DeepFace: Closing the Gap to Human-Level Performance in Face Verification

Yaniv Taigman Ming Yang Marc'Aurelio Ranzato Lior Wolf

Facebook AI Research
Menlo Park, CA, USA
{yaniv, mingyang, ranzato}@fb.com

Tel Aviv University
Tel Aviv, Israel
wolf@cs.tau.ac.il

Abstract

*In modern face recognition, the conventional pipeline consists of four stages: detect ⇒ align ⇒ represent ⇒ classify. We revisit both the alignment step and the representation step by employing explicit 3D face modeling in order to apply a piecewise affine transformation, and derive a face representation from a nine-layer deep neural network. This deep network involves more than 120 million parameters using several locally connected layers without weight sharing, rather than the standard convolutional layers. Thus we trained it on the largest facial dataset to-date, an identity labeled dataset of four million facial images belonging to more than 4,000 identities. The learned representations coupling the accurate model-based alignment with the large facial database generalize remarkably well to faces in unconstrained environments, even with a simple classifier. Our method reaches an accuracy of 97.35% on the Labeled Faces in the Wild (*LFW*) dataset, reducing the error of the current state of the art by more than 27%, closely approaching human-level performance.*

1. Introduction

Face recognition in unconstrained images is at the forefront of the algorithmic perception revolution. The social and cultural implications of face recognition technologies are far reaching, yet the current performance gap in this domain between machines and the human visual system serves as a buffer from having to deal with these implications.

We present a system (*DeepFace*) that has closed the majority of the remaining gap in the most popular benchmark in unconstrained face recognition, and is now at the brink of human level accuracy. It is trained on a large dataset of faces acquired from a population vastly different than the one used to construct the evaluation benchmarks, and it is able to outperform existing systems with only very minimal adaptation. Moreover, the system produces an extremely compact face representation, in sheer contrast to the shift toward tens of thousands of appearance features in other recent systems [5, 7, 2].

The proposed system differs from the majority of contributions in the field in that it uses the deep learning (DL) framework [3, 21] in lieu of well engineered features. DL is especially suitable for dealing with large training sets, with many recent successes in diverse domains such as vision, speech and language modeling. Specifically with faces, the success of the learned net in capturing facial appearance in a robust manner is highly dependent on a very rapid 3D alignment step. The network architecture is based on the assumption that once the alignment is completed, the location of each facial region is fixed at the pixel level. It is therefore possible to learn from the raw pixel RGB values, without any need to apply several layers of convolutions as is done in many other networks [19, 21].

In summary, we make the following contributions : (i) The development of an effective deep neural net (DNN) architecture and learning method that leverage a very large labeled dataset of faces in order to obtain a face representation that generalizes well to other datasets; (ii) An effective facial alignment system based on explicit 3D modeling of faces; and (iii) Advance the state of the art significantly in (1) the Labeled Faces in the Wild benchmark (*LFW*) [18], reaching near human-performance; and (2) the YouTube Faces dataset (*YTF*) [30], decreasing the error rate there by more than 50%.

1.1. Related Work

Big data and deep learning In recent years, a large number of photos have been crawled by search engines, and uploaded to social networks, which include a variety of unconstrained material, such as objects, faces and scenes.

This large volume of data and the increase in computational resources have enabled the use of more powerful statistical models. These models have drastically improved the robustness of vision systems to several important variations, such as non-rigid deformations, clutter, occlusion and illumination, all problems that are at the core of many computer vision applications. While conventional machine

learning methods such as Support Vector Machines, Principal Component Analysis and Linear Discriminant Analysis, have limited capacity to leverage large volumes of data, deep neural networks have shown better scaling properties.

Recently, there has been a surge of interest in neural networks [19, 21]. In particular, deep and large networks have exhibited impressive results once: (1) they have been applied to large amounts of training data and (2) scalable computation resources such as thousands of CPU cores [11] and/or GPU's [19] have become available. Most notably, Krizhevsky et al. [19] showed that very large and deep convolutional networks [21] trained by standard back-propagation [25] can achieve excellent recognition accuracy when trained on a large dataset.

Face recognition state of the art Face recognition error rates have decreased over the last twenty years by three orders of magnitude [12] when recognizing frontal faces in still images taken in consistently controlled (constrained) environments. Many vendors deploy sophisticated systems for the application of border-control and smart biometric identification. However, these systems have shown to be sensitive to various factors, such as lighting, expression, occlusion and aging, that substantially deteriorate their performance in recognizing people in such unconstrained settings.

Most current face verification methods use hand-crafted features. Moreover, these features are often combined to improve performance, even in the earliest *LFW* contributions. The systems that currently lead the performance charts employ tens of thousands of image descriptors [5, 7, 2]. In contrast, our method is applied directly to RGB pixel values, producing a very compact yet sparse descriptor.

Deep neural nets have also been applied in the past to face detection [24], face alignment [27] and face verification [8, 16]. In the unconstrained domain, Huang et al. [16] used as input LBP features and they showed improvement when combining with traditional methods. In our method we use raw images as our underlying representation, and to emphasize the contribution of our work, we avoid combining our features with engineered descriptors. We also provide a new architecture, that pushes further the limit of what is achievable with these networks by incorporating 3D alignment, customizing the architecture for aligned inputs, scaling the network by almost two order of magnitudes and demonstrating a simple knowledge transfer method once the network has been trained on a very large labeled dataset.

Metric learning methods are used heavily in face verification, often coupled with task-specific objectives [26, 29, 6]. Currently, the most successful system that uses a large data set of labeled faces [5] employs a clever transfer learning technique which adapts a Joint Bayesian model [6] learned on a dataset containing 99,773 images from 2,995 different subjects, to the *LFW* image domain. Here, in order to demonstrate the effectiveness of the features, we keep the distance learning step trivial.

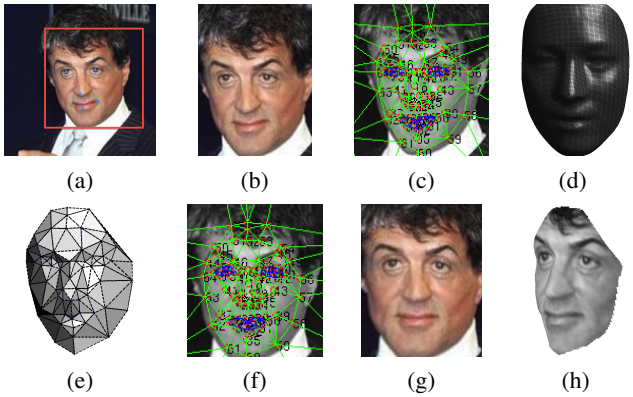

Figure 1. **Alignment pipeline.** (a) The detected face, with 6 initial fiducial points. (b) The induced 2D-aligned crop. (c) 67 fiducial points on the 2D-aligned crop with their corresponding Delaunay triangulation, we added triangles on the contour to avoid discontinuities. (d) The reference 3D shape transformed to the 2D-aligned crop image-plane. (e) Triangle visibility w.r.t. to the fitted 3D-2D camera; darker triangles are less visible. (f) The 67 fiducial points induced by the 3D model that are used to direct the piece-wise affine warpping. (g) The final frontalized crop. (h) A new view generated by the 3D model (not used in this paper).

2. Face Alignment

Existing aligned versions of several face databases (*e.g.* LFW-a [29]) help to improve recognition algorithms by providing a normalized input [26]. However, aligning faces in the unconstrained scenario is still considered a difficult problem that has to account for many factors such as pose (due to the non-planarity of the face) and non-rigid expressions, which are hard to decouple from the identity-bearing facial morphology. Recent methods have shown successful ways that compensate for these difficulties by using sophisticated alignment techniques. These methods can use one or more from the following: (1) employing an analytical 3D model of the face [28, 32, 14], (2) searching for similar fiducial-points configurations from an external dataset to infer from [4], and (3) unsupervised methods that find a similarity transformation for the pixels [17, 15].

While alignment is widely employed, no complete physically correct solution is currently present in the context of unconstrained face verification. 3D models have fallen out of favor in recent years, especially in unconstrained environments. However, since faces are 3D objects, done correctly, we believe that it is the right way. In this paper, we describe a system that includes analytical 3D modeling of the face based on fiducial points, that is used to warp a detected facial crop to a 3D frontal mode (*frontalization*).

Similar to much of the recent alignment literature, our alignment is based on using fiducial point detectors to direct the alignment process. We use a relatively simple fiducial

point detector, but apply it in several iterations to refine its output. At each iteration, fiducial points are extracted by a Support Vector Regressor (SVR) trained to predict point configurations from an image descriptor. Our image descriptor is based on LBP Histograms [1], but other features can also be considered. By transforming the image using the induced similarity matrix T to a new image, we can run the fiducial detector again on a new feature space and refine the localization.

2D Alignment We start our alignment process by detecting 6 fiducial points inside the detection crop, centered at the center of the eyes, tip of the nose and mouth locations as illustrated in Fig. 1(a). They are used to approximately scale, rotate and translate the image into six anchor locations by fitting $T_{2d}^{i} := (s_i, R_i, t_i)$ where: $x_{anchor}^{j} := s_i[R_i|t_i] * x_{source}^{j}$ for points $j = 1..6$ and iterate on the new warped image until there is no substantial change, eventually composing the final 2D similarity transformation: $T_{2d} := T_{2d}^{1} * ... * T_{2d}^{k}$. This aggregated transformation generates a 2D aligned crop, as shown in Fig. 1(b). This alignment method is similar to the one employed in LFW-a, which has been used frequently to boost recognition accuracy. However, similarity transformation fails to compensate for out-of-plane rotation, which is particularly important in unconstrained conditions.

3D Alignment In order to align faces undergoing out-of-plane rotations, we use a generic 3D shape model and register a 3D affine camera, which are used to warp the 2D-aligned crop to the image plane of the 3D shape. This generates the 3D-aligned version of the crop as illustrated in Fig. 1(g). This is achieved by localizing additional 67 fiducial points x_{2d} in the 2D-aligned crop (see Fig. 1(c)), using a second SVR. As a 3D generic shape model, we simply take the average of the 3D scans from the USF Human-ID database, which were post-processed to be represented as aligned vertices $v_i = (x_i, y_i, z_i)_{i=1}^{n}$. We manually place 67 anchor points on the 3D shape, and in this way achieve full correspondence between the 67 detected fiducial points and their 3D references. An affine 3D-to-2D camera P is then fitted using the generalized least squares solution to the linear system $x_{2d} = X_{3d}\vec{P}$ with a known covariance matrix Σ, that is, $\vec{P}$ that minimizes the following loss: $loss(\vec{P}) = r^T\Sigma^{-1}r$ where $r = (x_{2d} - X_{3d}\vec{P})$ is the residual vector and X_{3d} is a $(67 * 2) \times 8$ matrix composed by stacking the (2×8) matrices $[x_{3d}^{\top}(i), 1, \vec{0}; \vec{0}, x_{3d}^{\top}(i), 1]$, with $\vec{0}$ denoting a row vector of four zeros, for each reference fiducial point $x_{3d}(i)$. The affine camera P of size 2×4 is represented by the vector of 8 unknowns $\vec{P}$. The loss can be minimized using the Cholesky decomposition of Σ, that transforms the problem into ordinary least squares. Since, for example, detected points on the contour of the face tend to be more noisy, as their estimated location is largely influenced by the depth with respect to the camera angle, we use a $(67 * 2) \times (67 * 2)$ covariance matrix Σ given by the estimated covariances of the fiducial point errors.

Frontalization Since full perspective projections and non-rigid deformations are not modeled, the fitted camera P is only an approximation. In order to reduce the corruption of such important identity-bearing factors to the final warping, we add the corresponding residuals in r to the x-y components of each reference fiducial point x_{3d}, we denote this as $\widetilde{x_{3d}}$. Such a relaxation is plausible for the purpose of warping the 2D image with smaller distortions to the identity. Without it, faces would have been warped into the same shape in 3D, losing important discriminative factors. Finally, the frontalization is achieved by a piece-wise affine transformation T from x_{2d} (source) to $\widetilde{x_{3d}}$ (target), directed by the Delaunay triangulation derived from the 67 fiducial points[1]. Also, invisible triangles w.r.t. to camera P, can be replaced using image blending with their symmetrical counterparts.

3. Representation

In recent years, the computer vision literature has attracted many research efforts in descriptor engineering. Such descriptors when applied to face-recognition, mostly use the same operator to all locations in the facial image. Recently, as more data has become available, learning-based methods have started to outperform engineered features, because they can discover and optimize features for the specific task at hand [19]. Here, we learn a generic representation of facial images through a large deep network.

DNN Architecture and Training We train our DNN on a multi-class face recognition task, namely to classify the identity of a face image. The overall architecture is shown in Fig. 2. A 3D-aligned 3-channels (RGB) face image of size 152 by 152 pixels is given to a convolutional layer (C1) with 32 filters of size 11x11x3 (we denote this by 32x11x11x3@152x152). The resulting 32 feature maps are then fed to a max-pooling layer (M2) which takes the max over 3x3 spatial neighborhoods with a stride of 2, separately for each channel. This is followed by another convolutional layer (C3) that has 16 filters of size 9x9x16. The purpose of these three layers is to extract low-level features, like simple edges and texture. Max-pooling layers make the output of convolution networks more robust to local translations. When applied to aligned facial images, they make the network more robust to small registration errors. However, several levels of pooling would cause the network to lose information about the precise position of detailed facial structure and micro-textures. Hence, we apply max-pooling only to the first convolutional layer. We interpret these first layers as a front-end adaptive pre-processing stage. While they are responsible for most of the computation, they hold

[1] T_{2d} can be used here to avoid going through the 2D lossy warping.

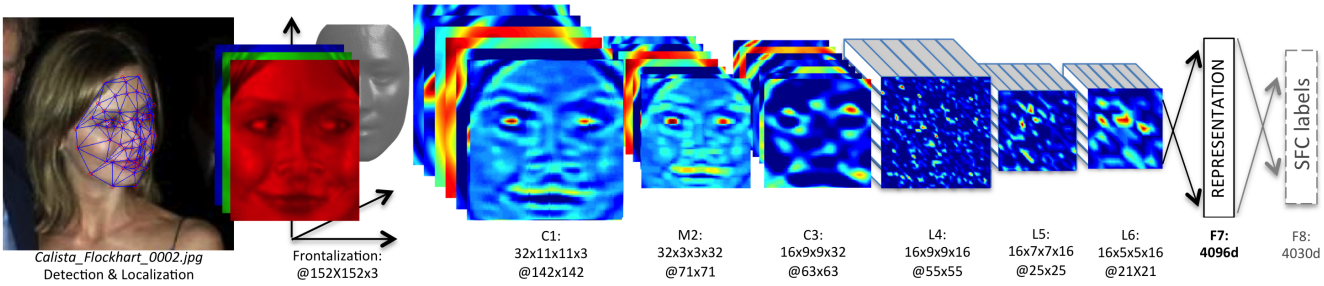

Figure 2. **Outline of the *DeepFace* architecture**. A front-end of a single convolution-pooling-convolution filtering on the rectified input, followed by three locally-connected layers and two fully-connected layers. Colors illustrate feature maps produced at each layer. The net includes more than 120 million parameters, where more than 95% come from the local and fully connected layers.

very few parameters. These layers merely expand the input into a set of simple local features.

The subsequent layers (L4, L5 and L6) are instead locally connected [13, 16], like a convolutional layer they apply a filter bank, but every location in the feature map learns a different set of filters. Since different regions of an aligned image have different local statistics, the spatial stationarity assumption of convolution cannot hold. For example, areas between the eyes and the eyebrows exhibit very different appearance and have much higher discrimination ability compared to areas between the nose and the mouth. In other words, we customize the architecture of the DNN by leveraging the fact that our input images are aligned. The use of local layers does not affect the computational burden of feature extraction, but does affect the number of parameters subject to training. Only because we have a large labeled dataset, we can afford three large locally connected layers. The use of locally connected layers (without weight sharing) can also be justified by the fact that each output unit of a locally connected layer is affected by a very large patch of the input. For instance, the output of L6 is influenced by a 74x74x3 patch at the input, and there is hardly any statistical sharing between such large patches in *aligned* faces.

Finally, the top two layers (F7 and F8) are fully connected: each output unit is connected to all inputs. These layers are able to capture correlations between features captured in distant parts of the face images, *e.g.*, position and shape of eyes and position and shape of mouth. The output of the first fully connected layer (F7) in the network will be used as our raw face representation feature vector throughout this paper. In terms of representation, this is in contrast to the existing LBP-based representations proposed in the literature, that normally pool very local descriptors (by computing histograms) and use this as input to a classifier.

The output of the last fully-connected layer is fed to a K-way softmax (where K is the number of classes) which produces a distribution over the class labels. If we denote by o_k the k-th output of the network on a given input, the probability assigned to the k-th class is the output of the softmax function: $p_k = \exp(o_k)/\sum_h \exp(o_h)$.

The goal of training is to maximize the probability of the correct class (face id). We achieve this by minimizing the cross-entropy loss for each training sample. If k is the index of the true label for a given input, the loss is: $L = -\log p_k$. The loss is minimized over the parameters by computing the gradient of L w.r.t. the parameters and by updating the parameters using stochastic gradient descent (SGD). The gradients are computed by standard back-propagation of the error [25, 21]. One interesting property of the features produced by this network is that they are very sparse. On average, 75% of the feature components in the topmost layers are exactly zero. This is mainly due to the use of the ReLU [10] activation function: $max(0, x)$. This soft-thresholding non-linearity is applied after every convolution, locally connected and fully connected layer (except the last one), making the whole cascade produce highly non-linear and sparse features. Sparsity is also encouraged by the use of a regularization method called dropout [19] which sets random feature components to 0 during training. We have applied dropout only to the first fully-connected layer. Due to the large training set, we did not observe significant overfitting during training[2].

Given an image I, the representation $G(I)$ is then computed using the described feed-forward network. Any feed-forward neural network with L layers, can be seen as a composition of functions g^l_ϕ. In our case, the representation is: $G(I) = g^{F_7}_\phi(g^{L_6}_\phi(...g^{C_1}_\phi(T(I, \theta_T))...))$ with the net's parameters $\phi = \{C_1, ..., F_7\}$ and $\theta_T = \{x_{2d}, \vec{P}, \vec{r}\}$ as described in Section 2.

Normaliaztion As a final stage we normalize the features to be between zero and one in order to reduce the sensitivity to illumination changes: Each component of the feature vector is divided by its largest value across the training set. This is then followed by L_2-normalization: $f(I) := \bar{G}(I)/||\bar{G}(I)||_2$ where $\bar{G}(I)_i = G(I)_i/\max(G_i, \epsilon)$ [3]. Since we employ ReLU activations, our system is not invariant to re-scaling of the image intensities. Without bi-

[2] See the **supplementary** material for more details.

[3] $\epsilon = 0.05$ in order to avoid division by a small number.

ases in the DNN, perfect equivariance would have been achieved.

4. Verification Metric

Verifying whether two input instances belong to the same class (identity) or not has been extensively researched in the domain of unconstrained face-recognition, with supervised methods showing a clear performance advantage over unsupervised ones. By training on the target-domain's training set, one is able to fine-tune a feature vector (or classifier) to perform better within the particular distribution of the dataset. For instance, *LFW* has about 75% males, celebrities that were photographed by mostly professional photographers. As demonstrated in [5], training and testing within different domain distributions hurt performance considerably and requires further tuning to the representation (or classifier) in order to improve their generalization and performance. However, fitting a model to a relatively small dataset reduces its generalization to other datasets. In this work, we aim at learning an unsupervised metric that generalizes well to several datasets. Our unsupervised similarity is simply the inner product between the two normalized feature vectors. We have also experimented with a supervised metric, the χ^2 similarity and the Siamese network.

4.1. Weighted χ^2 distance

The normalized *DeepFace* feature vector in our method contains several similarities to histogram-based features, such as LBP [1] : (1) It contains non-negative values, (2) it is very sparse, and (3) its values are between [0, 1]. Hence, similarly to [1], we use the weighted-χ^2 similarity: $\chi^2(f_1, f_2) = \sum_i w_i(f_1[i] - f_2[i])^2/(f_1[i] + f_2[i])$ where f_1 and f_2 are the *DeepFace* representations. The weight parameters are learned using a linear SVM, applied to vectors of the elements $(f_1[i] - f_2[i])^2/(f_1[i] + f_2[i])$.

4.2. Siamese network

We have also tested an end-to-end metric learning approach, known as Siamese network [8]: once learned, the face recognition network (without the top layer) is replicated twice (one for each input image) and the features are used to directly predict whether the two input images belong to the same person. This is accomplished by: a) taking the absolute difference between the features, followed by b) a top fully connected layer that maps into a single logistic unit (same/not same). The network has roughly the same number of parameters as the original one, since much of it is shared between the two replicas, but requires twice the computation. Notice that in order to prevent overfitting on the face verification task, we enable training for only the two topmost layers. The Siamese network's induced distance is: $d(f_1, f_2) = \sum_i \alpha_i |f_1[i] - f_2[i]|$, where α_i are trainable parameters. The parameters of the Siamese network are trained by standard cross entropy loss and back-propagation of the error.

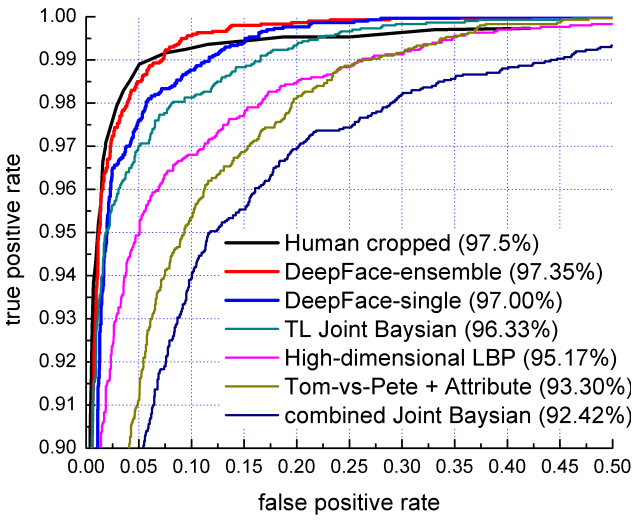

Figure 3. The ROC curves on the *LFW* dataset. Best viewed in color.

5. Experiments

We evaluate the proposed *DeepFace* system, by learning the face representation on a very large-scale labeled face dataset collected online. In this section, we first introduce the datasets used in the experiments, then present the detailed evaluation and comparison with the state-of-the-art, as well as some insights and findings about learning and transferring the deep face representations.

5.1. Datasets

The proposed face representation is learned from a large collection of photos from *Facebook*, referred to as the Social Face Classification (*SFC*) dataset. The representations are then applied to the Labeled Faces in the Wild database (*LFW*), which is the *de facto* benchmark dataset for face verification in unconstrained environments, and the YouTube Faces (*YTF*) dataset, which is modeled similarly to the *LFW* but focuses on video clips.

The *SFC* dataset includes 4.4 million labeled faces from 4,030 people each with 800 to 1200 faces, where the most recent 5% of face images of each identity are left out for testing. This is done according to the images' time-stamp in order to simulate continuous identification through aging. The large number of images per person provides a unique opportunity for learning the invariance needed for the core problem of face recognition. We have validated using several automatic methods, that the identities used for training do not intersect with any of the identities in the below-mentioned datasets, by checking their name labels.

The *LFW* dataset [18] consists of 13,323 web photos of 5,749 celebrities which are divided into 6,000 face pairs in 10 splits. Performance is measured by mean recognition accuracy using A) the *restricted* protocol, in which only same and not same labels are available in training; B) the *unrestricted* protocol, where additional training pairs are accessible in training; and C) an *unsupervised* setting in which no training whatsoever is performed on LFW images.

The *YTF* dataset [30] collects 3,425 YouTube videos of 1,595 subjects (a subset of the celebrities in the *LFW*). These videos are divided into 5,000 video pairs and 10 splits and used to evaluate the video-level face verification.

The face identities in *SFC* were labeled by humans, which typically incorporate about 3% errors. Social face photos have even larger variations in image quality, lighting, and expressions than the web images of celebrities in the *LFW* and *YTF* which were normally taken by professional photographers rather than smartphones[4].

5.2. Training on the *SFC*

We first train the deep neural network on the *SFC* as a multi-class classification problem using a GPU-based engine, implementing the standard back-propagation on feed-forward nets by stochastic gradient descent (SGD) with momentum (set to 0.9). Our mini-batch size is 128, and we have set an equal learning rate for all trainable layers to 0.01, which was manually decreased, each time by an order of magnitude once the validation error stopped decreasing, to a final rate of 0.0001. We initialized the weights in each layer from a zero-mean Gaussian distribution with $\sigma = 0.01$, and biases are set to 0.5. We trained the network for roughly 15 sweeps (epochs) over the whole data which took 3 days. As described in Sec. 3, the responses of the fully connected layer F7 are extracted to serve as the face representation.

We evaluated different design choices of DNN in terms of the classification error on 5% data of *SFC* as the test set. This validated the necessity of using a large-scale face dataset and a deep architecture. First, we vary the train/test dataset size by using a subset of the persons in the *SFC*. Subsets of sizes 1.5K, 3K and 4K persons (1.5M, 3.3M, and 4.4M faces, respectively) are used. Using the architecture in Fig. 2, we trained three networks, denoted by *DF-1.5K*, *DF-3.3K*, and *DF-4.4K*. Table 1 (left column) shows that the classification error grows only modestly from 7.0% on 1.5K persons to 7.2% when classifying 3K persons, which indicates that the capacity of the network can well accommodate the scale of 3M training images. The error rate rises to 8.7% for 4K persons with 4.4M images, showing the network scales comfortably to more persons. We've also varied the global number of samples in *SFC* to $10\%, 20\%, 50\%$, leaving the number of identities in place, denoted by *DF-10%*, *DF-20%*, *DF-50%* in the middle column of Table 1. We observed the test errors rise up to 20.7%, because of overfitting on the reduced training set. Since performance does not saturate at 4M images, this shows that the network would benefit from even larger datasets.

We also vary the depth of the networks by chopping off the C3 layer, the two local L4 and L5 layers, or all these 3 layers, referred respectively as *DF-sub1*, *DF-sub2*, and *DF-sub3*. For example, only four trainable layers remain in *DF-sub3* which is a considerably shallower structure compared to the 9 layers of the proposed network in Fig. 2. In training such networks with 4.4M faces, the classification errors stop decreasing after a few epochs and remains at a level higher than that of the deep network, as can be seen in Table 1 (right column). This verifies the necessity of network depth when training on a large face dataset.

5.3. Results on the *LFW* dataset

The vision community has made significant progress on face verification in unconstrained environments in recent years. The mean recognition accuracy on *LFW* [18] marches steadily towards the human performance of over 97.5% [20]. Given some very hard cases due to aging effects, large lighting and face pose variations in *LFW*, any improvement over the state-of-the-art is very remarkable and the system has to be composed by highly optimized modules. There is a strong diminishing return effect and any progress now requires a substantial effort to reduce the number of errors of state-of-the-art methods. *DeepFace* couples large feedforward-based models with fine 3D alignment. Regarding the importance of each component: 1) Without frontalization: when using only the 2D alignment, the obtained accuracy is "only" 94.3%. Without alignment at all, *i.e.*, using the center crop of face detection, the accuracy is 87.9% as parts of the facial region may fall out of the crop. 2) Without learning: when using frontalization only, and a naive LBP/SVM combination, the accuracy is 91.4% which is already notable given the simplicity of such a classifier.

All the *LFW* images are processed in the same pipeline that was used to train on the *SFC* dataset, denoted as *DeepFace-single*. To evaluate the discriminative capability of the face representation in isolation, we follow the **unsupervised** setting to directly compare the inner product of a pair of normalized features. Quite remarkably, this achieves a mean accuracy of 95.92% which is almost on par with the best performance to date, achieved by supervised transfer learning [5]. Next, we learn a kernel SVM (with C=1) on top of the χ^2-distance vector (Sec. 4.1) following the **restricted** protocol, *i.e.*, where only the 5,400 pair labels per split are available for the SVM training. This achieves an accuracy **97.00%**, reducing significantly the error of the state-of-the-art [7, 5], see Table 3.

[4] See the **supplementary** material for more details about *SFC*.

Network	Error	Network	Error	Network	Error
DF-1.5K	7.00%	*DF-10%*	20.7%	*DF-sub1*	11.2%
DF-3.3K	7.22%	*DF-20%*	15.1%	*DF-sub2*	12.6%
DF-4.4K	8.74%	*DF-50%*	10.9%	*DF-sub3*	13.5%

Table 1. Comparison of the classification errors on the *SFC w.r.t.* training dataset size and network depth. See Sec. 5.2 for details.

Network	Error (*SFC*)	Accuracy ± SE (*LFW*)
DeepFace-align2D	9.5%	0.9430 ±0.0043
DeepFace-gradient	8.9%	0.9582 ±0.0037
DeepFace-Siamese	NA	0.9617 ±0.0038

Table 2. The performance of various individual *DeepFace* networks and the Siamese network.

Ensembles of DNNs Next, we combine multiple networks trained by feeding different types of inputs to the DNN: 1) The network *DeepFace-single* described above based on 3D aligned RGB inputs; 2) The gray-level image plus image gradient magnitude and orientation; and 3) the 2D-aligned RGB images. We combine those distances using a non-linear SVM (with C=1) with a simple sum of power CPD-kernels: $K_{\text{Combined}} := K_{\text{single}} + K_{\text{gradient}} + K_{\text{align2d}}$, where $K(x, y) := -||x - y||_2$, and following the restricted protocol, achieve an accuracy **97.15%**.

The *unrestricted* protocol provides the operator with knowledge about the identities in the training sets, hence enabling the generation of many more training pairs to be added to the training set. We further experiment with training a Siamese Network (Sec. 4.2) to learn a verification metric by fine-tuning the Siamese's (shared) pre-trained feature extractor. Following this procedure, we have observed substantial overfitting to the training data. The training pairs generated using the *LFW* training data are redundant as they are generated out of roughly 9K photos, which are insufficient to reliably estimate more than 120M parameters. To address these issues, we have collected an additional dataset following the same procedure as with the *SFC*, containing an additional new 100K identities, each with only 30 samples to generate same and not-same pairs from. We then trained the Siamese Network on it followed by 2 training epochs on the LFW unrestricted training splits to correct for some of the data set dependent biases. The slightly-refined representation is handled similarly as before. Combining it into the above-mentioned ensemble, *i.e.*, K_{Combined} += K_{Siamese}, yields the accuracy **97.25%**, under the *unrestricted* protocol. By adding four additional *DeepFace-single* networks to the ensemble, each trained from scratch with different random seeds, *i.e.*, K_{Combined} += $\sum K_{\text{DeepFace-Single}}$, the obtained accuracy is **97.35%**. The performances of the individual networks, before combination, are presented in Table 2.

The comparisons with the recent state-of-the-art methods in terms of the mean accuracy and ROC curves are presented in Table 3 and Fig. 3, including human performance on the cropped faces. The proposed *DeepFace* method advances the state-of-the-art, closely approaching human performance in face verification.

Method	Accuracy ± SE	Protocol
Joint Bayesian [6]	0.9242 ±0.0108	restricted
Tom-vs-Pete [4]	0.9330 ±0.0128	restricted
High-dim LBP [7]	0.9517 ±0.0113	restricted
TL Joint Bayesian [5]	0.9633 ±0.0108	restricted
DeepFace-single	**0.9592** ±0.0029	unsupervised
DeepFace-single	**0.9700** ±0.0028	restricted
DeepFace-ensemble	**0.9715** ±0.0027	restricted
DeepFace-ensemble	**0.9735** ±0.0025	unrestricted
Human, cropped	0.9753	

Table 3. Comparison with the state-of-the-art on the *LFW* dataset.

Method	Accuracy (%)	AUC	EER
MBGS+SVM- [31]	78.9 ±1.9	86.9	21.2
APEM+FUSION [22]	79.1 ±1.5	86.6	21.4
STFRD+PMML [9]	79.5 ±2.5	88.6	19.9
VSOF+OSS [23]	79.7 ±1.8	89.4	20.0
DeepFace-single	**91.4** ±1.1	96.3	8.6

Table 4. Comparison with the state-of-the-art on the *YTF* dataset.

5.4. Results on the *YTF* dataset

We further validate *DeepFace* on the recent video-level face verification dataset. The image quality of YouTube video frames is generally worse than that of web photos, mainly due to motion blur or viewing distance. We employ the *DeepFace-single* representation directly by creating, for every pair of training videos, 50 pairs of frames, one from each video, and label these as same or not-same in accordance with the video training pair. Then a weighted χ^2 model is learned as in Sec. 4.1. Given a test-pair, we sample 100 random pairs of frames, one from each video, and use the mean value of the learned weighed similarity.

The comparison with recent methods is shown in Table 4 and Fig. 4. We report an accuracy of **91.4%** which reduces the error of the previous best methods by more than 50%. Note that there are about 100 wrong labels for video pairs, recently updated to the *YTF* webpage. After these are corrected, *DeepFace-single* actually reaches 92.5%. This experiment verifies again that the *DeepFace* method easily generalizes to a new target domain.

5.5. Computational efficiency

We have efficiently implemented a CPU-based feedforward operator, which exploits both the CPU's Single Instruction Multiple Data (SIMD) instructions and its cache by leveraging the locality of floating-point computations

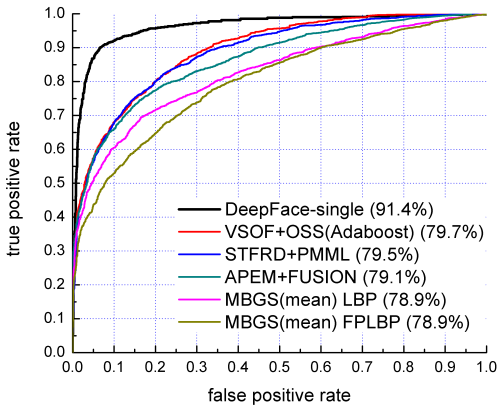

Figure 4. The ROC curves on the *YTF* dataset. Best viewed in color.

across the kernels and the image. Using a single core Intel 2.2GHz CPU, the operator takes 0.18 seconds to extract features from the raw input pixels. Efficient warping techniques were implemented for alignment; alignment alone takes about 0.05 seconds. Overall, the *DeepFace* runs at 0.33 seconds per image, accounting for image decoding, face detection and alignment, the feedforward network, and the final classification output.

6. Conclusion

An ideal face classifier would recognize faces in accuracy that is only matched by humans. The underlying face descriptor would need to be invariant to pose, illumination, expression, and image quality. It should also be general, in the sense that it could be applied to various populations with little modifications, if any at all. In addition, short descriptors are preferable, and if possible, sparse features. Certainly, rapid computation time is also a concern. We believe that this work, which departs from the recent trend of using more features and employing a more powerful metric learning technique, has addressed this challenge, closing the vast majority of this performance gap. Our work demonstrates that coupling a 3D model-based alignment with large capacity feedforward models can effectively learn from many examples to overcome the drawbacks and limitations of previous methods. The ability to present a marked improvement in face recognition, attests to the potential of such coupling to become significant in other vision domains as well.

References

[1] T. Ahonen, A. Hadid, and M. Pietikäinen. Face description with local binary patterns: Application to face recognition. *PAMI*, 2006. 3, 5

[2] O. Barkan, J. Weill, L. Wolf, and H. Aronowitz. Fast high dimensional vector multiplication face recognition. In *ICCV*, 2013. 1, 2

[3] Y. Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2009. 1

[4] T. Berg and P. N. Belhumeur. Tom-vs-pete classifiers and identity-preserving alignment for face verification. In *BMVC*, 2012. 2, 7

[5] X. Cao, D. Wipf, F. Wen, G. Duan, and J. Sun. A practical transfer learning algorithm for face verification. In *ICCV*, 2013. 1, 2, 5, 6, 7

[6] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun. Bayesian face revisited: A joint formulation. In *ECCV*, 2012. 2, 7

[7] D. Chen, X. Cao, F. Wen, and J. Sun. Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification. In *CVPR*, 2013. 1, 2, 6, 7

[8] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 2005. 2, 5

[9] Z. Cui, W. Li, D. Xu, S. Shan, and X. Chen. Fusing robust face region descriptors via multiple metric learning for face recognition in the wild. In *CVPR*, 2013. 7

[10] G. E. Dahl, T. N. Sainath, and G. E. Hinton. Improving deep neural networks for LVCSR using rectified linear units and dropout. In *ICASSP*, 2013. 4

[11] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, Q. Le, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Ng. Large scale distributed deep networks. In *NIPS*, 2012. 2

[12] P. J. P. et al. An introduction to the good, the bad, & the ugly face recognition challenge problem. In *FG*, 2011. 2

[13] K. Gregor and Y. LeCun. Emergence of complex-like cells in a temporal product network with local receptive fields. arXiv:1006.0448, 2010. 4

[14] T. Hassner. Viewing real-world faces in 3D. In *International Conference on Computer Vision (ICCV)*, Dec. 2013. 2

[15] G. B. Huang, V. Jain, and E. G. Learned-Miller. Unsupervised joint alignment of complex images. In *ICCV*, 2007. 2

[16] G. B. Huang, H. Lee, and E. Learned-Miller. Learning hierarchical representations for face verification with convolutional deep belief networks. In *CVPR*, 2012. 2, 4

[17] G. B. Huang, M. A. Mattar, H. Lee, and E. G. Learned-Miller. Learning to align from scratch. In *NIPS*, pages 773–781, 2012. 2

[18] 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. In *ECCV Workshop on Faces in Real-life Images*, 2008. 1, 6

[19] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In *ANIPS*, 2012. 1, 2, 3, 4

[20] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. In *ICCV*, 2009. 6

[21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient based learning applied to document recognition. *Proc. IEEE*, 1998. 1, 2, 4

[22] H. Li, G. Hua, Z. Lin, J. Brandt, and J. Yang. Probabilistic elastic matching for pose variant face verification. In *CVPR*, 2013. 7

[23] H. Mendez-Vazquez, Y. Martinez-Diaz, and Z. Chai. Volume structured ordinal features with background similarity measure for video face recognition. In *Int'l Conf. on Biometrics*, 2013. 7

[24] M. Osadchy, Y. LeCun, and M. Miller. Synergistic face detection and pose estimation with energy-based models. *JMLR*, 2007. 2

[25] D. Rumelhart, G. Hinton, and R. Williams. Learning representations by back-propagating errors. *Nature*, 1986. 2, 4

[26] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman. Fisher vector faces in the wild. In *BMVC*, 2013. 2

[27] Y. Sun, X. Wang, and X. Tang. Deep convolutional network cascade for facial point detection. In *CVPR*, 2013. 2

[28] Y. Taigman and L. Wolf. Leveraging billions of faces to overcome performance barriers in unconstrained face recognition. arXiv:1108.1122, 2011. 2

[29] Y. Taigman, L. Wolf, and T. Hassner. Multiple one-shots for utilizing class label information. In *BMVC*, 2009. 2

[30] L. Wolf, T. Hassner, and I. Maoz. Face recognition in unconstrained videos with matched background similarity. In *CVPR*, 2011. 1, 6

[31] L. Wolf and N. Levy. The SVM-minus similarity score for video face recognition. In *CVPR*, 2013. 7

[32] D. Yi, Z. Lei, and S. Z. Li. Towards pose robust face recognition. In *CVPR*, 2013. 2