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

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, dyl}@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. The existing works 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. To make our work reproducible, source code and models will be made publicly available.

1. Introduction

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, the hand-crafted features such as Local Binary Patterns (LBP) [1] and Local Phase Quantisation (LPQ) [2] 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 network (CNN) rebranded as ‘deep learning’ has achieved very impressive results on FRUE. Unlike the traditional hand-crafted features, the CNN learning-based features are more robust to complex intra-personal variations. More 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 [29] [22] [19]. The success of the latest CNNs on FRUE and more general object recognition task [14] [18] [13] stems from the following facts: (1) much larger labeled training sets are available; (2) GPU implementations greatly reduce the time of training a large CNN; (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 to adapt to 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 [21] [19] [22] [29] [25] are rather different from those for object recognition [14] [18] [23] [9]. For the task of face
recognition, it is important to make a systematic evaluation of the effect of different CNN design and implementation choices. In addition, those published CNNs [21, 29, 25, 28] are trained in 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 unfairness, the comparison of different CNNs should be conducted on a common ground.

To clarify the contributions of different components of CNN-based face recognition systems, in this paper, a systematic evaluation is conducted. 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. In addition, we evaluate the impact of multiple network fusion introduced by [21].
- Various implementation choices, 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.
- Finally, source code for our CNN architectures and trained networks will be made publicly available (the training data is already public). This provides an extremely competitive baseline for face recognition to the community. To our knowledge, we are the first to publish fully reproducible CNNs for face recognition.

2. 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 [25]. 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 [25] that one pooling layer is a good balance between local transformation robustness and preserving texture details. DeepFace is trained on the largest face database to-date which contains four million facial images of 4,000 subjects. Another contribution of [25] 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, [25] proposes a 3D alignment method using an affine camera model.

In [21], 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 a collection of small CNNs (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 images) × 60 (crops) × 160 (feature length of one network) = 19,200. One small 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 [19], 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+ [22] 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 [22], it is also discovered that DeepID2+ has three interesting properties: being sparse, selective and robust.

The work [28] 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 very deep architectures of [18, 23], WebFace trains a much deeper CNN than those [21, 19, 22, 25] 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 compares three typical CNNs (DeepFace [25], DeepID [21], WebFace [28]). 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.
### 3. Methodology

LFW is the de facto benchmark database for FRUE. Most existing CNNs [25,21,19,22] 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 [25,21,19,22] since our training data is much less extensive. We introduce three architectures adapting to our training set in subsection 3.2. To further improve the discrimination of CNN-learned features, metric learning method is usually used. One metric learning method, Joint Bayesian model [6], is detailed in subsection 3.2.

#### 3.1. 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 that used by the state of the art methods [25,21,19,22,28]; therefore, smaller architectures are designed.

We propose three CNN architectures adapting to the size of training data in 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] does not improve the performance of our CNNs, therefore, it is not applied to our networks. Following [21,28], softmax function is used in the last layer for predicting a single class 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.

<table>
<thead>
<tr>
<th>Input Image</th>
<th>Architecture</th>
<th>No. of para.</th>
<th>Feature Length</th>
<th>Training set</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DeepFace</strong> [25]</td>
<td>C1:32×11×11; M2; C3:16×9×9, L4:16×9×9, L5:16×7×7, L6:16×5×5, F7, F8</td>
<td>120M+</td>
<td>No</td>
<td>4096</td>
</tr>
<tr>
<td><strong>DeepID</strong> [21]</td>
<td>C1:20×4×4, M2, C3:40×3×3, M4, C5:60×3×3, M6, C7:80×2×2, M8, M9</td>
<td>101M+</td>
<td>Yes</td>
<td>19200</td>
</tr>
</tbody>
</table>

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 respectively. These capital letters are followed by the indices of CNN layers.
3 The number of filters and filter size are denoted as ‘num × size × size’.
3.2. Metric Learning

Metric Learning (MeL), which aims to find a new metric to make two classes more separable, is often used for face verification. MeL is independent of the feature extraction process and any feature (hand-crafted and learning-based) can be fed into a MeL method. Joint Bayesian (JB) model is a well-known MeL method and it is the most widely used MeL method which is applied to the features learned by CNNs [21, 19, 28].

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:

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

(1)

where $x_1$ and $x_2$ are features of one face pair. It is assumed that $P(x_1, x_2 | H_I)$ and $P(x_1, x_2 | H_E)$ 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 $\mu$ and intra-personal variations $\epsilon$):

$$x = \mu + \epsilon$$

(2)

$\mu$ and $\epsilon$ follow two Gaussian distributions $N(0, S_\mu)$ and $N(0, S_\epsilon)$, respectively. $S_\mu$ and $S_\epsilon$ 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:

$$\text{cov}(x_1, x_2) = \text{cov}(\mu_1, \mu_2) + \text{cov}(\epsilon_1, \epsilon_2)$$

(3)

Then $S_I$ and $S_E$ can be derived as:

$$S_I = \begin{vmatrix} S_\mu & S_\epsilon \\ S_\epsilon & S_\mu + S_\epsilon \end{vmatrix}$$

(4)

and

$$S_E = \begin{vmatrix} S_\mu + S_\epsilon & 0 \\ 0 & S_\mu + S_\epsilon \end{vmatrix}$$

(5)

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

4. Evaluation

LFW contains 5,749 subjects and 13,233 images and the training and test sets are defined in [12]. For evaluation, LFW is divided into 10 predefined splits for 10-fold cross validation. Each time nine of them are used for model training and the other one (600 image pairs) for testing. 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 x 58 based on the coordinates of two eye centers. Some sample crops are visualised in Fig. 1. 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 [27] is used to train our CNNs. In this section, different components of our CNN-based face recognition system are evaluated and analysed.

Architectures Choosing a ‘good’ architecture is crucial for CNN training. Overlarge or extremely small networks relative to the training data can lead to overfitting or underfitting, in the case of which the network does not converge at all during training. In this comparison, the RGB colour images are fed into CNNs and feature distance is measured by cosine distance. The performances of the three architectures are compared in Table 3. CNN-M achieves the best face recognition performance, indicating that the CNN-M generalises best among these three architectures using only LFW data. From this point, all the other 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. Comparisons of Our Three Architectures

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN-S</td>
<td>0.7828±0.0046</td>
</tr>
<tr>
<td>CNN-M</td>
<td>0.7882±0.0037</td>
</tr>
<tr>
<td>CNN-L</td>
<td>0.7807±0.0035</td>
</tr>
</tbody>
</table>

Feature Distance The existing research offers little discussion about the distance measurement for CNN-learned features. In particular, it is interesting to know what is the
Table 4. Distance Comparison

<table>
<thead>
<tr>
<th>Distance</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>euclidean</td>
<td>0.6898±0.0092</td>
</tr>
<tr>
<td>city block</td>
<td>0.6892±0.0088</td>
</tr>
<tr>
<td>chebychev</td>
<td>0.6692±0.0088</td>
</tr>
<tr>
<td>cosine</td>
<td>0.7882±0.0037</td>
</tr>
<tr>
<td>correlation</td>
<td>0.7882±0.0040</td>
</tr>
<tr>
<td>spearman</td>
<td>0.7878±0.0031</td>
</tr>
</tbody>
</table>

best distance measure for face recognition. Table 4 compares the impact of six distance measures on face recognition accuracy. Cosine and correlation achieve the best recognition rates, however, the standard deviation of cosine is smaller than that of correlation. Therefore, cosine distance is the best among these distances.

**Grey vs Colour** In [28] and [25], CNNs are trained using grey-level and RGB colour images, respectively. In comparison, both grey and colour images are used in [21]. We quantitatively compare the impact of these two image types on face recognition. Their comparative evaluation yields face recognition accuracies using grey and colour images of 0.7830±0.0077 and 0.7882±0.0118, respectively. The performances using grey and colour images are very close to each other. Although colour images contain more information, they do not deliver a significant improvement.

**Data Augmentation** Flip, mirroring images horizontally producing two samples from each, is a commonly used data augmentation technique for face recognition. Both original and mirrored images are used for training in all our evaluations. However, little discussion in the existing work was made to analyse 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, two fusion strategies (feature and score fusion) are implemented in this work. 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 5 compares the three scenarios: no flip during the test, feature and score fusions. As is shown in Table 5, mirroring images does improve the face recognition performance. In addition, feature fusion works slightly better than score fusion, however, the improvements are not statistically significant.

Table 5. Comparison of Data Augmentation during Test

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>no flip on test set</td>
<td>0.7882 ± 0.0037</td>
</tr>
<tr>
<td>feature fusion</td>
<td>0.7895 ± 0.0036</td>
</tr>
<tr>
<td>score fusion</td>
<td>0.7893 ± 0.0035</td>
</tr>
</tbody>
</table>

Figure 2. The impact of feature dimensionality in PCA space on face recognition rate

**Learned Feature Analysis** It is interesting to investigate the properties of CNN-learned face representations. First, we discuss feature normalisation, which standardises the range of features and is generally performed during the data preprocessing step. For example, to implement eigenface [26], the features (pixel values) are usually normalised via Eq. (6) before training a PCA space.

\[
\hat{x} = \frac{x - \mu_x}{\sigma_x}
\]

where \(x \in R\) and \(\hat{x} \in R\) are original and normalised feature vectors, respectively. \(\mu_x\) and \(\sigma_x\) are the mean and standard deviation of \(x\). Motivated by this, our CNN features are normalised by Eq. (6) before computing cosine distance. The accuracies with and without normalisation are 0.7927±0.0126 and 0.7882±0.0118, respectively. Thus normalisation is effective to improve recognition rate.

Second, we perform dimensionality reduction on the learned 160D features using PCA. As shown in Figure 2, only 16 dimensions of the PCA feature space can achieve 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 storage space and computation, which is crucial for large scale applications or mobile devices such as smartphone.

**Network Fusion** The work DeepID [21] and its variants [19, 22] apply the fusion of multiple networks. Specifically, the images of different facial regions and scales are separately fed into the networks that have the same architecture. The features learned from different networks are concatenated to 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 [21], 120 networks are trained separately for this fusion. However, it is not very clear how greatly this fusion improves the face recognition performance. To clarify this issue, we implement the network fusion.

We extract $d \times d$ crops from four corners and center and then upsample them to the original image size $58 \times 58$. The crops have 6 different scales: $d = \text{floor}(58 \times \{0.3, 0.4, 0.5, 0.6, 0.7, 0.8\})$, where $\text{floor}$ is the operator to get the integer part. Therefore we obtain 30 local patches with size of $58 \times 58$ from one original image. Figure 3 shows these 30 crops. To evaluate the performance of network fusion, we separately train 30 different networks using these crops. Then one face image can be represented by concatenating the features learned from different networks. Table 6 compares the performance of single network and network fusion. Note that we choose 16 best networks of 30 ones for the fusion. It is clear that network fusion works much better than a single network. Specifically, the fusion of 16 best networks improves the face recognition accuracy of single network by 4.51%. Clearly, the face representation of network fusion is actually the fusion of features of different facial components and scales. Similar ideas have widely been used to improve the facial representation capacity of hand-crafted features such as multi-scale local binary pattern [16], multi-scale local phase quantisation [4] and high-dimensional local features [7].

**Metric Learning** For metric learning, the features of the fusion of best 16 networks are used. The feature dimensionality $(2560 = 160 \times 16)$ is reduced to 320 via PCA before they are fed into JB. Figure 4 compares the face recognition accuracies with and without JB in each split of LFW database. JB consistently and significantly improves the face recognition rates, showing the importance of metric learning.

Table 7 compares our method with non-commercial state-of-the-art methods. The performance of our method is slightly better than [24, 8, 15] but worse than [7, 17, 20]. However, the feature dimensionality of [7, 17] is much higher than ours. In [20], a large number of new pairs are generated in addition to those provided by LFW to train the model, while we do not generate new pairs.

**Table 6. Comparison of Network Fusion**

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>single network</td>
<td>0.7882 ± 0.0037</td>
</tr>
<tr>
<td>network fusion</td>
<td>0.8333 ± 0.0042</td>
</tr>
</tbody>
</table>

**Table 7. Comparison with state-of-the-art methods on LFW under ‘unrestricted, label-free outside data’**

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>LBP multishot [24]</td>
<td>0.8517 ± 0.0061</td>
</tr>
<tr>
<td>LDML-MkNN [8]</td>
<td>0.8750 ± 0.0040</td>
</tr>
<tr>
<td>LBP+PLDA [15]</td>
<td>0.8733 ± 0.0055</td>
</tr>
<tr>
<td>high-dim LBP [7]</td>
<td>0.9318 ± 0.0107</td>
</tr>
<tr>
<td>Fisher vector faces [12]</td>
<td>0.9303 ± 0.0105</td>
</tr>
<tr>
<td>ConvNet+RBM [20]</td>
<td>0.9175 ± 0.0048</td>
</tr>
<tr>
<td>Network fusion +JB</td>
<td>0.8763 ± 0.0064</td>
</tr>
</tbody>
</table>

**5. Conclusions**

Recently, convolutional neural networks have attracted a lot of attention in the field of face recognition. In this work, we present 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 perfor-
Acknowledgements  This work is supported by the European Union’s Horizon 2020 research and innovation program under grant agreement No 640891. EPSRC/dstl project ‘Signal processing in a networked battlespace’ under contract EP/K014307/1. EPSRC Programme Grant ‘S3A: Future Spatial Audio for Immersive Listener Experiences at Home’ under contract EP/L000539, and the European Union project BEAT. We also gratefully acknowledge the support of NVIDIA Corporation for the donation of the GPUs used for this research.

References


