling layers make learned features robust to local transformations but result in missing local texture details. Pooling layers are important for object recognition since the objects in images are not well aligned. However, face images are well aligned before training a CNN. It is claimed in [28] that one pooling layer is a good balance between local transformation robustness and preserving texture details. DeepFace is trained on a large face database which contains four million facial images of 4,000 subjects. Another contribution of [28] is the 3D face alignment. Traditionally, face images are aligned using 2D similarity transformation before they are fed into CNNs. However, this 2D alignment cannot handle out-of-plane rotations. To overcome this limitation, [28] proposes a 3D alignment method using an affine camera model.

In [24], a CNN-based face representation, referred to as Deep hidden IDentity feature (DeepID), is proposed. Unlike DeepFace whose features are learned by one single big CNN, DeepID is learned by training an ensemble of small CNNs, used for network fusion. The input of one single CNN is the crops/patches of facial images and the features learned by all CNNs are concatenated to form a powerful feature. Both RGB and grey crops extracted around facial points are used to train the DeepID. The length of DeepID is 2 (RGB and Grey) $\times$ 60 (crops) $\times$ 160 (feature length of one network) = 19,200. Each network consists of 4 convolutional layers, 3 max pooling layers and 2 fully connected layers shown in Table 1. DeepID uses identification information only to supervise the CNN training. In comparison, DeepID2 [21], an extension of DeepID, uses both identification and verification information to train a CNN, aiming to maximise the inter-class difference but minimise the intra-class variations. To further improve the performance of DeepID and DeepID2, DeepID2+ [25] is proposed. DeepID2+ adds the supervision information to all the convolutional layers rather than the topmost layers like DeepID and DeepID2. In addition, DeepID2+ improves the number of filters of each layer and uses a much bigger training set than DeepID and DeepID2 . In [25], it is also discovered that DeepID2+ has three interesting properties: being sparse, selective and robust.

The work [31] proposes another face recognition pipeline, referred to as WebFace, which also learns the face representation using a CNN. WebFace collects a database which contains around 10,000 subjects and 500,000 images and makes this database publicly available. Motivated by the very deep architectures of [20, 26], WebFace trains a much deeper CNN than those [24, 21, 25, 28] used for face recognition as shown in Table 1. Specifically, WebFace trains a 17-layer CNN which includes 10 convolutional layers, 5 pooling layers and 2 fully connected layers detailed in Table 1. Note that the use of very small convolutional filters (3×3), which avoids too much texture information decrease along a very deep architecture, is crucial to learn a powerful feature. In addition, WebFace stacks two 3×3 convolutional layers (without pooling in between) which is as effective as a 5×5 convolutional layer but with fewer parameters.

Table 1. Comparisons of 3 Published CNNs

	Input Image [1]	Architecture [2]	No. of para.	Patch Fusion	Feature Length	Training set
DeepFace [28]	152×152×3	C1:32×11×11,[3] M2, C3:16×9×9, L4: 16×9×9, L5:16×7×7, L6:16×5×5, F7, Loss	120M+	No	4096	120M+ images 4K+ subjects
DeepID [24]	39×31×{3,1} 31×31×{3,1}	C1:20×4×4, M2, C3:40×3×3, M4, C5:60×3×3, M6, C7:80×2×2, F8, Loss	101M+	Yes	19200	100K+ images 10K+ subjects
WebFace [31]	100×100×1	C1:32×3×3, C2:64×3×3, M3, C4:64×3×3, C5:128×3×3, M6, C7:96×3×3, C8:192×3×3, M9, C10:128×3×3, C11:256×3×3, M12, C13:160×3×3, C14:320×3×3, A15, F16, Loss	5M+	No	320	500K+ images 10K subjects

[1] The input image is represented as width×height×channels. 1 and 3 mean grey or RGB images respectively.
[2] The capital letters C, M, L, A, F represent convolutional, max pooling, locally connected, average pooling and fully connected layers.
[3] The number of filters and filter size are denoted as 'num × size × size'

Because face recognition is a special case of object recognition, good architectures for object recognition can be introduced for face recognition. Motivated by this, the work Facenet [18] adapted Zeiler&Fergus [32] style networks and the recent Inception [26] type networks from the field of object recognition to face recognition. Unlike the other face CNNs [31, 21, 28] which learn a metric or classifier, Facenet simply uses the euclidean distance to determine the classification of *same* and *different*, showing that the learned features are very discriminative. Finally DeepID3 [22] also modified two famous networks: Inception [26] and VGG Net [20] by adding supervision information to each layer and network ensemble fusion.

Table 1 compares three typical CNNs (DeepFace [28], DeepID [24], WebFace [31]). It is clear that their architectures and implementation choices are rather different, which motivates our work. In this study, we make systematic evaluations to clarify the contributions of different components on a common ground.

2. Methodology

LFW is the *de facto* benchmark database for FRUE. Most existing CNNs [28, 24, 21, 25] train their networks on private databases and test the trained models on LFW. In comparison, we train our CNNs only using LFW data to make our work easily reproducible. In this way, we cannot directly use the reported CNN architectures [28, 24, 21, 25, 31] since our training data is much less extensive. We introduce three architectures adapted to our training set in subsection 2.1. To further improve the discrimination of CNN-learned features, metric learning method is usually used. One state of the art metric learning method, the Joint Bayesian model [6], is detailed in subsection 2.2.

2.1. CNN Architectures

How to design a 'good' CNN architecture remains an open problem. Generally, the architecture depends on the size of training data. Less data should drive a smaller network (fewer layers and filters) to avoid overfitting. In this study, the size of our training data is much smaller than those used by [28, 24, 21, 25, 31]; therefore, smaller architectures are designed.

We propose three CNN architectures suitable for LFW. These architectures are of three different sizes: small (CNN-S), medium (CNN-M), and large (CNN-L). CNN-S and CNN-M have 3 convolutional layers and two fully connected layers, while CNN-M has more filters than CNN-S. Compared with CNN-S and CNN-M, CNN-L has 4 convolutional layers. The activation function we used is REctification Linear Unit (RELU) [14]. In our experiments, dropout [10] did not improve the performance of our CNNs, therefore, it is not applied to our networks. Following [24, 31], softmax is used in the last layer for predicting one of K (the number of subjects in the context of face recognition) mutually exclusive classes. During training, the learning rate is set to 0.001 for three networks, and the batch size is fixed to 100. Table 2 details these three architectures.

2.2. Metric Learning

Metric Learning (MeL), especially Discriminative Metric Learning is a popular means to enhance the feature with the goal that the similarity measure serves as a better bridge to the label similarity. Intuitively, it aims to 'pull' the objects that have the same label closer while 'pushing' the objects that have different labels away. In the area of face verification, metric learning is usually an extra step that tunes the feature learned from the former steps before feeding them into a classifier. Among MeL methods, Joint Bayesian

Table 2. Our CNN Architectures

CNN-S	CNN-M	CNN-L
conv1		
$12 \times 5 \times 5$ st. 1, pad 0 x2 pool	$16 \times 5 \times 5$ st. 1, pad 0 x2 pool	$16 \times 3 \times 3$ st. 1, pad 1 -
conv2		
$24 \times 4 \times 4$ st. 1, pad 0 x2 pool	$32 \times 4 \times 4$ st. 1, pad 0 x2 pool	$16 \times 3 \times 3$ st. 1, pad 1 x2 pool
conv3		
$32 \times 3 \times 3$ st. 2, pad 0 x2 pool	$48 \times 3 \times 3$ st. 2, pad 0 x2 pool	$32 \times 3 \times 3$ st. 1, pad 1 x3 pool, st. 2
conv4		
-	-	$48 \times 3 \times 3$ st. 1, pad 1 x2 pool
fully connected		
160	160	160
5000, softmax	5000, softmax	5000, softmax

Convolutional layers are detailed in 3 sub-rows: the 1st indicates the number of filters and filter size; the 2nd specifies the convolutional stride ('st.') and padding ('pad'); and the 3rd specifies the max-pooling downsampling factor. For fully connected layers, we specify their dimensionality: 160 for feature length and 5000 for the approximate number of classes/subjects.

(JB) [6] model is the most widely used one applied to the learned CNN features [24, 21, 31].

JB models the face verification task as a Bayesian decision problem. Let H_I and H_E represent intra-personal (matched) and extra-personal (unmatched) hypotheses, respectively. Based on the MAP (Maximum a Posteriori) rule, the decision is made by:

$$r(x_1, x_2) = \log \frac{P(x_1, x_2 \mid H_I)}{P(x_1, x_2 \mid H_E)} \tag{1}$$

where x_1 and x_2 are features of one face pair. It is assumed that $P(x_1, x_2 \mid H_I)$ and $P(x_1, x_2 \mid H_E)$ have Gaussian distributions $N(0, S_I)$ and $N(0, S_E)$, respectively.

Before discussing the way of computing S_I and S_E, we first explain the distribution of a face feature. A face x is modelled by the sum of two independent Gaussian variables (identity μ and intra-personal variations ε):

$$x = \mu + \varepsilon \tag{2}$$

μ and ε follow two Gaussian distributions $N(0, S_\mu)$ and $N(0, S_\varepsilon)$, respectively. S_μ and S_ε are two unknown covariance matrices and they are regarded as face prior. For the case of two faces, the joint distribution of $\{x_1, x_2\}$ is also assumed as a Gaussian with zero mean. Based on Eq. (2), the covariance of two faces is:

$$\mathrm{cov}(x_1, x_2) = \mathrm{cov}(\mu_1, \mu_2) + \mathrm{cov}(\varepsilon_1, \varepsilon_2) \tag{3}$$

Then S_I and S_E can be derived as:

$$S_I = \begin{vmatrix} S_\mu + S_\varepsilon & S_\mu \\ S_\mu & S_\mu + S_\varepsilon \end{vmatrix} \tag{4}$$

and

$$S_E = \begin{vmatrix} S_\mu + S_\varepsilon & 0 \\ 0 & S_\mu + S_\varepsilon \end{vmatrix} \tag{5}$$

Clearly, $r(x_1, x_2)$ in Eq. (1) only depends on S_μ and S_ε, which are learned from data using an EM algorithm [6].

3. Evaluation

LFW contains 5,749 subjects and 13,233 images and the training and test sets are defined in [12]. LFW defines three standard protocols (*unsupervised, restricted* and *unrestricted*) to evaluate face recognition performance. *'Unrestricted'* protocol is applied here because the information of both subject identities and matched/unmatched labels is used in our system. The face recognition rate is evaluated by mean classification accuracy and standard error of the mean.

The images we used are aligned by deep funneling [11]. Each image is cropped to 58×58 based on the coordinates of two eye centers. It is commonly believed that data augmentation can boost the generalisation capacity of a neural network; therefore, each image is horizontally flipped. The mean of the images is subtracted before network training. The open source implementation MatConvNet [30] is used to train our CNNs. In the following sections, different components of our CNN-based face recognition system are evaluated and analysed.

3.1. Architectures

It is an open problem on how to design the architecture of a neural network. Though the high level structure of CNN usually starts with a number of convolutional layers, followed by a (fewer) number of fully-connected layers, the choices of filter size, number of neurons per layer, and stride size etc are usually determined by trial-and-error. Instead of reporting the final design directly, we show some inferior designs on the way to finding the optimal one. Our strategy is as follows: we start from a relatively small network, then extend it (by adding more neurons and/or layers) while the performance improves, and stop when it gets worse as the CNN size increases, which indicates that overfitting occurs.

In the comparison of different architectures, RGB colour images are fed into CNNs and feature distance is measured by cosine distance. (Note that cosine distance is significantly better than euclidean for this task). The performance of the architectures is compared in Table 3. CNN-M achieves the best face recognition performance, indicating that CNN-M generalises best among these three architectures using only LFW data. From this point, all evaluations are conducted using CNN-M. The face recognition rate 0.7882 of CNN-M is considered as the baseline, and all the remaining investigations will be compared with it.

Table 3. Comparison of our three CNN architectures.

Model	Accuracy
CNN-S	0.7828±0.0046
CNN-M	0.7882±0.0037
CNN-L	0.7807±0.0035

3.2. Implementation Details

Grey vs Colour In [31] and [28], CNNs are trained using grey-level and RGB colour images, respectively. In comparison, both grey and colour images are used in [24]. We quantitatively compare the impact of these two images types on face recognition. Using grey and colour images yields face recognition accuracies of 0.7830±0.0077 and 0.7882±0.0037 respectively. These results are very close. Although colour images contain more information, they do not deliver a significant improvement.

Data Augmentation Data augmentation is a set of label-preserving transforms that introduce some new instances without collecting the new data. Flipping and mirroring images horizontally producing two samples from each, is a commonly used data augmentation technique for face recognition. In all our evaluations, both original and mirrored images are used for training. However, little discussion in the existing work analysed the impact of image flipping during *testing*. Naturally, the test images can also be mirrored. A pair of test images can produce 2 new mirrored ones. These 4 images can generate 4 pairs instead of one original pair. To combine these 4 images/pairs, we implemented two fusion strategies: feature and score fusion. For feature fusion, the learned features of a test image and its mirrored one are concatenated to one feature, which is then used for score computing. For score fusion, 4 scores generated from 4 pairs are averaged to one score. Table 4 compares the three scenarios: no flip during test, feature and score fusion. As is shown in Table 4, mirroring images does improve face recognition performance. In addition, feature fusion works slightly better than score fusion, however, the improvements are not statistically significant.

3.3. Properties of the Learned Features

While a common pipeline of deep learning is to use the output of the penultimate layer (fully-connected layer) as a feature vector and feed it into a classifier, we analyse the impact of some post-processing methods on these feature vectors. It is also interesting to know the performance of other layers apart from the penultimate one. In this section, we analyse (1) the effectiveness of feature normalisation (2) the effects of dimensionality reduction and (3) the efficacy of features from different layers.

Table 4. Comparison of test-time data augmentation strategies

Technique	Accuracy
no flip on test set	0.7882 ± 0.0037
feature fusion	0.7895 ± 0.0036
score fusion	0.7893 ± 0.0035

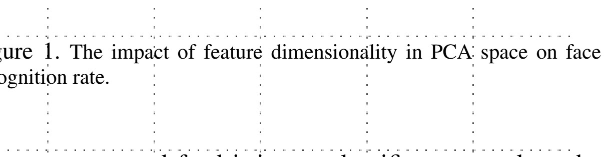
Figure 1. The impact of feature dimensionality in PCA space on face recognition rate.

Feature normalisation First, we discuss feature normalisation, which standardises the range of features and is generally performed as a data preprocessing step. For example, to implement eigenface [29], the features (pixel values) are usually normalised via Eq. (6) before training a PCA space.

$$\hat{\mathbf{x}} = \frac{\mathbf{x} - \mu_{\mathbf{x}}}{\sigma_{\mathbf{x}}} \tag{6}$$

where $\mathbf{x} \in R$ and $\hat{\mathbf{x}} \in R$ are original and normalised feature vectors, respectively. $\mu_{\mathbf{x}}$ and $\sigma_{\mathbf{x}}$ are the mean and standard deviation of $\mathbf{x}$. Motivated by this, our CNN features are normalised by Eq. (6) before computing cosine distance. The accuracies with and without normalisation are 0.7927±0.0040 and 0.7882±0.0037, respectively. Thus normalisation is effective to improve recognition rate.

Feature dimensionality reduction Second, we perform dimensionality reduction on the learned 160D features using PCA. Surprisingly, as shown in Figure 1, only 16 dimensions of the PCA feature space produces comparable face recognition rates to those of the original space. It is a very interesting property of CNN-learned features, because low dimensionality can significantly reduce memory and computation requirements, which is crucial for large scale applications or for mobile devices such as smartphones.

Features of Different Layers We next evaluate the face recognition performance of the features extracted from different layers. It is common knowledge that the features of topmost layers are more discriminative than those of lower layers. It results from the fact that the higher layers can learn more abstract, invariant and semantic features. Usually, the features from the fully connected (FC) layers are used for different computer vision tasks such as object recognition, action recognition and segmentation because these features are compact, invariant and discriminative. Two natural questions are: 1) Do FC features work better than others? and 2) Do FC features and others complement each other? To our knowledge, the existing works about face recognition do not answer these two questions. We answer question 1) in this section and 2) in the next.

For 1), we evaluate the face recognition performance using the features from three topmost layers of CNN-M: the last pooling layer (Pool4), fully connected layer (FC) and softmax layer (Softmax). Though the features obtained from the fully connected layer is commonly used in classifier training, there is very limited spatial information retained in them. On the other hand, the features from the convolutional layers contains some local and spatial information. It happens that the higher the layers are, the more discriminative power they have. The features extracted from the higher convolutional layers could be beneficial along with those extracted from the top fully-connected layers.

Specifically, the output of Pool4 is a matrix of $48 \times 5 \times 5$, which is converted to a 1200D feature vector. Table 5 compares these three layers. The face recognition performance of Pool4 is much worse than the other two. The performance of Softmax is slightly higher than that of FC because i) Softmax layer is deeper than FC, and can extract more abstract and semantic information; and ii) The dimensionality of Softmax is much higher than FC, leading to a stronger representation capacity. Note that using softmax layer as a representation has president in the related idea of prototype similarity representations [15].

Table 5. Impact of layer choice on face recognition rate.

Layer	Accuracy	Feature dimension
Pool4	0.7203 ± 0.0041	1200
FC	0.7882 ± 0.0037	160
Softmax	0.7937 ± 0.0044	4000

3.4. Network Fusion

The work DeepID [24] and its variants [21, 25] exploit fusion of multiple networks. Specifically, the images of different facial regions and scales are separately fed into networks that have the same architecture. The features learned from different networks are concatenated to form a powerful face representation, which implicitly captures the spatial information of facial parts. The size of these images can be different as shown in Table 1. In [24], 120 networks are trained separately for this fusion. However, it is not very clear how significant this fusion to the performance. To clarify this issue, we implement network fusion.

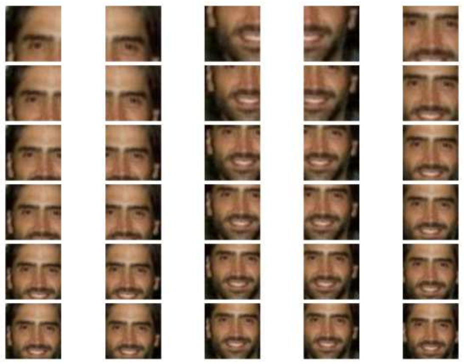

Figure 2. Sample crops in LFW. Rows correspond to 5 regions from 4 corners and center; Columns correspond to 6 scales.

3.4.1 Settings

The landmark AlextNet [14] model is trained using image crops from four corners and the center. Motivated by AlexNet, we extract $d \times d$ crops from these five regions and then upsample them to the original image size 58×58. The crops have 6 different scales: $d = floor(58 \times \{0.3, 0.4, 0.5, 0.6, 0.7, 0.8\})$, where $floor$ is the operator to get the integer part. Therefore we obtain 30 local patches with size of 58×58 from one original image. Figure 2 illustrates these 30 crops.

Note that it is an open question which regions are the most discriminative. Apart from choosing regions from four corners and the center, semantic regions are chosen for training networks by [17] and [24]. Specifically, facial landmarks are first detected, then the regions centered at those landmarks are chosen. These regions can be either rectangles or squares. However, this method depends on many accurately detected landmarks. It is beyond the scope of this work to investigate how to choose the most discriminative regions and how many such regions are most suitable for the task of face recognition. For simplicity, therefore, we choose the mentioned 30 regions.

To evaluate the performance of network fusion, we separately train 30 different networks using these crops. Two fusion strategies are applied to these 30 networks: 1) concatenating all the features of 30 networks (C-Fusion) and 2) averaging the feature values over 30 networks (A-Fusion).

3.4.2 Results

In this section, we evaluate the effectiveness of network fusion in Figs 3, 4, 5 and 6. Three single layers (Pool4, FC and Softmax) and one combination of FC and Softmax (FC+Softmax), which is a fusion of these 2 layers, are evaluated. The 30 crops are sorted in descending or-

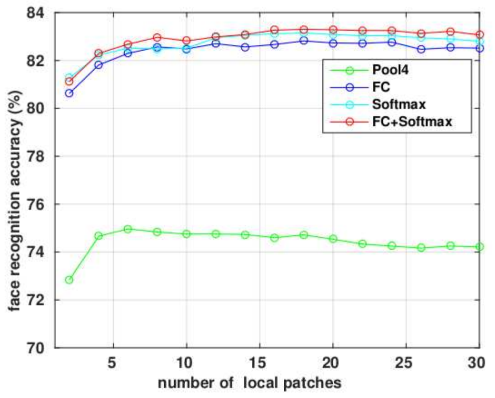

Figure 3. The performance of C-Fusion

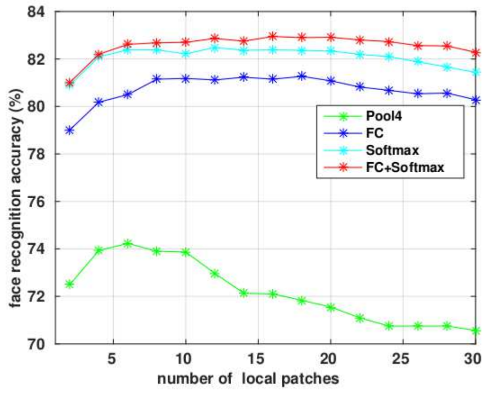

Figure 4. The performance of A-Fusion

der according to face recognition accuracies. The x axis of Figs 3, 4 and 5 represents the number of sorted crops. In other words, the number x' in x axis means the x' most discriminative patches are used for C-Fusion or A-Fusion.

First, we investigate whether the features of different layers complement each other. The features from two best layers, FC and Softmax evaluated in Table 5, are fused with C-Fusion and A-Fusion in Fig. 3 and Fig. 4 respectively. Softmax+FC works consistently better than single Softmax or FC in the case of both C- and A-Fusion, meaning that the features from Softmax are complementary to the most widely used FC features. To our knowledge, it is the first investigation on the combination of CNN features from different layers in the field of face recognition.

Second, we analyse how network fusion performance is affected by the number of local patches. For C-Fusion, Fig. 3 shows that face recognition accuracy increases with the number of patches approximately in the region $x' \in (1, 16)$, confirming the effectiveness of network fusion. However, in the region $x' > 16$, performance decreases slightly because the less discriminative patches degrade fusion performance. Similar results with A-Fusion can also be observed in Fig. 4 for the same reasons. It means that increasing the number of patches cannot uniformly improve results, as less discriminative patches eventually degrade performance. This conclusion is reflected by [21] which chooses 25 most discriminative patches out of 400 ones, ignoring those less discriminative ones, though the performance over the number of regions is not detailed in [21].

After presenting the effectiveness of C- and A-Fusion, we compare these two fusion strategies in Figure 5. The y axis is the accuracy difference obtained by subtracting the A-Fusion from C-Fusion accuracy. It is clear that all the differences are positive, meaning that C-Fusion consistently outperforms A-Fusion. However, the feature dimensionality of A-Fusion is much less than C-Fusion. For example, given 30 networks, the dimensionality of Softmax using C-Fusion is 30×1200, in comparison with 1200 using A-Fusion. Thus, in real applications, the choice of fusion methods depends on both accuracy and storage space requirements.

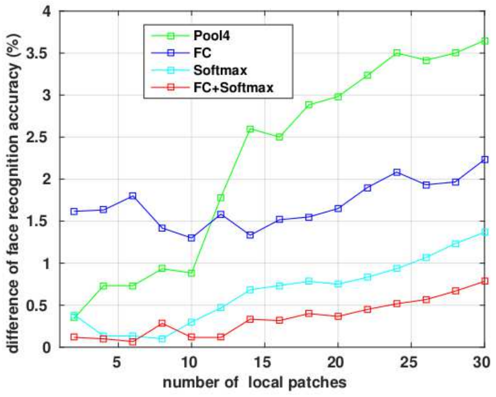

Figure 5. The accuracy margin of C-Fusion over A-Fusion.

Last but not least, we compare the performance with and without network fusion. The best network fusion results of C-Fusion are compared with those of a single network trained using the images of the whole face in Fig. 6. It is clear that network fusion greatly improves face recognition performance for all the three layers. The performance of a single network trained with a fixed image region is limited, and the blessing of dimensionality [7] is again observed when hand-crafted feature extraction is replaced by CNNs. Thus dense sampling remains useful in era of deep learning

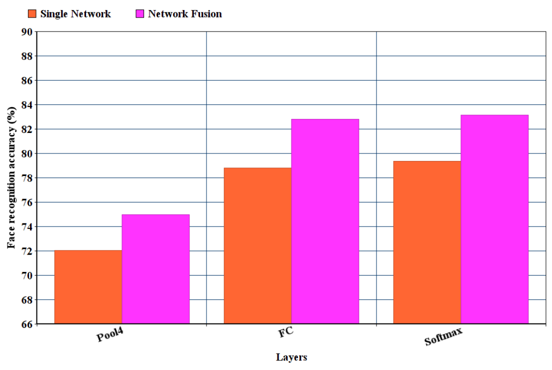

Figure 6. Comparison of single networks versus network fusion

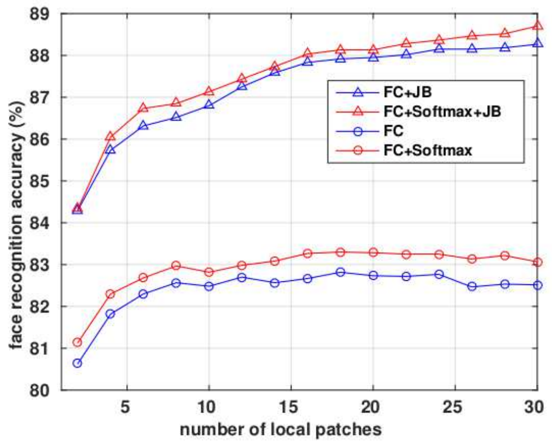

Figure 7. Face recognition accuracy using Joint Bayesian method

3.5. Metric Learning

Metric learning (MeL) is independent of facial features. Thus the features fed into MeL can be hand-crafted (SIFT, LBP, Fisher vector [19]) or learning-based (CNN [24, 31] and the learning-based descriptor [3]). In this work, the features we use are learned from the CNN framework and Joint Bayesian method (JB) is used for metric learning.

To evaluate the performance of JB, we use the features from 1) the FC layer, which is the most widely used and 2) FC+Softmax layer, which is used to validate whether these features complement each other based on the learned metric. C-Fusion is used for FC features due to higher accuracy and acceptable dimensionality, and A-Fusion is used for Softmax features to reduce the dimensionality.

Figure 7 illustrates the improvement gained by applying JB to the FC and FC+Softmax layers. First, the performance of FC and FC+Softmax using JB (FC+JB and FC+Softmax+JB in Fig. 7) is significantly better than that without JB, showing the importance of metric learning. Second, JB consistently improves face recognition rate with the increasing number of regions. This is different from to the result without using JB (FC and FC+Softmax in Fig. 7), the performance of which eventually decreases when the less discriminative regions (regions beyond about16) are used. It means JB is more robust and can extract useful information even from less discriminative regions.

3.6. Comparison with State of the Art

We finally compare our full method (FC+Softmax+JB with 30 regions) with non-commercial state of the art methods. The results in Table 6 show that our method is better than [27, 8, 16] but worse than [7, 19, 23]. However, [7] needs to detect many accurate facial landmarks to assist feature extraction, we do not; Compared with the fisher vector face [19], the feature dimensionality of our model is much lower. [23] generates a large number of new pairs to train the model, while we do not.

Table 6. Comparison with state-of-the-art methods on LFW under 'unrestricted, label-free outside data'

Method	Accuracy
LBP multishot [27]	0.8517 ± 0.0061
LDML-MkNN [8]	0.8750 ± 0.0040
LBP+PLDA [16]	0.8733 ± 0.0055
High-dim LBP [7]	0.9318 ± 0.0107
Fisher vector faces [19]	0.9303 ± 0.0105
ConvNet+RBM [23]	0.9175 ± 0.0048
Network fusion +JB	$\mathbf{0.8870 \pm 0.0063}$

4. Conclusions

We presented a rigorous empirical evaluation of CNN-based face recognition systems. Specifically, we quantitatively evaluate the impact of different architectures and implementation choices of CNNs on face recognition performances on common ground. We have shown that network fusion can significantly improve the face recognition performance because different networks capture the information from different regions and scales to form a powerful face representation. In addition, metric learning such as the Joint Bayesian method further improves face recognition greatly. Last, we observed that fusion of features from different CNN layers can boost face recognition performance.

Acknowledgements This work is supported by the European Union's Horizon 2020 research and innovation program under grant No 640891, EPSRC/dstl project 'Signal processing in a networked battlespace' under contract EP/K014307/1. We also gratefully acknowledge NVIDIA Corporation for the donation of the GPUs for this research.

References

[1] T. Ahonen, A. Hadid, and M. Pietikäinen. Face recognition with local binary patterns. In *Computer vision-eccv 2004*, pages 469–481. Springer, 2004.

[2] T. Ahonen, E. Rahtu, V. Ojansivu, and J. Heikkila. Recognition of blurred faces using local phase quantization. In *International Conference on Pattern Recognition*, 2008.

[3] Z. Cao, Q. Yin, X. Tang, and J. Sun. Face recognition with learning-based descriptor. In *Computer Vision and Pattern Recognition*, pages 2707–2714, 2010.

[4] C. H. Chan, M. A. Tahir, J. Kittler, and M. Pietikainen. Multiscale local phase quantization for robust component-based face recognition using kernel fusion of multiple descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(5):1164–1177, 2013.

[5] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.

[6] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun. Bayesian face revisited: A joint formulation. In *Computer Vision–ECCV 2012*, pages 566–579. Springer, 2012.

[7] D. Chen, X. Cao, F. Wen, and J. Sun. Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3025–3032. IEEE, 2013.

[8] M. Guillaumin, J. Verbeek, and C. Schmid. Is that you? metric learning approaches for face identification. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 498–505. IEEE, 2009.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *arXiv preprint arXiv:1502.01852*, 2015.

[10] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[11] G. Huang, M. Mattar, H. Lee, and E. G. Learned-Miller. Learning to align from scratch. In *Advances in Neural Information Processing Systems*, pages 764–772, 2012.

[12] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[13] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[15] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and Simile Classifiers for Face Verification. In *ICCV*, Oct 2009.

[16] P. Li, Y. Fu, U. Mohammed, J. H. Elder, and S. J. Prince. Probabilistic models for inference about identity. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(1):144–157, 2012.

[17] J. Liu, Y. Deng, and C. Huang. Targeting ultimate accuracy: Face recognition via deep embedding. *arXiv preprint arXiv:1506.07310*, 2015.

[18] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. *arXiv preprint arXiv:1503.03832*, 2015.

[19] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman. Fisher Vector Faces in the Wild. In *British Machine Vision Conference*, 2013.

[20] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[21] Y. Sun, Y. Chen, X. Wang, and X. Tang. Deep learning face representation by joint identification-verification. In *Advances in Neural Information Processing Systems*, pages 1988–1996, 2014.

[22] Y. Sun, D. Liang, X. Wang, and X. Tang. Deepid3: Face recognition with very deep neural networks. *arXiv preprint arXiv:1502.00873*, 2015.

[23] Y. Sun, X. Wang, and X. Tang. Hybrid deep learning for face verification. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1489–1496. IEEE, 2013.

[24] Y. Sun, X. Wang, and X. Tang. Deep learning face representation from predicting 10,000 classes. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1891–1898. IEEE, 2014.

[25] Y. Sun, X. Wang, and X. Tang. Deeply learned face representations are sparse, selective, and robust. *arXiv preprint arXiv:1412.1265*, 2014.

[26] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014.

[27] Y. Taigman, L. Wolf, T. Hassner, et al. Multiple one-shots for utilizing class label information. In *BMVC*, 2009.

[28] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1701–1708. IEEE, 2014.

[29] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *Computer Vision and Pattern Recognition*, pages 586–591, 1991.

[30] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for matlab. *CoRR*, abs/1412.4564, 2014.

[31] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014.

[32] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014*, pages 818–833. Springer, 2014.

[33] E. Zhou, Z. Cao, and Q. Yin. Naive-deep face recognition: Touching the limit of lfw benchmark or not? *arXiv preprint arXiv:1501.04690*, 2015.