Realtime Face Verification with Lightweight Convolutional Neural Networks

Nhan Dam[1], Vinh-Tiep Nguyen[2], Minh N. Do[3], Anh-Duc Duong[4], and Minh-Triet Tran[2(✉)]

[1] AI Lab, University of Science, VNUHCM, Ho Chi Minh, Vietnam
dntnhan@apcs.vn
[2] Faculty of IT, University of Science, VNUHCM, Ho Chi Minh, Vietnam
{nvtiep,tmtriet}@fit.hcmus.edu.vn
[3] Department of ECE, University of Illinois at Urbana-Champaign, Champaign, USA
minhdo@illinois.edu
[4] University of Information Technology, VNUHCM, Ho Chi Minh, Vietnam
ducda@uit.edu.vn

Abstract. Face verification is a promising method for user authentication. Besides existing methods with deep convolutional neural networks to handle millions of people using powerful computing systems, the authors aim to propose an alternative approach of a lightweight scheme of convolutional neural networks (CNN) for face verification in realtime. Our goal is to propose a simple yet efficient method for face verification that can be deployed on regular commodity computers for individuals or small-to-medium organizations without super-computing strength. The proposed scheme targets unconstrained face verification, a typical scenario in reality. Experimental results on original data of Labeled Faces in the Wild dataset show that our best CNN found through experiments with 10 hidden layers achieves the accuracy of $(82.58 \pm 1.30)\,\%$ while many other instances in the same scheme can also approximate this result. The current implementation of our method can run at 60 fps and 235 fps on a regular computer with CPU-only and GPU configurations respectively. This is suitable for deployment in various applications without special requirements of hardware devices.

Keywords: Unconstrained face verification · Convolutional neural network · Lightweight

1 Introduction

User authentication is one of the most important steps for various applications to ensure that only authorized users can access systems, and to provide appropriate information and services corresponding to the users. Face recognition or verification can be used as a natural yet efficient method for user authentication.

For face verification, we simply answer whether a pair of photos are of the same person or not. This is appropriate for applications in which we cannot

G. Bebis et al. (Eds.): ISVC 2015, Part II, LNCS 9475, pp. 420–430, 2015.
DOI: 10.1007/978-3-319-27863-6_39

know or do not need to know all persons in advance but only to match people in various scenarios from their faces, such as in parking system, video surveillance, etc. Face verification can also be used to recognize a user from a face photo by comparing it with registered photos of known persons.

Face verification under unconstrained conditions is still a challenging problem because of various obstacles, such as partial occlusion, illumination and especially pose variation, etc [12]. Different methods have been proposed for face verification in unconstrained environment, most of which use hand-crafted local or global features based on the known structure of a human face.

Instead of manually constructing a new type of hand-crafted features, features can be learned from training samples in deep learning with convolutional neural networks. However, most of these methods usually use complex networks with very deep structures and require powerful computing systems that may not be available for individuals or small-to-medium organizations. This motivates our proposal to use lightweight CNNs with less complex structures for face verification that can be deployed on a regular commodity computer. Specifically, our goal is to propose a novel method that can balance the following criteria: (i) achieve reasonable high accuracy, (ii) process in realtime, and (iii) can be deployed in regular commodity computers.

As there are different ways to devise a CNN, we first define a scheme with a common structure of CNN instances for consideration. Through experiments, we find the best instance in this family of CNNs according to the above criteria can achieve the accuracy of $(82.58 \pm 1.30)\,\%$ on the standard dataset Labeled Faces in the Wild (LFW [11]) following the "Image Restricted, No Outside Data" protocol, higher than the results of existing methods on original images with the same protocol.

The main contributions of our paper are as follows:

- We propose a scheme of lightweight Convolutional Neural Networks for face verification that can be deployed on a popular commodity computer. Not only the best CNN but many other instances of our proposed lightweight CNN scheme can achieve the accuracy of up to 80 % on LFW.
- To speed up the processing time, we eliminate the funnel step [8,10] to transform an input face image to an upright one. Experiments show that multiple CNNs in our method can achieve the accuracy up to 80 % even with original images without transformation.
- The best found CNN instance conforming our proposed lightweight CNN scheme can perform face verification task at 60 fps and 235 fps with CPU-only and GPU implementations on a regular computer respectively. Therefore, it will be suitable to integrate face verification component into various applications for individuals and small-to-medium organizations without special requirements on hardware devices.

The rest of the paper is organized as follows. In Sect. 2, we briefly review the background and related works. Our proposed scheme of lightweight CNNs for face verification is in Sect. 3. Experimental results on LFW dataset with various instances of our proposed lightweight CNN scheme and the best found CNN are discussed in Sect. 4. The final section is for conclusion and future work.

2 Background and Related Works

Although face verification with images captured in constrained conditions has been well studied and achieves high accuracy, it is still a challenging problem to handle face images captured in unconstrained environment because of various factors, such as illumination, partial occlusion, and especially pose variation [13].

One of the common trends for face verification is to extract features or facial components from a face image, then perform different techniques and metrics [2] to evaluate the matching score between faces in a pair of photos. Several methods use local features, such as SIFT [4,5], LBP [4,5], Walsh LBP [12], Gabor [21], LE [4]. These features can also be combined to boost the overall performance. Structures of a face are also extracted, such as hierarchical-probabilistic elastic parts [13]. Some methods also exploit 3D structure of a face from an image for face frontalization [7] or face alignment with explicit 3D modeling of faces [20].

While most methods for face verification use hand-crafted features, using deep neural networks has become a new trend for face verification [6,9], especially DeepFace [20]. In these methods, instead of manually constructing a specific feature type, features can be learned from trained data.

Most of existing methods with deep neural networks for face verification use complex networks with very deep structures and process large data to train such networks. These tasks require super-computing systems with high computational resources to handle huge computation workload. These methods and systems provide high accuracy to process millions of people. However, it would be necessary to devise lightweight neural networks that can utilize the structures and techniques in common deep neural networks for face verification. Such lightweight versions can be deployed on regular commodity computers and may be appropriate for individuals or small-to-medium organizations to realize their own solutions for face verification.

3 Proposed Method

3.1 Overview

Face verification is solved as a binary classification, in which input is a correlation representation of a pair of images and output is a binary label determining whether the pair is matched or mismatched.

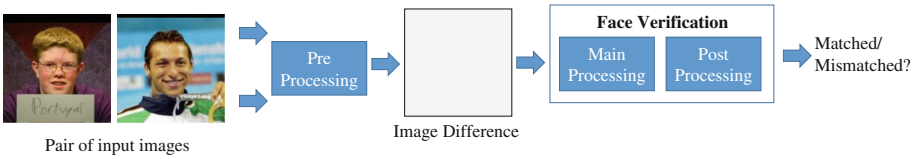

Fig. 1. Overview of face verification process

Figure 1 illustrates the overview of our proposed process of face verification. An image difference is generated from a pair of face images and fed into Face Verification module, which contains two main parts of *Main processing* and *Post-processing*. The first part is for feature extraction and learning while the second part improves the overall performance and guarantees the desired format of output. The final output is "Matched" or "Mismatched" which means that the network predicts the 2 images to be of the same person or to capture different faces respectively.

In the pre-processing step, each input image is transformed into grayscale, resize to a specific size, then normalized so that its histogram follows the zero-mean Gaussian distribution $N(0, \sigma)$ to reduce the impact of illumination. In our system, we choose the image size of 100×100 pixels and $\sigma = 1$.

To speed up the whole process of face verification, we intentionally skip all pre-processing techniques that may require extra/high computational cost, such as Face Frontalization [7] or even the funnel transformation [8,10].

There are several ways to calculate element-wise differences between two normalized images [15], such as squared, absolute-value, and square-root absolute-value. In our proposed method, we take the pixel intensity difference between 2 normalized images as the correlation representation to feed into the face verification process. In the training phase, for a given pair of input images (I_1 and I_2), we generate two different training inputs corresponding to $I_1 - I_2$ and $I_2 - I_1$ with the same label (matched/mismatched) so that the CNN can learn this pair in two scenarios. In the testing phase, we only use either $I_1 - I_2$ or $I_2 - I_1$ as the input.

There are different ways to construct the initial structure and modify the structure of a convolutional neural network for a specific problem. Although there are best practices or common approaches to design the structure of a CNN, there are not concrete guidelines for this task. Therefore, to define the search space for a promising structure of CNN for face verification that satisfies our objectives (c.f. Sect. 1), we propose a scheme to construct a family of CNNs with reference to common design patterns of existing CNNs.

3.2 Main Processing

Figure 2 shows the first part in our proposed scheme for CNNs. A face has some features that can be used to distinguish different people: forehead, eye, nose, mouth, chin, cheek, overall face shape, etc. The network is designed to first extract and describe those features separately, then combine them to form a complete feature vector of the difference of faces. In other words, local features of individual parts are used to distinguish the difference of faces. Then, those features are combined in a relatively spatial manner, which means that spatial information of the facial parts is considered but the image transformation (such as translation, reflection, rotation) has little effect on the network performance.

Main processing consists of K rounds, each consists of a convolutional layer and a pooling layer. Since the face has some small parts (eyes, nose, mouth, chin, cheek, ears, etc.) that can be used to distinguish different people, a *convolutional*

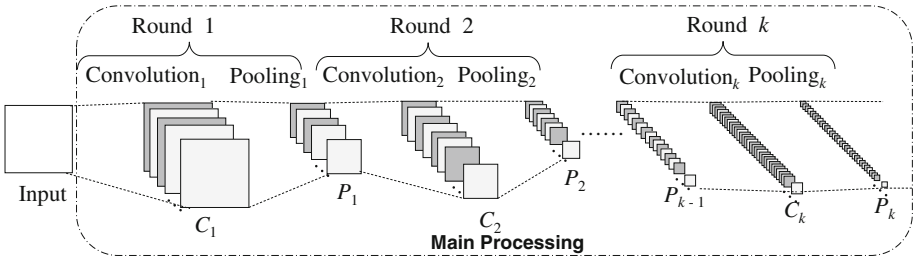

Fig. 2. Structure of main processing module.

layer applies a filter bank with multiple filters of the same size to the image to focus on amplifying local features while considering only a neighboring area for each pixel. A ReLU (rectified linear unit) layer is always embedded right after each convolutional layer to add non-linearity property to the network, otherwise the learning algorithm only works for linear models. Besides, ReLU layers help to ensure the sparsity for features learned in the network. For simplicity of presentation, we do not mention ReLU layers in the following sections as they implicitly exist in convolutional layers.

After a convolutional layer, a *pooling layer* with a kernel of 2×2 makes the image representation more compact and more abstract by downsampling and eliminating redundant information. In the proposed scheme, we use the max-pooling, which keeps only the maximum value in a region. The underlying assumption to use pooling layer is that the probability that any neighboring pixels of a feature point is also a feature point is low. The deeper the network is, the more detailed and abstract the learned features are.

The i^{th} convolutional layer has the following parameters: $kernel_i$ and n_i^C are the size of the kernel and the number of filters in its filter banks, respectively. The number of filters in filter banks are in ascending order, i.e. $n_i^C < n_{i+1}^C$ for $1 \leq i < K$. We set the value of the stride in all convolutional layers to 1.

In practice, we define two *strategies for a filter bank* in each convolutional layer in a round: (A) small filters and (B) large filters. We consider small filters with $3 \leq kernel_i \leq 7$ and large filters with $17 \leq kernel_i \leq 23$. In the first convolutional layer, the number of filters n_1^C starts at 10 and increases with the step of 5. In subsequent convolutional layer, we always ensure that $10 \leq n_{i+1}^C - n_i^C \leq 50$.

3.3 Post Processing

Figure 3 shows the detailed structure of the post-processing part. Similar to those in main processing part, two more convolutional layers are used to extract local features from the feature map. To generate a flat feature vector the last layers in the network, we gradually decrease the number of filters in filter banks in post processing part, i.e. $n_K^C > n_{K+1}^C > n_{K+2}^C$. We also apply Dropout [18] as a simple technique to prevent the network from overfitting.

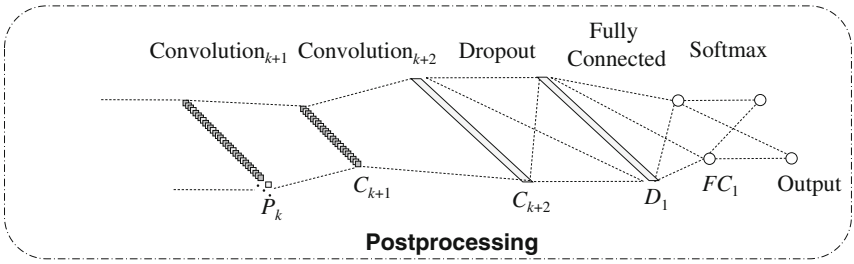

Fig. 3. Structure of post processing module.

Several fully connected layers are usually used as the last layers of a neural network, e.g. GooLeNet [19], VGG network [3,17], fully convolutional semantic segmentation model [14]. However, we use convolutional layers instead of fully connected ones to follow the pure idea of CNN: each feature can be extracted from a local area. Only one fully connected layer is used right before the softmax layer the feature dimension is small then. Moreover, the use of convolutional layers reduces complexity and the number of weights for the network due to shared weight and spatial subsampling properties of CNN. In test (or practical use) phase, the very last layer computes the probabilities that the correlation difference belongs to each label ("matched" or "mismatched") and stores them as a vector. In training, softmax layer is replaced by log-loss softmax, which first computes the probabilities that the correlation representation belongs to each label and then the error penalty for wrong prediction, which is the complement of the probability the correlation representation is categorized to the correct label.

With the mentioned above structure of main processing and post processing parts of CNN in our proposed scheme, we can define a family of CNN instances with similar structures. For each value of K, we can instantiate multiple convolutional neural networks with $2K+4$ hidden layers, depending on different choices for parameters in $K+2$ convolutional layers. As we only consider lightweight CNNs, we choose $K \leq 5$ to keep CNN not to be too complicated with a very deep structure.

4 Experiments and Results

4.1 Labeled Faces in the Wild

Experiments are conducted on Labeled Faces in the Wild (LFW) [11], a standard dataset for unconstrained face verification. LFW contains $13,233$ images of $5,749$ people collected from the Internet. There are numerous variations in images: 2-D rotation, inconsistent illumination, non-frontal poses, major change in expression, accessories obscuring, inconsistent localization, etc. The only constraint is that faces must be detected by Viola-Jones face detector. Figure 4 shows samples of matched pairs (left) and mismatched pairs (right) of LFW.

Fig. 4. Samples of matched pairs (left) and mismatched pairs (right) in LFW dataset.

There are 2 main groups of data in LFW: the original dataset and 3 aligned datasets, including funneled images [8], LFW-a , and deep funneled images [10]. All these aligned methods try to standardize the images under the criteria: the face is rotated to be straight vertically, scaling is applied in order to ensure an equivalent proportion that face occupy over images.

In this paper, we target the original dataset without any transformations to evaluate the efficiency of CNN in face verification even with raw original inputs. Furthermore, we also want to skip extra computation for alignment in real applications to speed up the whole face verification process.

Among 6 common protocols to conduct experiments with LFW, we follow the protocol of "Image Restricted, No Outside Data". In the restricted configuration, the pairs of matched and mismatched images have been given beforehand by configuration files on LFW dataset homepage, whereas the unrestricted configuration allows the researchers to freely create matched and mismatched pairs. In both cases, the identity of each image is not allowed to be used to support classification. Besides, we do not use any extra outside data for training.

4.2 Experiment Set 1: Model Selection for Main Processing Part

We use View 1 of LFW to quickly evaluate different CNN instances conforming our proposed CNN scheme with different number of rounds K and different kernel sizes in convolutional layers. View 1 of LFW only contains $1,100$ matched and mismatched pairs of photos for training and 500 pairs of photo for testing.

As mentioned in Sect. 3.2, there are two strategies to select the kernel size for each convolutional layer. Small filters with kernel size $3 \leq kernel_i \leq 7$, denoted by Strategy A, can be used to extract small parts/features on the face, such as nose, mouth, eye, chin, or cheek. Large filters with kernel size $17 \leq kernel_i \leq 23$, denoted by Strategy B, can be appropriate to capture the overall features in a face. For each convolutional layer in the main processing part, we try either of the two strategies independently from other convolutional layers. In general, there are 2^K different configurations for each value of K.

For $K = 1$, as there is only one convolutional layer, there are 2 configurations, A or B. For $K = 2$, A-B is a possible configuration with small filters in the first convolutional layer and large filters in the second convolutional layer. There are 3 other possible configurations for $K = 2$: A-A, B-A, and B-B.

For each configuration, we empirically evaluate on View 1 of LFW to select the best instance of CNN conforming to our proposed CNN scheme.

Table 1. Results of model selection for main processing part on dataset LFW.

K	1		2			3
Configuration	A	B	A-B	A-A	B-A	A-B-A
Accuracy	79.5 %	79.2 %	78.8 %	81.5 %	80.1 %	**83.7 %**

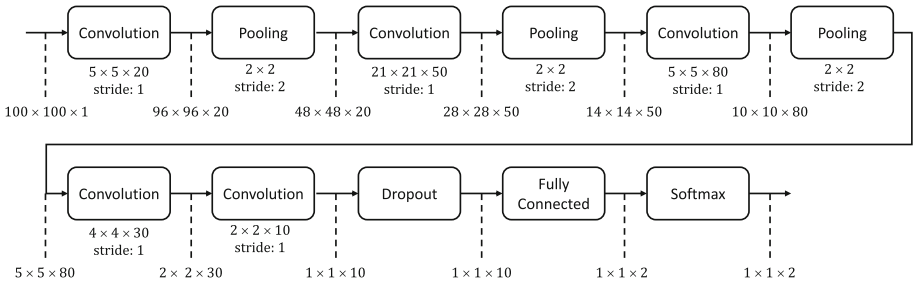

Fig. 5. Structure of the chosen CNN instance with $K = 3$ (feature dimensions are connected with dotted lines)

Table 1 demonstrates the results of model selection for main processing part with several configurations. For $K = 1$, configuration B leads to the lowest accuracy. In case $K = 2$, configuration A - B has less significant performance than other settings. This may be because of using a large window size kernel at the end may reduce the chance of learning detail information. For $K = 4$ and 5, the results are lower than that of $K = 3$, as the networks become too complicated and there are not enough samples to train such networks.

The best CNN instance found empirically (Fig. 5) in our proposed scheme has $K = 3$ and follows the configuration A-B-A, which means that the second convolutional layer uses a large kernel size while the other two convolutional layers use small kernel sizes. This CNN instance achieves the accuracy of 83.7 % on View 1 of LFW. We select this CNN instance for experiment set 2.

4.3 Experiment Set 2: Benchmark Result on LFW Dataset

View 2 of LFW is used for final performance report. There are 10 disjoint sets divided by the dataset creatorsto be used in 10-fold cross validation learning algorithm. At each time, a set is considered as the test set and the other 9 sets are fed to training phase. Table 2 shows the results of face verification on LFW with "Image Restricted, No Outside Data" protocol. We learn 10 models and have 10 corresponding accuracy results.

Experiments show that the best CNN instance found in Sect. 4.2 achieves the accuracy of (82.58 ± 1.30) % on 10 sets of LFW. Table 3 shows the results of other works on dataset LFW with the same setting as our experiments (image-restricted and no outside data). The first 3 rows are results of methods on the original dataset, while other methods use funneled dataset in which faces have

Table 2. Results of face verification on 10-fold dataset LFW.

Test set	1	2	3	4	5
Accuracy	85.500 %	81.833 %	82.500 %	83.667 %	80.833 %
Test set	6	7	8	9	10
Accuracy	82.667 %	82.167 %	82.333 %	81.333 %	83.000 %

Table 3. Results on LFW with image-restricted and no outside data setting. *(Source:* http://vis-www.cs.umass.edu/lfw/results.html. *Accessed on October 1, 2015)*

Methods	Accuracy
Matthew A. Turk et al., CVPR 1991, *original*	(60.02 ± 0.79) %
Eric Nowak et al., CVPR 2007, *original*	(72.45 ± 0.40) %
Conrad Sanderson et al., ICB 2009, *original*	(72.95 ± 0.55) %
Gary B. Huang et al., ICCV 2007, *funneled*	(73.93 ± 0.49) %
Lior Wolf et al., ECCV 2008, *funneled*	(78.47 ± 0.51) %
Nicolas Pinto et al., CVPR 2009, *funneled*	(79.35 ± 0.55) %
Shervin Rahimzadeh Arashloo et al., BTAS 2013, *funneled*	(79.08 ± 0.14) %
Our proposed method, ***original***	**(82.58 ± 1.30) %**
Haoxiang Li et al., CVPR 2013, *funneled*	(84.08 ± 1.20) %
Karen Simonyan et al., BMVC 2013, *funneled*	(87.47 ± 1.49) %
Haoxiang Li et al., ACCV 2014, *funneled*	(88.97 ± 1.32) %
Haoxiang Li et al., CVPR 2015, *funneled*	(91.10 ± 1.47) %
Shervin Rahimzadeh Arashloo et al., TIFS 2014, *funneled*	(95.89 ± 1.94) %

been transformed and aligned for better results. Our method provides the best result on the original data, outperform nearly 10 % higher than the current best result of (72.95 ± 0.55) % on original data [16]. Besides, the result of our method is also higher than those of several methods that take advantages of funneled data [1,15].

4.4 Performance

As one of the criteria for our proposed method is that it can be deployed on a regular computer, we evaluate the processing speed of our method with two implementations. Using CPU-only implementation, our method runs at 60 fps on a computer with Intel Core i7 2.4 GHz and 8 GB RAM. With GPU-enabled implementation, our method runs up to 235 fps on a computer with Intel Core i7 2.4 GHz, 16 GB RAM, and Nvidia GT750M.

5 Conclusion

We propose a lightweight Convolutional Neural Network for face verification with images captured in unconstrained environment. We first define the structure of a CNN scheme with the parameter K, the number of convolution-pooling pairs in the main processing part of the network, and other necessary parameters for each layer in a network. By this way, we can define the search space for a lightweight CNN instance conforming to our proposed scheme that can provide the best accuracy for face verification.

Experiments on LFW dataset show that the best found CNN instance (with $K = 3$ and 10 hidden layers) achieves the best accuracy of $(82.58 \pm 1.30)\,\%$, higher than the current best accuracy of $(72.95 \pm 0.55)\,\%$ on the original data following the protocol "Image Restricted, No Outside Data" [16]. Besides, not only the best instance but many other CNN instances in our proposed CNN scheme can achieve up to 80 % of accuracy on the original data. This supports the idea that it is possible to devise a not-very-complex CNN that can provide sufficient accuracy for face verification, even with photos without special transformation or alignment.

Because of the simplicity of our chosen CNN structure, face verification can be performed in realtime with up to 60 fps (CPU only) and 235 fps (GPU-enabled). Thus, our method can be applied into various practical applications, such as smart house, video surveillance, robotics, or platforms for Internet-of-Things to provide appropriate services and information to users in realtime.

As current successful methods with deep convolutional neural networks usually have complex structures with very deep layers to process huge amount of data, it might lead to the consideration that CNN may not be applicable for regular commodity computers. It is obvious that a lightweight CNN cannot achieve the high accuracy as deep networks for face verification or other tasks. However, it may be useful for individuals and small-to-medium organizations in various applications without special requirements of computing resources and devices.

Acknowledgement. This article was funded in part by a grant from the Vietnam Education Foundation (VEF). The opinions, findings, and conclusions stated herein are those of the authors and do not necessarily reflect those of VEF.

References

1. Arashloo, S.R., Kittler, J.: Efficient processing of mrfs for unconstrained-pose face recognition. In: IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems, BTAS 2013, pp. 1–8 (2013)
2. Cao, Q., Ying, Y., Li, P.: Similarity metric learning for face recognition. In: IEEE International Conference on Computer Vision, ICCV 2013, pp. 2408–2415 (2013)
3. Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: delving deep into convolutional nets. In: British Machine Vision Conference, BMVC 2014 (2014)

4. Chen, D., Cao, X., Wang, L., Wen, F., Sun, J.: Bayesian face revisited: a joint formulation. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part III. LNCS, vol. 7574, pp. 566–579. Springer, Heidelberg (2012)
5. Chen, D., Cao, X., Wen, F., Sun, J.: Blessing of dimensionality: high-dimensional feature and its efficient compression for face verification. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2013, pp. 3025–3032 (2013)
6. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005, pp. 539–546 (2005)
7. Hassner, T., Harel, S., Paz, E., Enbar, R.: Effective face frontalization in unconstrained images. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015 (2015)
8. Huang, G.B., Jain, V., Learned-Miller, E.G.: Unsupervised joint alignment of complex images. In: IEEE International Conference on Computer Vision, ICCV 2007, pp. 1–8 (2007)
9. Huang, G.B., Lee, H., Learned-Miller, E.G.: Learning hierarchical representations for face verification with convolutional deep belief networks. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2012, pp. 2518–2525 (2012)
10. Huang, G.B., Mattar, M.A., Lee, H., Learned-Miller, E.G.: Learning to align from scratch. In: 26th Annual Conference on Neural Information Processing Systems, NIPS 2012, pp. 773–781 (2012)
11. Huang, G.B., Ramesh, M., Berg, T., Learned-Miller, E.: Labeled faces in the wild: a database for studying face recognition in unconstrained environments. Technical report 07-49, University of Massachusetts, Amherst, October 2007
12. Juefei-Xu, F., Luu, K., Savvides, M.: Spartans: single-sample periocular-based alignment-robust recognition technique applied to non-frontal scenarios. IEEE Trans. Image Process. **24**(12), 4780–4795 (2015)
13. Li, H., Hua, G.: Hierarchical-PEP model for real-world face recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, pp. 4055–4064 (2015)
14. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. CoRR abs/1411.4038 (2014)
15. Pinto, N., DiCarlo, J.J., Cox, D.D.: How far can you get with a modern face recognition test set using only simple features? In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2009, pp. 2591–2598 (2009)
16. Sanderson, C., Lovell, B.C.: Multi-region probabilistic histograms for robust and scalable identity inference. In: Tistarelli, M., Nixon, M.S. (eds.) ICB 2009. LNCS, vol. 5558, pp. 199–208. Springer, Heidelberg (2009)
17. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Computing Research Repository - CoRR abs/1409.1556 (2014)
18. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
19. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. Computing Research Repository-CoRR abs/1409.4842 (2014)
20. Taigman, Y., Yang, M., Ranzato, M., Wolf, L.: Deepface: closing the gap to human-level performance in face verification. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, pp. 1701–1708 (2014)
21. Yi, D., Lei, Z., Li, S.Z.: Towards pose robust face recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2013, pp. 3539–3545 (2013)