# PCANet: A Simple Deep Learning Baseline for Image Classification?

Tsung-Han Chan, *Member, IEEE*, Kui Jia, Shenghua Gao, Jiwen Lu, *Senior Member, IEEE*, Zinan Zeng, and Yi Ma, *Fellow, IEEE*

***Abstract*— In this paper, we propose a very simple deep learning network for image classification that is based on very basic data processing components: 1) cascaded principal component analysis (PCA); 2) binary hashing; and 3) blockwise histograms. In the proposed architecture, the PCA is employed to learn multistage filter banks. This is followed by simple binary hashing and block histograms for indexing and pooling. This architecture is thus called the PCA network (PCANet) and can be extremely easily and efficiently designed and learned. For comparison and to provide a better understanding, we also introduce and study two simple variations of PCANet: 1) RandNet and 2) LDANet. They share the same topology as PCANet, but their cascaded filters are either randomly selected or learned from linear discriminant analysis. We have extensively tested these basic networks on many benchmark visual data sets for different tasks, including Labeled Faces in the Wild (LFW) for face verification; the MultiPIE, Extended Yale B, AR, Facial Recognition Technology (FERET) data sets for face recognition; and MNIST for hand-written digit recognition. Surprisingly, for all tasks, such a seemingly naive PCANet model is on par with the state-of-the-art features either prefixed, highly hand-crafted, or carefully learned [by deep neural networks (DNNs)]. Even more surprisingly, the model sets new records for many classification tasks on the Extended Yale B, AR, and FERET data sets and on MNIST variations. Additional experiments on other public data sets also demonstrate the potential of PCANet to serve as a simple but highly competitive baseline for texture classification and object recognition.**

***Index Terms*— Convolution neural network, deep learning, PCA network, random network, LDA network, face recognition, handwritten digit recognition, object classification.**

## I. Introduction

IMAGE classification based on visual content is a very challenging task, largely because there is usually a large amount of intra-class variability, arising from different lighting conditions, misalignment, non-rigid deformations, occlusion and corruptions. Numerous efforts have been made to counter the intra-class variability by manually designing low-level features for classification tasks. Representative examples are Gabor features and local binary patterns (LBP) for texture and face classification and SIFT and HOG features for object recognition. Although the low-level features can be hand crafted with great success for certain data and tasks, designing effective features for new data and tasks usually requires new domain knowledge because most hand-crafted features cannot simply be adapted to new conditions [1], [2].

Learning features from data of interest is considered as a plausible method of remedying the limitations of hand-crafted features. An example of such methods is learning through deep neural networks (DNNs), which has recently garnered significant attention [1]. The idea of deep learning is to discover multiple levels of representation with the hope that higher level features can represent more abstract semantics of the data. Such abstract representations learned from a deep network are expected to provide greater robustness to intra-class variability. One key ingredient to the success of deep learning in image classification is the use of convolutional architectures [3]–[10]. A convolutional deep neural network (ConvNet) architecture [3]–[5], [8], [9] consists of multiple trainable stages stacked on top of each other followed by a supervised classifier. Each stage generally consists of "three layers" – a convolutional filter bank layer, a nonlinear processing layer, and a feature pooling layer. To learn a filter bank in each stage of ConvNet, a variety of techniques, such as restricted Boltzmann machines (RBM) [7] and regularized auto-encoders and their variations, has been proposed; see [2] for a review and references therein. In general, such a network is typically learned using a stochastic gradient descent (SGD) method. However, learning a network that is useful for classification critically depends on expertise in parameter tuning and various *ad hoc* tricks.

Although many variations of deep convolutional networks have been proposed for different vision tasks and their success is usually empirically justified, arguably the first instance that has led to a clear mathematical justification is the wavelet scattering networks (ScatNet) [6], [10]. The only difference in that case is that the convolutional filters in ScatNet are prefixed – they are simply wavelet operators; hence,

Manuscript received August 27, 2014; revised January 17, 2015 and June 12, 2015; accepted August 28, 2015. Date of publication September 1, 2015; date of current version September 23, 2015. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Jean-Philippe Thiran.

T.-H. Chan is with MediaTek Inc., Hsinchu 30078, Taiwan (e-mail: thchan@ieee.org).

K. Jia is with Department of Computer and Information Science, Faculty of Science and Technology, University of Macau, Macau SAR, China (e-mail: kuijia@umac.mo).

S. Gao is with the School of Information Science and Technology, ShanghaiTech University, Shanghai 200031, China (e-mail: gaoshh@shanghaitech.edu.cn).

J. Lu is with the Department of Automation, Tsinghua University, Beijing 100084, China (e-mail: elujiwen@gmail.com).

Z. Zeng is with the Advanced Digital Sciences Center, Singapore 138632 (e-mail: zeng.zinan@gmail.com).

Y. Ma is with the School of Information Science and Technology, ShanghaiTech University, Shanghai 200031, China, and also with the Department of Electrical and Computer Engineering, University of Illinois at Urbana–Champaign, Urbana, IL 61801 USA (e-mail: mayi@shanghaitech.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2015.2475625

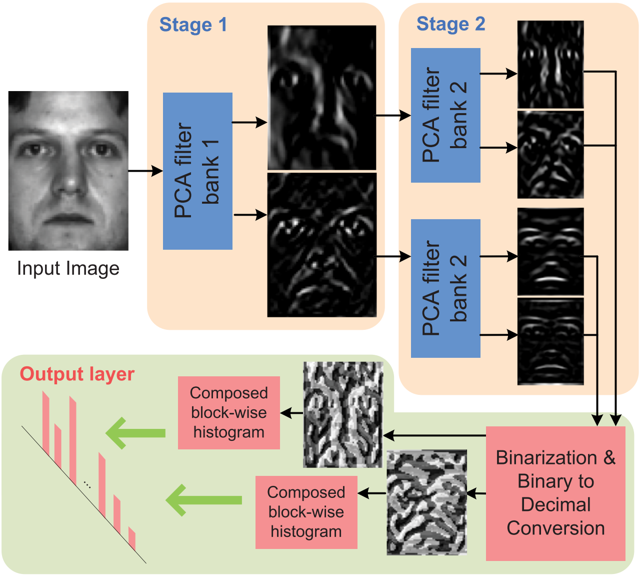

Stage 1
Stage 2
Input Image
PCA filter bank 1
PCA filter bank 2
PCA filter bank 2
Output layer
Composed block-wise histogram
Composed block-wise histogram
Binarization & Binary to Decimal Conversion

Fig. 1. Illustration of how the proposed PCANet extracts features from an image through the three simplest processing components: PCA filters, binary hashing, and histograms.

no learning is required. Somewhat surprisingly, such a prefixed filter bank, once utilized in the similar multistage architectures of ConvNet or DNNs, has demonstrated superior performance over ConvNet and DNNs in several challenging vision tasks such as hand-written digits and texture recognition [6], [10]. However, as we will observe in this paper, such a prefixed architecture does not generalize very well to tasks such as face recognition, where the intra-class variability includes significant illumination changes and corruption.

### A. Motivations

An initial motivation of our study is the desire to resolve certain apparent discrepancies between ConvNet and ScatNet. We want to achieve two simple goals: First, we want to design a simple deep learning network that should be very easy, even trivial, to train and to adapt to different data and tasks. Second, such a basic network could serve as a good baseline for people to empirically justify the use of more advanced processing components or more sophisticated architectures for their deep learning networks.

The solution comes as no surprise: We use the most basic and easy operations to emulate the processing layers in a typical (convolutional) neural network mentioned above: The data-adapting convolution filter bank in each stage is chosen to be the most basic PCA filters; the nonlinear layer is set to be the simplest binary quantization (hashing); and for the feature pooling layer, we simply use the block-wise histograms of the binary codes, which are considered as the final output features of the network. For ease of reference, we call this data-processing network a *PCA Network* (PCANet). As an example, Figure 1 illustrates how a two-stage PCANet extracts features from an input image.

At least one characteristic of the PCANet model seems to challenge common wisdom regarding building a deep learning network such as ConvNet [4], [5], [8] and ScatNet [6], [10]: no nonlinear operations in the early stages of the PCANet until the very last output layer, where binary hashing and histograms are utilized, to compute the output features. Nevertheless, as we will see through extensive experiments, such a drastic simplification does not appear to undermine the performance of the network on various typical datasets.

A network closely related to the PCANet could be two-stage oriented PCA (OPCA), which was first proposed for audio processing [11]. Noticeable differences from the PCANet lie in that OPCA does not couple with hashing and local histograms in the output layer. With the covariances of noises as input, OPCA gains additional robustness to noises and distortions. The baseline PCANet can also incorporate the merits of OPCA, thereby likely offering greater robustness to intra-class variability. To this end, we have also explored a supervised extension of PCANet, where we replace the PCA filters with filters that are learned from linear discriminant analysis (LDA), called LDANet. As we will see through extensive experiments, the additional discriminative information does not appear to improve the performance of the network; see Sections II-C, III, and IV. Another, somewhat extreme variation of the PCANet is to replace the PCA filters with completely random filters (say, the filter entries are i.i.d. Gaussian variables), which is called RandNet. In this work, we conducted extensive experiments and fair comparisons of these types of networks with other existing networks such as ConvNet and ScatNet. We hope our experiments and observations will help people gain a better understanding of these networks.

### B. Contributions

Although our initial intention of studying the simple PCANet architecture is to obtain a simple baseline for comparing and justifying other, more advanced deep learning components or architectures, our findings lead to various pleasant but thought-provoking surprises: The very basic PCANet, in a *fair* experimental comparison, is already quite on par with, and often better than, state-of-the-art features (prefixed, hand crafted, or learned from DNNs) for almost all image classification tasks, including face images, hand-written digits, texture images, and object images. More specifically, for face recognition with one gallery image per person, the model achieves a 99.58% accuracy on the Extended Yale B dataset and a greater than 95% accuracy across disguise/illumination subsets in the AR dataset. On the FERET dataset, the model obtains a state-of-the-art average accuracy of 97.25% and achieves its best accuracy of 95.84 and 94.02% on the Dup-1 and Dup-2 subsets, respectively.[1] On the LFW dataset, the model achieves a competitive 86.28% face verification accuracy under the "unsupervised settings". On the MNIST datasets, the model achieves state-of-the-art results for subtasks such as basic, background random, and background image; see Sections III and IV for more details. The overwhelming empirical evidence demonstrates the effectiveness of the proposed PCANet in learning robust invariant features for various image classification tasks.[2]

[1] The results were obtained by following the FERET standard training CD and could be marginally better when the PCANet is trained on the MultiPIE database.

[2] We have uploaded the Matlab source codes of the PCANet to http://mx.nthu.edu.tw/~tsunghan/.

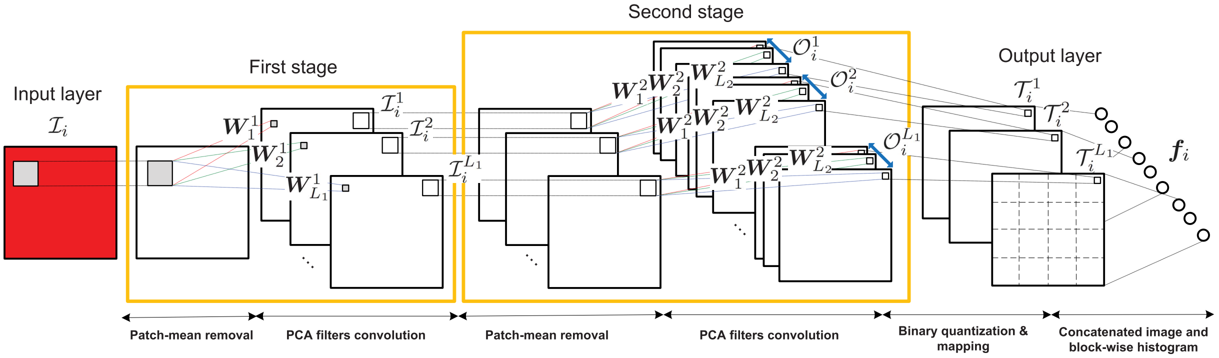


Fig. 2. A detailed block diagram of the proposed (two-stage) PCANet.

The method hardly contains any deep or new techniques, and our study so far is entirely empirical.[3] Nevertheless, a thorough report on such a baseline system has tremendous value to the deep learning and visual recognition community, therein sending both *sobering and encouraging* messages: On the one hand, for future study, the PCANet can serve as a simple but surprisingly competitive baseline for empirically justifying advanced designs of multistage features or networks. On the other hand, the empirical success of the PCANet (and even that of RandNet) reconfirms certain remarkable benefits of cascaded feature learning and extraction architectures. More importantly, because the PCANet consists of only a (cascaded) linear map followed by binary hashing and block histograms, it is amenable to mathematical analysis and justification of its effectiveness. This could lead to fundamental theoretical insights about general deep networks, which currently seem to be an urgent need in deep learning.

## II. Cascaded Linear Networks

### A. Structures of the PCA Network (PCANet)

Suppose that we are given $N$ input training images $\{\mathcal{I}_i\}_{i=1}^N$ of size $m \times n$, and we assume that the patch size (or 2D filter size) is $k_1 \times k_2$ at all stages. The proposed PCANet model is illustrated in Figure 2, and only the PCA filters need to be learned from the input images $\{\mathcal{I}_i\}_{i=1}^N$. In what follows, we more precisely describe each component of the block diagram.

*1) The First Stage (PCA):* Around each pixel, we take a $k_1 \times k_2$ patch, and we collect all (overlapping) patches of the $i$-th image, i.e., $\boldsymbol{x}_{i,1}, \boldsymbol{x}_{i,2}, \ldots, \boldsymbol{x}_{i,\tilde{m}\tilde{n}} \in \mathbb{R}^{k_1 k_2}$, where each $\boldsymbol{x}_{i,j}$ denotes the $j$-th vectorized patch in $\mathcal{I}_i$, $\tilde{m} = m - \lceil k_1/2 \rceil$, $\tilde{n} = n - \lceil k_2/2 \rceil$, and $\lceil z \rceil$ gives the smallest integer greater than or equal to $z$. We then subtract the patch mean from each patch and obtain $\bar{\boldsymbol{X}}_i = [\bar{\boldsymbol{x}}_{i,1}, \bar{\boldsymbol{x}}_{i,2}, \ldots, \bar{\boldsymbol{x}}_{i,\tilde{m}\tilde{n}}]$, where $\bar{\boldsymbol{x}}_{i,j} = \boldsymbol{x}_{i,j} - \frac{\mathbf{1}^T \boldsymbol{x}_{i,j}}{k_1 k_2}\mathbf{1}$ is a mean-removed patch. Here, $\mathbf{1}$ is an all-one vector of proper dimension. By constructing the same matrix for all input images and combining them, we obtain

$$\boldsymbol{X} = [\bar{\boldsymbol{X}}_1, \bar{\boldsymbol{X}}_2, \ldots, \bar{\boldsymbol{X}}_N] \in \mathbb{R}^{k_1 k_2 \times N\tilde{m}\tilde{n}}. \tag{1}$$

Assuming that the number of filters in layer $i$ is $L_i$, PCA minimizes the reconstruction error within a family of orthonormal filters, i.e.,

$$\min_{\boldsymbol{V} \in \mathbb{R}^{k_1 k_2 \times L_1}} \|\boldsymbol{X} - \boldsymbol{V}\boldsymbol{V}^T\boldsymbol{X}\|_F^2, \quad \text{s.t. } \boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}_{L_1}, \tag{2}$$

where $\boldsymbol{I}_{L_1}$ is the identity matrix of size $L_1 \times L_1$. The solution is known as the $L_1$ principal eigenvectors of $\boldsymbol{X}\boldsymbol{X}^T$. The PCA filters are therefore expressed as

$$\boldsymbol{W}_l^1 \doteq \text{mat}_{k_1,k_2}(\boldsymbol{q}_l(\boldsymbol{X}\boldsymbol{X}^T)) \in \mathbb{R}^{k_1 \times k_2}, \quad l = 1, 2, \ldots, L_1, \tag{3}$$

where $\text{mat}_{k_1,k_2}(\boldsymbol{v})$ is a function that maps $\boldsymbol{v} \in \mathbb{R}^{k_1 k_2}$ to a matrix $\boldsymbol{W} \in \mathbb{R}^{k_1 \times k_2}$ and $\boldsymbol{q}_l(\boldsymbol{X}\boldsymbol{X}^T)$ denotes the $l$-th principal eigenvector of $\boldsymbol{X}\boldsymbol{X}^T$. The leading principal eigenvectors capture the main variation of all of the mean-removed training patches. Of course, similar to DNN and ScatNet, we can stack multiple stages of PCA filters to extract higher level features.

*2) The Second Stage (PCA):* Almost repeating the same process as in the first stage, let the $l$-th filter output of the first stage be

$$\mathcal{I}_i^l \doteq \mathcal{I}_i * \boldsymbol{W}_l^1, \quad i = 1, 2, \ldots, N, \tag{4}$$

where $*$ denotes 2D convolution, and the boundary of $\mathcal{I}_i$ is zero-padded before convolving with $\boldsymbol{W}_l^1$ so as to make $\mathcal{I}_i^l$ have the same size as $\mathcal{I}_i$. As in the first stage, we can collect all of the overlapping patches of $\mathcal{I}_i^l$, subtract the patch mean from each patch, and form $\bar{\boldsymbol{Y}}_i^l = [\bar{\boldsymbol{y}}_{i,l,1}, \bar{\boldsymbol{y}}_{i,l,2}, \ldots, \bar{\boldsymbol{y}}_{i,l,\tilde{m}\tilde{n}}] \in \mathbb{R}^{k_1 k_2 \times \tilde{m}\tilde{n}}$, where $\bar{\boldsymbol{y}}_{i,l,j}$ is the $j$-th mean-removed patch in $\mathcal{I}_i^l$. We further define $\boldsymbol{Y}^l = [\bar{\boldsymbol{Y}}_1^l, \bar{\boldsymbol{Y}}_2^l, \ldots, \bar{\boldsymbol{Y}}_N^l] \in \mathbb{R}^{k_1 k_2 \times N\tilde{m}\tilde{n}}$ for the matrix, collecting all mean-removed patches of the $l$-th filter output and concatenate $\boldsymbol{Y}^l$ for all of the filter outputs as

$$\boldsymbol{Y} = [\boldsymbol{Y}^1, \boldsymbol{Y}^2, \ldots, \boldsymbol{Y}^{L_1}] \in \mathbb{R}^{k_1 k_2 \times L_1 N\tilde{m}\tilde{n}}. \tag{5}$$

The PCA filters of the second stage are then obtained as

$$\boldsymbol{W}_\ell^2 \doteq \text{mat}_{k_1,k_2}(\boldsymbol{q}_\ell(\boldsymbol{Y}\boldsymbol{Y}^T)) \in \mathbb{R}^{k_1 \times k_2}, \quad \ell = 1, 2, \ldots, L_2. \tag{6}$$

For each input $\mathcal{I}_i^l$ of the second stage, one will output $L_2$ images of size $m \times n$, and each convolves $\mathcal{I}_i^l$ with $\boldsymbol{W}_\ell^2$ for $\ell = 1, 2, \ldots, L_2$:

$$\mathcal{O}_i^l \doteq \{\mathcal{I}_i^l * \boldsymbol{W}_\ell^2\}_{\ell=1}^{L_2}. \tag{7}$$

The number of output images at the second stage is $L_1 L_2$. One can simply repeat the above process to build more (PCA) stages if a deeper architecture is found to be beneficial.

[3] We would be surprised if something similar to PCANet or variations to OPCA [11] have not been suggested or experimented with before in the vast learning literature.

*3) Output Stage (Hashing and Histograms):* Each of the $L_1$ input images $\mathcal{I}_i^l$ for the second stage has $L_2$ real-valued outputs $\{\mathcal{I}_i^l * \boldsymbol{W}_\ell^2\}_{\ell=1}^{L_2}$ from the second stage. We binarize these outputs and obtain $\{H(\mathcal{I}_i^l * \boldsymbol{W}_\ell^2)\}_{\ell=1}^{L_2}$, where $H(\cdot)$ is a Heaviside step (like) function, whose value is one for positive entries and zero otherwise.

Around each pixel, we view the vector of $L_2$ binary bits as a decimal number. This converts the $L_2$ outputs in $\mathcal{O}_i^l$ back into a single integer-valued "image":

$$\mathcal{T}_i^l \doteq \sum_{\ell=1}^{L_2} 2^{\ell-1} H(\mathcal{I}_i^l * \boldsymbol{W}_\ell^2), \tag{8}$$

whose every pixel is an integer in the range $\left[0, 2^{L_2} - 1\right]$. The order and weights of the $L_2$ outputs are irrelevant because here we treat each integer as a distinct "word."

Each of the $L_1$ images $\mathcal{T}_i^l, l = 1, \ldots, L_1$ is partitioned into $B$ blocks. We compute the histogram (with $2^{L_2}$ bins) of the decimal values in each block and concatenate all $B$ histograms into one vector and denote this vector as $\text{Bhist}(\mathcal{T}_i^l)$. After this encoding process, the "feature" of the input image $\mathcal{I}_i$ is then defined to be the set of block-wise histograms, i.e.,

$$\boldsymbol{f}_i \doteq [\text{Bhist}(\mathcal{T}_i^1), \ldots, \text{Bhist}(\mathcal{T}_i^{L_1})]^T \in \mathbb{R}^{(2^{L_2})L_1 B}. \tag{9}$$

The local blocks can be either overlapping or non-overlapping, depending on the application. Our empirical experience suggests that non-overlapping blocks are suitable for face images, whereas overlapping blocks are appropriate for hand-written digits, textures, and object images. Furthermore, the histogram offers some degree of translation invariance in the extracted features, as in hand-crafted features (e.g., scale-invariant feature transform (SIFT) [12] and histogram of oriented gradients (HOG) [13]), learned features (e.g., bag-of-word (BoW) model [14]), and the average and maximum pooling process in ConvNet [3]–[5], [8], [9].

The hyper-parameters of the PCANet include the filter size $k_1, k_2$, the number of filters in each stage $L_1, L_2$, the number of stages, and the block size for local histograms in the output layer. PCA filter banks require that $k_1 k_2 \geq L_1, L_2$. In our experiments in Section III and Section IV, excluding object recognition, we always set $L_1 = L_2 = 8$, which is inspired from the common setting of Gabor filters [15] with 8 orientations, although some fine-tuned $L_1, L_2$ could lead to marginal performance improvements. The hyper-parameters, such as the filter size $k_1, k_2$ and the block size for local histograms, are determined through a grid search with either cross-validation or a validation set. Moreover, we have empirically observed that two-stage PCANet is in general sufficient to achieve good performance and that a deeper architecture does not necessarily lead to further improvements. In addition, a larger block size for local histograms provides greater translation invariance in the extracted feature $\boldsymbol{f}_i$.

*4) Comparison With ConvNet and ScatNet:* Clearly, PCANet shares various similarities with ConvNet [5]. The patch-mean removal in PCANet is reminiscent of local contrast normalization in ConvNet.[4] This operation moves all of the patches to be centered around the origin of the vector space so that the learned PCA filters can better capture major variations in the data. In addition, PCA can be viewed as the simplest class of auto-encoders, which minimizes reconstruction error.

The PCANet contains no non-linearity processes between/in stages, in contrast to the common wisdom regarding building deep learning networks, e.g., the absolute rectification layer in ConvNet [5] and the modulus layer in ScatNet [6], [10]. We have tested the PCANet with an absolute rectification layer added immediately after the first stage, but we did not observe any improvement in the final classification results. This could be because the use of quantization plus a local histogram (in the output layer) already introduces sufficient invariance and robustness in the final feature.

The overall process prior to the output layer in the PCANet is completely linear. One may wonder what would occur if we merge the two stages into only one stage that has an equivalently equal number of PCA filters and receptive field size. Specifically, one may be interested in how the single-stage PCANet with $L_1 L_2$ filters of size $(2k_1 - 1) \times (2k_2 - 1)$ could perform compared to the two-stage PCANet described in Section II-A. We have experimented with such settings on faces and hand-written digits and observed that the two-stage PCANet outperforms this single-stage alternative in most cases; see the last several rows of Tables III, X, and XI. In comparison to the filters learned by the single-stage alternative, the resulting two-stage PCA filters essentially have a low-rank factorization, possibly resulting in a lower chance of over-fitting the dataset. Regarding why we need the deep structure, from a computational perspective, the single-stage alternative requires learning filters with $L_1 L_2 (2k_1 - 1)(2k_2 - 1)$ variables, whereas the two-stage PCANet only learns filters with in total $(L_1 + L_2) k_1 k_2$ variables. Another benefit of the two-stage PCANet is that the larger receptive field, because it contains more holistic observations of the objects in images, and its learning invariance can essentially capture more semantic information. Our comparative experiments verify that hierarchical architectures with large receptive fields and multiple stacked stages are more efficient in terms of learning semantically related representations, which agrees with what has been observed in [7].

### B. Computational Complexity

The components for constructing the PCANet are extremely basic and computationally efficient. To observe how low the computational complexity of PCANet would be, let us take the two-stage PCANet as an example. In each stage of the PCANet, forming the patch-mean-removed matrix $\boldsymbol{X}$ costs $k_1 k_2 + k_1 k_2 \tilde{m}\tilde{n}$ flops; the inner product $\boldsymbol{X}\boldsymbol{X}^T$ has a complexity of $2(k_1 k_2)^2 \tilde{m}\tilde{n}$ flops; and the complexity of eigen-decomposition is $\mathcal{O}((k_1 k_2)^3)$. The PCA filter convolution requires $L_i k_1 k_2 mn$ flops for stage $i$. In the output layer, the conversion of $L_2$ binary bits to a decimal number costs $2L_2 \tilde{m}\tilde{n}$, and the naive histogram operation is of complexity $\mathcal{O}(mnBL_2 \log 2)$. By $\tilde{m} = m - \lceil k_1/2 \rceil$, $\tilde{n} = n - \lceil k_2/2 \rceil$ and assuming $mn \gg \max(k_1, k_2, L_1, L_2, B)$, the overall complexity of the PCANet is easily verified as

$$\mathcal{O}(mnk_1k_2(L_1 + L_2) + mn(k_1k_2)^2).$$

[4] We have tested the PCANet without patch-mean removal, and a slightly degraded performance is observed.

The above computational complexity applies to the training and testing phase of the PCANet because the extra computational burden in the training phase from the testing phase is the eigen-decomposition, whose complexity is ignorable when $mn \gg \max(k_1, k_2, L_1, L_2, B)$.

In comparison to ConvNet, the SGD for filter learning is also a simple gradient-based optimization solver, but the overall training time remains much longer than that of the PCANet. For example, training the PCANet on approximately 100,000 images of 80×60 pixels took only half an hour, but CNN-2 took 6 hours, excluding the fine-tuning process; see Section III-A4 for details.

### *C. Two Variations (RandNet and LDANet)*

The PCANet is an extremely simple network, requiring only minimal learning of the filters from the training data. One can immediately think of two possible variations of the PCANet along two opposing directions:

1) We could further eliminate the necessity of training data and replace the PCA filters at each layer with random filters of the same size. Specifically, the random filters, i.e., the elements of $\boldsymbol{W}_l^1$ and $\boldsymbol{W}_l^2$, are generated following standard Gaussian distributions. We call such a network a *Random Network*, or RandNet for short. It is natural to wonder how much degradation such a randomly chosen network would generate compared to the PCANet.
2) If the task of the learned network is classification, we can further enhance the supervision of the learned filters by incorporating the information of class labels in the training data and learn the filters based on the idea of multi-class linear discriminant analysis (LDA). We call such a composed network *LDA Network*, or LDANet for ease of reference. Again, we are interested in how much the enhanced supervision would help improve the performance of the network.

Specifically, we now describe in greater detail how to construct the LDANet. Suppose that the $N$ training images are classified into $C$ classes $\{\mathcal{I}_i\}_{i\in S_c}$, $c = 1, 2, \ldots, C$, where $S_c$ is the set of indices of images in class $c$, and the mean-removed patches associated with each image of distinct classes $\bar{\boldsymbol{X}}_i \in \mathbb{R}^{k_1k_2\times mn}$, $i \in S_c$ (in the spirit of $\bar{\boldsymbol{X}}_i$ in (1)) are given. We can first compute the class mean $\boldsymbol{\Gamma}_c$ and the intra-class variability $\boldsymbol{\Sigma}_c$ for all of the patches as follows:

$$\boldsymbol{\Gamma}_c = \sum_{i\in S_c} \bar{\boldsymbol{X}}_i/|S_c|, \tag{10}$$

$$\boldsymbol{\Sigma}_c = \sum_{i\in S_c} (\bar{\boldsymbol{X}}_i - \boldsymbol{\Gamma}_c)(\bar{\boldsymbol{X}}_i - \boldsymbol{\Gamma}_c)^T/|S_c|. \tag{11}$$

Each column of $\boldsymbol{\Gamma}_c$ denotes the mean of the patches around each pixel in the class $c$, and $\boldsymbol{\Sigma}_c$ is the sum of all of the patch-wise sample covariances in class $c$. Likewise, the inter-class variability of the patches is defined as

$$\boldsymbol{\Phi} = \sum_{c=1}^{C} (\boldsymbol{\Gamma}_c - \boldsymbol{\Gamma})(\boldsymbol{\Gamma}_c - \boldsymbol{\Gamma})^T/C, \tag{12}$$

where $\boldsymbol{\Gamma}$ is the mean of the class means. The idea of LDA is to maximize the ratio of the inter-class variability to the sum of the intra-class variability within a family of orthonormal filters, i.e.,

$$\max_{\boldsymbol{V}\in\mathbb{R}^{k_1k_2\times L_1}} \frac{\mathrm{Tr}(\boldsymbol{V}^T\boldsymbol{\Phi}\boldsymbol{V})}{\mathrm{Tr}(\boldsymbol{V}^T(\sum_{c=1}^{C}\boldsymbol{\Sigma}_c)\boldsymbol{V})}, \quad \text{s.t. } \boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}_{L_1}, \tag{13}$$

where $\mathrm{Tr}(\cdot)$ is the trace operator. The solution is known as the $L_1$ principal eigenvectors of $\tilde{\boldsymbol{\Phi}} = (\sum_{c=1}^{C}\boldsymbol{\Sigma}_c)^{\dagger}\boldsymbol{\Phi}$, where the superscript † denotes the pseudo-inverse. The pseudo-inverse is used to address the case where $\sum_{c=1}^{C}\boldsymbol{\Sigma}_c$ is not of full rank, although there might be another method of addressing this with better numeric stability [16]. The LDA filters are thus expressed as $\boldsymbol{W}_l^1 = \mathrm{mat}_{k_1,k_2}(\boldsymbol{q}_l(\tilde{\boldsymbol{\Phi}})) \in \mathbb{R}^{k_1\times k_2}$, $l = 1, 2, \ldots, L_1$. A deeper network can be built by repeating the same process as above.

## III. Experiments: Face Recognition and Face Verification

In this section, we first explore how the proposed PCANet and the two simple variations (RandNet and LDANet) perform in face recognition and face verification tasks. Additional experiments on hand-written digit recognition, texture discrimination, and object recognition will be reported in the next section.

Face image databases and the classifiers employed in this section are introduced below.

*Databases:* The MultiPIE dataset [17] contains 337 subjects with varying poses, expressions, and illumination conditions. The Extended Yale B dataset [18] consists of 2,414 frontal-face images of 38 individuals, where the images were captured under various laboratory-controlled lighting conditions. The AR dataset [19] consists of over 4,000 frontal images of 126 subjects. These images contain different facial expressions, illumination conditions and disguises. The FERET dataset [20] is a standard dataset used for facial recognition system evaluation. This dataset contains images of 1,196 different individuals, with up to 5 images of each individual captured under different lighting conditions, with non-neutral expressions and over a period of three years. LFW [21] contains 13,233 face images of 5,749 different individuals collected from the web with large variations in pose, expression, illumination, clothing, hairstyles, etc.

Images of various selected subjects from the MultiPIE dataset are used to learn the PCA filters in the PCANet. This trained PCANet is then applied to extract features of new subjects from the MultiPIE, Extended Yale B, AR, and FERET datasets for face recognition. Furthermore, the LFW dataset is used to conduct experiments concerning face verification. Table I lists these databases and the experimental settings.

*Classifiers:* A nearest neighbor (NN) classifier with a chi-squared or cosine distance measure is employed throughout this section.[5] Linear SVM and softmax classifiers are not selected due to the insufficient number of gallery training faces because we are interested in *face recognition with one gallery image per person*, that is, each subject contains

[5] It is known that the chi-squared distance is an effective measure for comparing histogram features, although the cosine distance may be a more common choice.

TABLE I

Summary of the Databases and Experimental Settings Used in Section III: The MultiPIE, Extended Yale B, AR, and FERET Datasets Are Exploited for Face Recognition, and the LFW Dataset Is Used for Face Verification. Additional Information About the Training Sets and Test Sets Is Provided in Their Respective Subsections

| Data Sets | Descriptions | Number of Classes Used | Cropped Image Sizes | Problem Setting |
|---|---|---|---|---|
| MultiPIE [17] | simultaneous variations in pose, expression, and illumination | 120 | 80×60 | Recognition |
| Extended Yale B [18] | illumination changes | 38 | 192×168 | |
| AR [19] | variations in expression, illumination, and disguise | 126 | 165×120 | |
| FERET [20] | variations in expression, illumination, and age | 1,196 | 150×90 | |
| LFW [21] | variations in pose, expression, illumination, clothing, hairstyle, etc. | 2 | 150×80 | Verification |

only one image in the gallery training set. A comparison of the NN, linear SVM and softmax classifiers will also be presented in Section III-A4.

### A. Face Recog.: Training and Testing on MultiPIE Dataset

We first depict how a **generic face training set** is assembled from the MultiPIE dataset [17]. The assembled dataset is used to train the PCANet or to learn the PCA filters. From the total of 337 subjects in the MultiPIE dataset, we selected the images of the 129 subjects that participated in all four sessions. Images of a subject under all illumination conditions and all expressions at pose $-30°$ to $+30°$ with step size $15°$, a total of 5 poses, were collected. We manually selected eye corners as the ground truth for registration and down-sampled the images to 80×60 pixels. The distance between the two outer eye corners is normalized to be 50 pixels. All cropped images are converted to gray scale and normalized to Euclidean unit length. Briefly, this *generic faces training set* is composed of approximately 100,000 images of 129 subjects under all combinations of illuminations, expressions, and near-frontal poses (within $\pm 30°$).

We use the generic faces training set to train the PCANet and, together with data labels, to learn LDANet; then, we apply the trained networks to extract features of the new subjects in the Multi-PIE dataset. Because the 129 subjects who enrolled in all four sessions are used for the PCANet training, all images of the remaining 120 new subjects in Session 1 are used for gallery training and testing. A frontal view of each subject with neutral expression and frontal illumination is used in the gallery, and the remainder are used for testing. We classify all possible variations into 7 test sets: cross illumination, cross expression, cross pose, cross expression-plus-pose, cross illumination-plus-expression, cross illumination-plus-pose, and cross illumination-plus-expression-and-pose. The cross-pose test set is specifically gathered over the poses $-30°$, $-15°$, $+15°$, and $+30°$.

In what follows, the cross illumination test set is used to investigate the impact of the number of filters, different block overlap ratios, and the number of generic faces training samples for the proposed networks. We then compare the proposed networks with state-of-the-art methods on all test sets. Moreover, a performance comparison among the NN, linear SVM and softmax classifiers is also conducted.

*1) Impact of the Number of Filters:* The impact of the number of filters used in these networks on the cross-illumination test set is studied here. The filter size of the networks is $k_1 = k_2 = 5$, and their non-overlapping blocks are of size 8×6. We vary the number of filters in the first stage $L_1$ from 2 to 12 for one-stage networks. When considering two-stage networks, we set $L_2 = 8$ and vary $L_1$ from 4 to 24. The results are shown in Figure 3. One can observe that PCANet-1 achieves the best results for $L_1 \geq 4$ and that PCANet-2 provides the best performance for all $L_1$ being tested. Moreover, the accuracy of the PCANet and LDANet (for both one-stage and two-stage networks) increases for larger $L_1$, and RandNet also exhibits a similar performance trend. Note that the performance of RandNet was averaged over 10 independent runs.

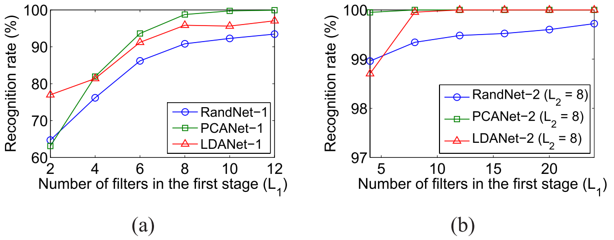


Fig. 3. Recognition accuracy of the PCANet on the MultiPIE cross-illumination test set for varying number of filters in the first stage. (a) PCANet-1; (b) PCANet-2 with $L_2 = 8$.

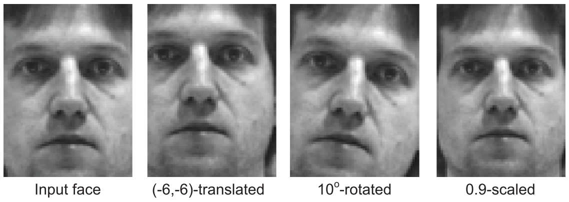
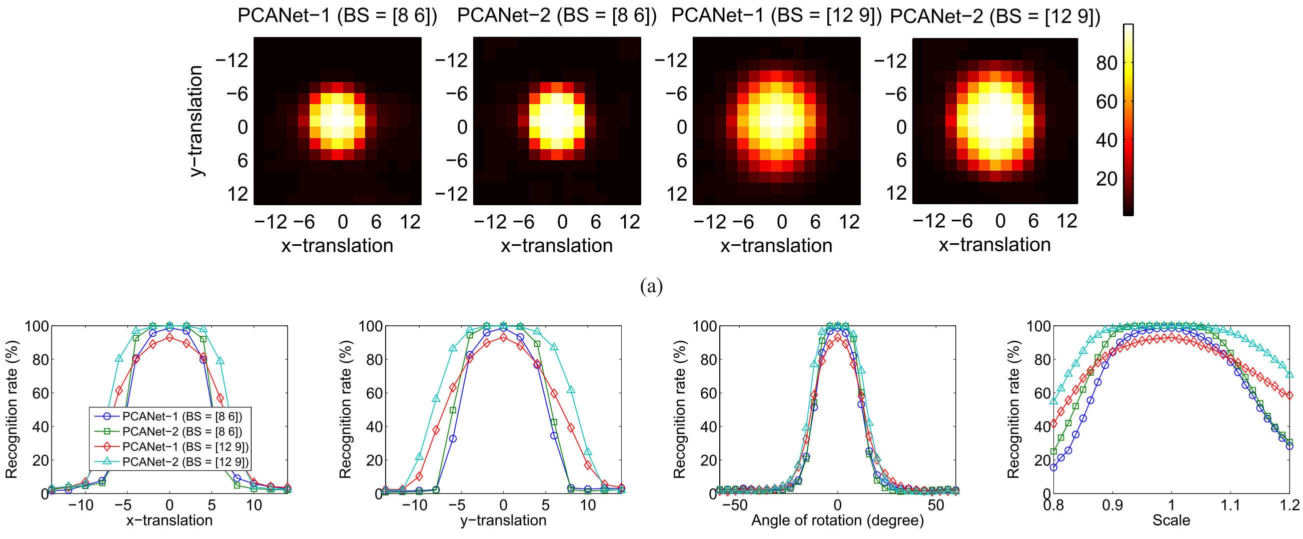


Fig. 4. Original image and its artificially deformed images.

*2) Impact of the the Block Size:* We next examine the impact of the block size (for histogram computations) on the robustness of the PCANet against image deformations. We use the cross-illumination test set and introduce artificial deformation to the testing image with translations, in-plane rotations and scaling; see Figure 4. The parameters of the PCANet are set to $k_1 = k_2 = 5$ and $L_1 = L_2 = 8$. Two block sizes, 8×6 and 12×9, are considered. Figure 5 shows the recognition accuracy for each artificial deformation. PCANet-2 is found to achieve a greater than 90 percent accuracy with translations up to 4 pixels in all directions, with up to 8° in-plane rotations, and with scale varying from 0.9 to 1.075. Moreover, the



(a)

(b) (c) (d) (e)

Fig. 5. Recognition rate of PCANet on MultiPIE cross-illumination test set for different PCANet block sizes and deformations in the test image. Two block sizes [8 6] and [12 9] for histogram aggregation are tested. (a) Simultaneous translation in the $x$ and $y$ directions. (b) Translation in the $x$ direction. (c) Translation in the $y$ direction. (d) In-plane rotation. (e) Scale variation.

TABLE II

FACE RECOGNITION RATES (%) OF THE PCANet ON THE MultiPIE CROSS-ILLUMINATION TEST SET WITH RESPECT TO DIFFERENT NUMBERS OF GENERIC FACES TRAINING IMAGES ($S$)

| $S$ | 100 | 500 | 1,000 | 5,000 | 10,000 | 50,000 |
|---|---|---|---|---|---|---|
| PCANet-1 | 98.01 | 98.44 | 98.61 | 98.65 | 98.70 | 98.70 |
| PCANet-2 | 100 | 100 | 100 | 100 | 100 | 100 |

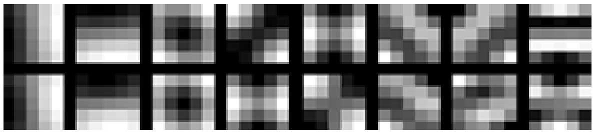


Fig. 6. The PCANet filters learned on the MultiPIE dataset. Top row: the first stage. Bottom row: the second stage.

results suggest that PCANet-2 with a larger block size provides greater robustness against various deformations, but a larger block size may sacrifice performance in PCANet-1.

*3) Impact of the Number of Generic Faces Training Samples:* We also report the recognition accuracy of the PCANet for different numbers of generic faces training images. Again, we use the cross-illumination test set. We randomly select $S$ images from the generic faces training set to train the PCANet and vary $S$ from 100 to $50,000$. The parameters of the PCANet are set to $k_1 = k_2 = 5$, $L_1 = L_2 = 8$, and a block size of 8×6. The results are tabulated in Table II. The accuracy of the PCANet is surprisingly somewhat less sensitive to the number of generic faces training images. The performance of PCANet-1 gradually improves as the number of generic faces training samples increases, and PCANet-2 obtains perfect recognition even when there are only 100 generic faces training samples.

*4) Comparison With State of the Art:* We compare the RandNet, PCANet, and LDANet with Gabor[6] [15], LBP[7] [22], and two-stage ScatNet (ScatNet-2) [6]. We set the parameters of PCANet to the filter size $k_1 = k_2 = 5$, the number of filters $L_1 = L_2 = 8$, and 8×6 block size, and the learned PCANet filters are shown in Figure 6. The number of scales and the number of orientations in ScatNet-2 are set to 3 and 8, respectively. We use an NN classifier with the chi-squared distance for RandNet, PCANet, LDANet and LBP and the cosine distance for Gabor and ScatNet. We should emphasize that the NN classifier with different distance measures is used to secure the best performances for respective features.

We also provide a comparison with CNN. Because we could not find any work that successfully applies CNN to the same face recognition tasks, we used the Caffe framework [23] to pre-train a two-stage CNN (CNN-2) on the generic faces training set. The CNN-2 is a fully supervised network with filter size 5×5, with 20 channels for the first stage and 50 channels for the second stage. Each convolution output is followed by a rectified linear function $relu(x) = \max(x, 0)$ and 2×2 max-pooling. The output layer is a softmax classifier. After pre-training the CNN-2 on the generic faces training set, the CNN-2 is also fine-tuned on the 120 gallery images for 500 epochs.

The performances of all methods are given in Table III. Except on the cross-pose test set, the PCANet yields the best precision. For all test sets, RandNet and LDANet are inferior to the PCANet, and LDANet does not appear to utilize discriminative information. One can also observe that whenever there are illumination variations, the performance of LBP significantly decreases. The PCANet overcomes this drawback and offers a competitive performance compared to

[6] Each face is convolved with a family of Gabor kernels with 5 scales and 8 orientations. Each filter response is down-sampled by a $3 \times 3$ uniform lattice and normalized to zero mean and unit variance.

[7] Each face is divided into several blocks, each with a size that is the same as in PCANet. The histogram of 59 uniform binary patterns is then computed, and the patterns are generated by thresholding 8 neighboring pixels in a circle of radius 2 using the central pixel value.

TABLE III

COMPARISON OF FACE RECOGNITION RATES (%) OF VARIOUS METHODS ON THE MultiPIE TEST SETS. THE FILTER SIZE $k_1 = k_2 = 5$ AND THE NUMBER OF FILTERS $L_1 = L_2 = 8$ ARE SET, AND THE NN CLASSIFIER IS USED IN RandNet, PCANet, AND LDANet UNLESS OTHERWISE SPECIFIED. HERE, $L_1' = L_1L_2$, AND $k_i' = 2k_i - 1,\ i = 1, 2$

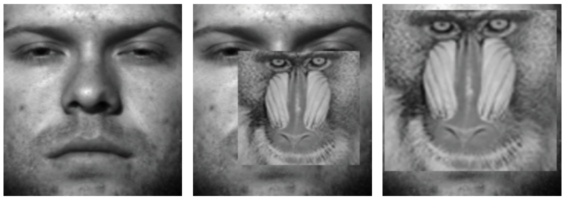
| Test Sets | Illum. | Exps. | Pose | Exps.+Pose | Illum.+Exps. | Illum.+Pose | Illum.+Exps.+Pose |
|---|---|---|---|---|---|---|---|
| Gabor [15] | 68.75 | 94.17 | 84.17 | 64.70 | 38.09 | 39.76 | 25.92 |
| LBP [22] | 79.77 | 98.33 | **95.63** | 86.88 | 53.77 | 50.72 | 40.55 |
| ScatNet-2 [6] | 20.88 | 66.67 | 71.46 | 54.37 | 14.51 | 15.00 | 14.47 |
| CNN-2 [8] | 46.71 | 75.00 | 73.54 | 57.50 | 23.38 | 25.05 | 18.74 |
| RandNet-1 | 80.88 | 98.33 | 87.50 | 75.62 | 46.57 | 42.80 | 31.85 |
| RandNet-2 | 97.64 | 97.50 | 83.13 | 75.21 | 63.87 | 53.50 | 42.47 |
| PCANet-1 | 98.70 | 99.17 | 94.17 | **87.71** | 72.40 | 65.76 | 53.80 |
| PCANet-2 | **100** | 99.17 | 93.33 | 87.29 | **87.89** | **75.29** | **66.49** |
| LDANet-1 | 99.95 | 98.33 | 92.08 | 82.71 | 77.89 | 68.55 | 57.97 |
| LDANet-2 | 96.02 | 99.17 | 93.33 | 83.96 | 65.78 | 60.14 | 46.72 |
| PCANet-1 ($L_1' = 64,\ k_1' = k_2' = 9$) | **100** | 99.17 | 89.58 | 81.46 | 75.74 | 67.59 | 56.95 |
| PCANet-2 (SVM) | 99.77 | **100** | 93.33 | 84.38 | 79.66 | 64.88 | 55.18 |
| PCANet-2 (softmax) | 14.81 | 2.50 | 2.71 | 0.83 | 0.88 | 0.90 | 0.86 |

LBP for cross-pose and cross-expression variations. As a final note, ScatNet and CNN do not perform well.[8] This is the case for all face-related experiments below, and therefore, ScatNet and CNN are not included for comparison in these experiments. We also do not include RandNet and LDANet in the following face-related experiments because they did not show performance improvements over the PCANet.

We also investigated PCANet-1 with $L_1L_2$ filters of size $k_1' \times k_2'$, where $k_i' = 2k_i - 1 = 9,\ i = 1, 2$, as shown in the third to last row of Table III. PCANet-1 with such a parameter setting is used to mimic the reported PCANet-2 in a single-stage network because both have the same number of PCA filters and receptive field size. This may help us understand if one could combine two-stage PCANet into one stage because the process prior to the output layer is entirely linear. The results show that PCANet-2 outperforms the PCANet-1 alternative, which immediately validates the advantages of deeper networks. This observation also holds for all face-related experiments, and thus, the results of such an experimental setting are not tabulated in these experiments.

The last two rows of Table III show the performances of PCANet-2 followed by linear SVM and softmax classifiers. The softmax classifier apparently overfits the 120 gallery face images. Although the linear SVM classifier performs modestly well, it is still not competitive with the NN classifier in most cases.

Another merit worth mentioning in this subsection is the training efficiency of the PCANet. Training PCANet-2 on the generic faces training set (i.e., on approximately 100,000 face images of 80×60 pixels) took only half an hour, but CNN-2 took 6 hours, excluding the fine-tuning process.



0% occlusion 40% occlusion 80% occlusion

Fig. 7. Illustration of varying level of an occluded test face image.

TABLE IV

RECOGNITION RATES (%) ON EXTENDED YALE B DATASET

| Percent occluded | 0% | 20% | 40% | 60% | 80% |
|---|---|---|---|---|---|
| LBP [22] | 75.76 | 65.66 | 54.92 | 43.22 | 18.06 |
| P-LBP [24] | 96.13 | 91.84 | 84.13 | 70.96 | 41.29 |
| PCANet-1 | 97.77 | 96.34 | 93.81 | 84.60 | **54.38** |
| PCANet-2 | **99.58** | **99.16** | **96.30** | **86.49** | 51.73 |

## B. Face Recog.: Testing on Extended Yale B Dataset

The MultiPIE trained PCANet model (or to be more precise, PCA filters) are applied to the Extended Yale B dataset [18]. The cropped 192×168 face images were normalized to Euclidean unit length. For each subject, we selected frontal illumination as the gallery images, and the remainder were used for testing. To challenge ourselves, in the test images, we also simulated various levels of contiguous occlusion, from 0 to 80 percent, by replacing a randomly located square block of each test image with an unrelated image; see Figure 7 for an example. The size of the non-overlapping blocks in the PCANet was set to 8×8. We compare with LBP [22] and LBP of the test images being processed by illumination normalization, P-LBP [24]. We use the NN classifier with the chi-squared distance measure.

The experimental results are given in Table IV. One can observe that the PCANet outperforms the P-LBP with different levels of occlusion. In addition, the PCANet is not only illumination-insensitive but also robust against block occlusion. Under such a single sample per person setting and such difficult lighting conditions, the PCANet surprisingly achieves an almost perfect recognition rate of 99.58% and still sustains 86.49% accuracy when 60% of the pixels of every test image are occluded! The "non-occluded facial part" seems to provide essential information for discriminating between different subjects because the cropped images are spatially

[8] The performance of CNN could be further promoted if the model parameters are more fine-tuned.

TABLE V

RECOGNITION RATES (%) ON AR DATASET

| Test sets | Illum. | Exps. | Disguise | Disguise + Illum. |
|---|---|---|---|---|
| LBP [22] | 93.83 | 81.33 | 91.25 | 79.63 |
| P-LBP [24] | 97.50 | 80.33 | 93.00 | 88.58 |
| PCANet-1 | 98.00 | **85.67** | 95.75 | 92.75 |
| PCANet-2 | **99.50** | 85.00 | **97.00** | **95.00** |

dominated by the entire face and because no background is retained; see Figure 7. This could be because each PCA filter can be seen as a detector with a maximum response for patches from a face. In other words, the contribution from occluded patches was somehow ignored after PCA filtering and are not passed onto the output layer of the PCANet, thereby yielding striking robustness to occlusion.

### C. Face Recog.: Testing on AR Dataset

We further evaluate the ability of the MultiPIE-learned PCANet to address real, possibly malicious occlusions using the AR dataset [19]. In the experiment, we chose a subset of the data consisting of 50 male subjects and 50 female subjects. The images are cropped with dimensions of 165×120, converted to gray scale, and normalized to Euclidean unit length. For each subject, we select the face with frontal illumination and neural expression in the gallery training, and the remainder are used for testing. The size of the non-overlapping blocks in the PCANet is set to 8×6. We also compare with LBP [22] and P-LBP [24]. We use the NN classifier with the chi-squared distance measure.

The results are given in Table V. For the test set of illumination variations, the recognition by PCANet is again almost perfect, and for the cross-disguise-related test sets, the accuracy is greater than 95%. The results are consistent with those on the MultiPIE and Extended Yale B datasets: PCANet is insensitive to illumination and robust to occlusions. To the best of our knowledge, no single feature with a simple classifier can achieve such performances, even if using extended representation-based classification (ESRC) [25]!

### D. Face Recog.: Testing on FERET Dataset

Finally, we apply the MultiPIE-learned PCANet to the popular FERET dataset [20]. The dataset is partitioned into disjoint sets: gallery and probe. The probe set is further subdivided into four categories: *Fb*, with different expression changes; *Fc*, with different lighting conditions; *Dup-I*, taken within a period of three to four months; and *Dup-II*, taken at least one and a half years apart. We use the gray scale images cropped to an image size of 150×90 pixels. The cropped images are also normalized to Euclidean unit length. The size of the non-overlapping blocks in the PCANet is set to 15×15. To perform a fair comparison with prior methods, the dimensions of the PCANet features are reduced to 1000 via a whitening PCA (WPCA),[9] where the projection matrix is learned from the features of gallery samples. The NN classifier with cosine distance is used. Furthermore, in addition to the PCANet trained on the MultiPIE database, we also train the PCANet on the FERET generic training set, which consists of 1,002 images of 429 people listed in the FERET standard training CD.

TABLE VI

RECOGNITION RATES (%) ON THE FERET DATASET

| Probe sets | *Fb* | *Fc* | *Dup-I* | *Dup-II* | Avg. |
|---|---|---|---|---|---|
| LBP [22] | 93.00 | 51.00 | 61.00 | 50.00 | 63.75 |
| DMMA [26] | 98.10 | 98.50 | 81.60 | 83.20 | 89.60 |
| P-LBP [24] | 98.00 | 98.00 | 90.00 | 85.00 | 92.75 |
| POEM [27] | 99.60 | 99.50 | 88.80 | 85.00 | 93.20 |
| G-LQP [28] | **99.90** | **100** | 93.20 | 91.00 | 96.03 |
| LGBP-LGXP [29] | 99.00 | 99.00 | 94.00 | 93.00 | 96.25 |
| sPOEM+POD [30] | 99.70 | **100** | 94.90 | 94.00 | 97.15 |
| GOM [31] | 99.90 | **100** | 95.70 | 93.10 | 97.18 |
| PCANet-1 (Trn. CD) | 99.33 | 99.48 | 88.92 | 84.19 | 92.98 |
| PCANet-2 (Trn. CD) | 99.67 | 99.48 | **95.84** | **94.02** | 97.25 |
| PCANet-1 | 99.50 | 98.97 | 89.89 | 86.75 | 93.78 |
| PCANet-2 | 99.58 | **100** | 95.43 | **94.02** | **97.26** |

TABLE VII

COMPARISON OF VERIFICATION RATES (%) ON LFW USING THE UNSUPERVISED SETTING

| Methods | Accuracy |
|---|---|
| POEM [27] | 82.70±0.59 |
| High-dim. LBP [36] | 84.08 |
| High-dim. LE [36] | 84.58 |
| SFRD [37] | 84.81 |
| I-LQP [28] | 86.20±0.46 |
| OCLBP [33] | **86.66±0.30** |
| PCANet-1 | 81.18 ± 1.99 |
| PCANet-1 (sqrt) | 82.55 ± 1.48 |
| PCANet-2 | 85.20 ± 1.46 |
| PCANet-2 (sqrt) | 86.28 ± 1.14 |

The results of the PCANet and state-of-the-art methods are listed in Table VI. Surprisingly, MultiPIE-learned PCANet-2 and FERET-learned PCANet-2 (with Trn. CD in parentheses) achieve state-of-the-art accuracies of 97.25 and 97.26% on average, respectively. Because the variations in the MultiPIE dataset are much richer than in the standard FERET training set, it is natural that the MultiPIE-learned PCANet slightly outperforms the FERET-learned PCANet. More importantly, PCANet-2 breaks the records in *Dup-I* and *Dup-II*.

*Concluding remarks on face recognition:* A prominent conclusion drawn from the above experiments in Sections III-A, III-B, III-C, and III-D is that training the PCANet from a face dataset can be very effective for capturing the abstract representation of new subjects and new datasets. After the PCANet was trained, extracting a PCANet-2 feature for one test face takes only 0.3 second in Matlab. We can anticipate that the performance of the PCANet can be further improved and moved toward practical use if the PCANet is trained on a wide and deep dataset that collects sufficient inter-class and intra-class variations.

### E. Face Verification on LFW Dataset

In addition to the previous experiments with laboratory face datasets, we also applied the PCANet to the LFW dataset [21] for unconstrained face verification. We consider the "unsupervised setting", which is the best choice for

[9] The PCA projection directions are weighted by the inverse of their corresponding square-root energies.

TABLE VIII

SUMMARY OF THE DATABASES USED IN SECTION IV. MNIST AND ITS VARIATIONS ARE FOR HAND-WRITTEN DIGIT RECOGNITION, CUReT IS FOR TEXTURE DISCRIMINATION, AND CIFAR10 IS FOR OBJECT RECOGNITION

| Data Sets | Description | Num. of Classes | Image sizes | Train-Valid-Test |
|---|---|---|---|---|
| MNIST [4] | Standard MNIST | 10 | 28×28 | 60000-0-10000 |
| *basic* [38] | Smaller subset of MNIST | 10 | 28×28 | 10000-2000-50000 |
| *rot* [38] | MNIST with rotation | 10 | 28×28 | 10000-2000-50000 |
| *bg-rand* [38] | MNIST with noise background | 10 | 28×28 | 10000-2000-50000 |
| *bg-img* [38] | MNIST with image background | 10 | 28×28 | 10000-2000-50000 |
| *bg-img-rot* [38] | MNIST with rotation and image background | 10 | 28×28 | 10000-2000-50000 |
| *rect* [38] | Discriminate between tall and wide rectangles | 2 | 28×28 | 1000-200-50000 |
| *rect-img* [38] | Dataset *rect* with image background | 2 | 28×28 | 10000-2000-50000 |
| *convex* [38] | Discriminate between convex and concave shape | 2 | 28×28 | 6000-2000-50000 |
| CUReT [39] | Textures with various poses and illuminations | 61 | 200×200 | 2806-0-2806 |
| CIFAR10 [40] | Objects with various positions, scales, colors, and textures | 10 | 32×32 | 50000-0-10000 |

evaluating the learned features because it does not depend on metric learning and discriminative model learning. The aligned version of the faces, namely, LFW-a, provided by Wolf *et al.* [32] was exploited. The face images were cropped to 150 × 80 pixels[10] and are normalized to Euclidean unit length. We follow the standard evaluation protocol, which splits the View 2 dataset into 10 subsets, with each subset containing 300 intra-class pairs and 300 inter-class pairs. We perform 10-fold cross validation using the 10 subsets of pairs in View 2. In the PCANet, the filter size, the number of filters, and the (non-overlapping) block size are set to $k_1 = k_2 = 7$, $L_1 = L_2 = 8$, and 15×13, respectively. The performances are measured by averaging the 10-fold cross validation. We project the PCANet features onto 400 and 3,200 dimensions using WPCA for PCANet-1 and PCANet-2, respectively, and we use the NN classifier with the cosine distance.

Table VII lists the results.[11] Note that PCANet followed by sqrt in parentheses represents the PCANet feature taking the square-root operation. One can observe that the square-root PCANet outperforms the PCANet, and such a performance boost can also be noticed for other features used in this dataset [33]. Moreover, the square-root PCANet-2 achieves an accuracy of 86.28%, which is quite competitive to current state-of-the-art methods. This shows that the proposed PCANet is also effective in learning invariant features for face images captured in less controlled conditions.

In preparation of this paper, we became aware of two concurrent works [34], [35] that employ ConvNet for LFW face verification. Although both works achieve very impressive results on LFW, their experimental setting greatly differs from ours. These two works require an outside database to train the ConvNet, and the face images have to be more precisely aligned; e.g., [34] uses a 3D model for face alignment, and [35] extracts multi-scale features based on detected landmark positions. In contrast, we only trained the PCANet using LFW-a [32], an aligned version of LFW images using the commercial alignment system of face.com.

## IV. EXPERIMENTS: HAND-WRITTEN DIGITS, TEXTURE AND OBJECT RECOGNITION

The proposed PCANet, RandNet and LDANet were extensively tested on face databases in the last section. We now turn our attention to their applications to hand-written digit recognition, texture discrimination, and object recognition in this section. Before proceeding further, we introduce the databases and classifiers used in this section as follows.

*Databases:* The MNIST [4] dataset consists of 28×28 gray-scale images of handwritten digits 0-9. MNIST variations [38] introduce more controllable factors of variation, such as rotations, background noise, and background images, to MNIST. The CUReT texture dataset [39] contains 61 classes of image textures with dimensions of 200×200. Each texture class has images of the same material with different poses, illumination conditions, specularities, shadowing and surface normal variations. CIFAR10 [40] is a set of natural RGB images of dimensions 32×32, and the images vary significantly not only in object position and object scale within each class but also in the colors and textures of these objects. Table VIII summarizes the databases.

*Classifiers:* A linear SVM classifier [41] is employed throughout this section. The NN and softmax classifiers are not chosen because the high-dimensionality nature of the PCANet features and thousands of training images make the entire course of training much longer. However, their performances may not be comparable to that of the linear SVM classifier; see Section IV-A3 for details.

### *A. Digit Recognition on MNIST Datasets*

The *MNIST basic* dataset is used to investigate the influence of the number of filters and different block overlap ratios on the proposed networks. We then compare the proposed networks with state-of-the-art methods on all of the MNIST datasets. A performance comparison among NN, linear SVM and softmax classifiers is also performed.

*1) Impact of the Number of Filters:* We vary the number of filters in the first stage $L_1$ from 2 to 12 for the one-stage networks. Regarding the two-stage networks, we set $L_2 = 8$ and change $L_1$ from 4 to 24. The filter size of the networks is $k_1 = k_2 = 7$, the block size is 7×7, and the overlapping region between blocks is half of the block size. Figure 8 shows

[10] The x-y coordinates (97, 111) and (156, 111) of the original images are aligned to (10,60) and (70,60), respectively, of the template.

[11] For a fair comparison, we only report the results of single descriptors. The best-known LFW result using the unsupervised setting is 88.57% [33], which is inferred from four different descriptors.

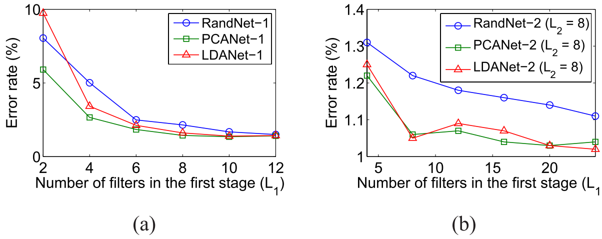


Fig. 8. Error rate of PCANet on MNIST basic test set for varying number of filters in the first stage. (a) PCANet-1; (b) PCANet-2 with $L_2 = 8$.

TABLE IX

ERROR RATES (%) OF PCANet-2 ON THE *basic* DATASET FOR VARYING BLOCK OVERLAP RATIOS (BORs)

| BOR | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 |
|---|---|---|---|---|---|---|---|
| RandNet-2 | 1.31 | 1.35 | 1.23 | 1.34 | 1.18 | 1.14 | 1.24 |
| PCANet-2 | **1.12** | **1.12** | **1.07** | 1.06 | 1.06 | **1.02** | **1.05** |
| LDANet-2 | 1.14 | 1.14 | 1.11 | **1.05** | **1.05** | 1.05 | 1.06 |

the results. The results are consistent with those on the MultiPIE face database shown in Figure 3; PCANet outperforms RandNet and LDANet for almost all cases.

*2) Impact of the Block Overlap Ratio:* The number of filters is fixed to $L_1 = L_2 = 8$, the filter size is again $k_1 = k_2 = 7$, and the block size is 7×7. We only alter the block overlap ratio (BOR) from 0.1 to 0.7. Table IX lists the results of RandNet-2, PCANet-2, and LDANet-2. Clearly, PCANet-2 and LDANet-2 achieve their minimum error rates for BOR equal to 0.5 and 0.6, respectively, and PCANet-2 performs the best under all conditions.

*3) Comparison With State of the Art:* We compare RandNet, PCANet, and LDANet with ConvNet [5], 2-stage ScatNet (ScatNet-2) [6], and other methods. In ScatNet, the number of scales and the number of orientations are set to 3 and 8, respectively. Regarding the parameters of PCANet, we set the filter size $k_1 = k_2 = 7$ and the number of PCA filters $L_1 = L_2 = 8$; the block size is tuned through cross-validation for *MNIST* and the validation sets for MNIST variations.[12] The overlapping region between blocks is half of the block size. Unless otherwise specified, we use the linear SVM classifier for ScatNet, RandNet, PCANet and LDANet for the 9 classification tasks. Furthermore, the softmax classifier for the PCANet is also tested because it has been widely used in the final layer of DNN/ConvNet.

The testing error rates of the various methods on *MNIST* are shown in Table X. For a fair comparison, we do not include the results of methods using augmented training samples with distortions or other information; for that, the best-known result is 0.23% [42]. We note that RandNet-2, PCANet-2, and LDANet-2 are comparable with the state-of-the-art methods on this standard MNIST task. However, because *MNIST* has a substantial amount of training data, all methods perform very well and are very similar – the difference is not statistically meaningful.

TABLE X

COMPARISON OF ERROR RATES (%) OF THE METHODS ON *MNIST*, EXCLUDING METHODS THAT AUGMENT THE TRAINING DATA. THE FILTER SIZE $k_1 = k_2 = 7$ AND THE NUMBER OF FILTERS $L_1 = L_2 = 8$ ARE SET, AND THE LINEAR SVM IS USED IN RandNet, PCANet, AND LDANet UNLESS OTHERWISE SPECIFIED. HERE, $L_1' = L_1 L_2$ AND $k_i' = 2k_i - 1,\ i = 1, 2$

| Methods | *MNIST* |
|---|---|
| HSC [44] | 0.77 |
| K-NN-SCM [45] | 0.63 |
| K-NN-IDM [46] | 0.54 |
| CDBN [7] | 0.82 |
| ConvNet [5] | 0.53 |
| Stochastic pooling ConvNet [47] | 0.47 |
| Conv. Maxout + Dropout [3] | 0.45 |
| ScatNet-2 (SVM$_{rbf}$) [6] | **0.43** |
| RandNet-1 | 1.32 |
| RandNet-2 | 0.63 |
| PCANet-1 | 0.94 |
| PCANet-2 | 0.66 |
| LDANet-1 | 0.98 |
| LDANet-2 | 0.62 |
| PCANet-1 ($L_1' = 64,\ k_1' = k_2' = 13$) | 0.62 |
| PCANet-2 (NN) | 1.36 |
| PCANet-2 (softmax) | 0.70 |

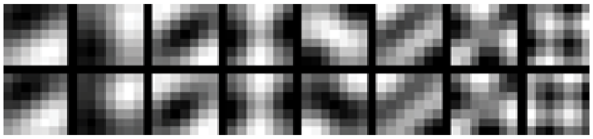


Fig. 9. The PCANet filters learned on the MNIST dataset. Top row: the first stage. Bottom row: the second stage.

Accordingly, we also report the results of different methods on MNIST variations in Table XI. To the best of our knowledge, PCANet-2 achieves state-of-the-art results for four out of the eight remaining tasks: *basic*, *bg-img*, *bg-img-rot*, and *convex*. In particular, for *bg-img*, the error rate decreases from 12.25 [43] to 10.95%.

Table X and Table XI also show the results obtained using PCANet-1 with $L_1' = L_1 L_2$ filters of size $k_1' \times k_2'$ for $k_i' = 2k_i - 1, i = 1, 2$. PCANet-1 with such a parameter setting is used to mimic the reported PCANet-2 in a single-stage structure. PCANet-2 continues to outperform this PCANet-1 alternative in most cases, repeatedly confirming the merits of a deeper network. In addition, one can also observe the performance of PCANet-2 with the NN and softmax classifiers in the last two rows, respectively. Both the NN and softmax classifiers do not provide a performance that is superior to the linear SVM classifier in most cases.

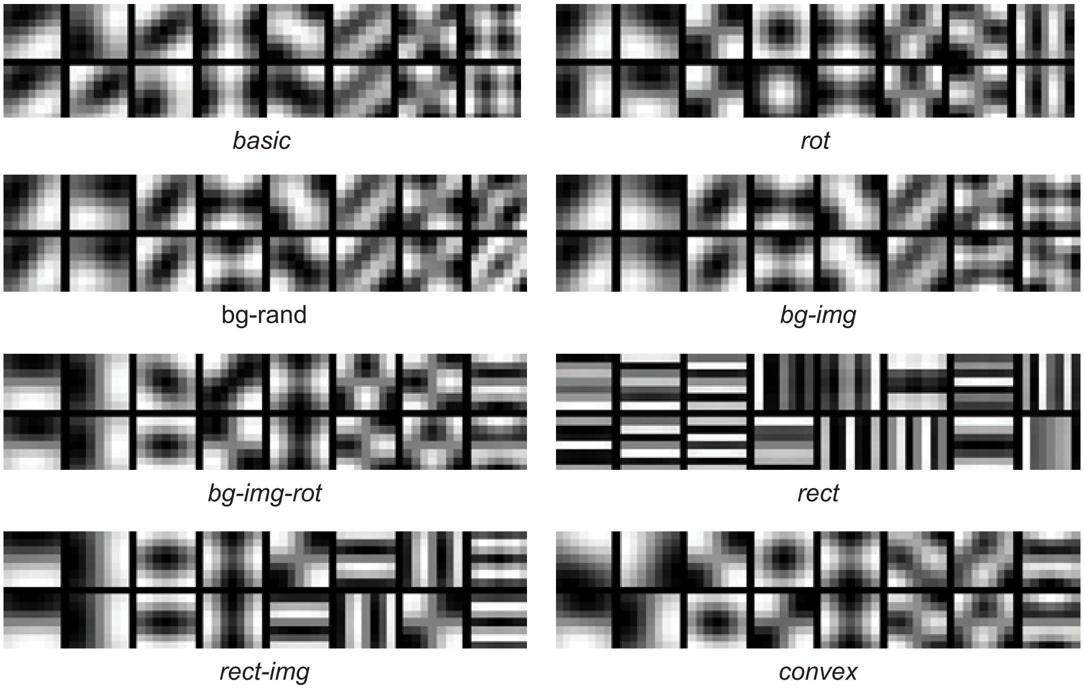
In addition, we present the learned PCANet filters in Figure 9 and Figure 10. An intriguing pattern is observed in the filters of the *rect* and *rect-img* datasets. For *rect*, we can see both horizontal and vertical stripes because these patterns attempt to capture the edges of the rectangles. When there is an image background in *rect-img*, several filters become low-pass to secure the responses from background images.

[12] Using either cross-validation or the validation set, the optimal block size is obtained as 7×7 for *MNIST*, *basic*, and *rec-img*; 4×4 for *rot*, *bg-img*, *bg-rnd*, and *bg-img-rot*; 14×14 for *rec*; and 28×28 for *convex*.

TABLE XI

COMPARISON OF TESTING ERROR RATES (%) FOR THE VARIOUS METHODS ON MNIST VARIATIONS. THE FILTER SIZE $k_1 = k_2 = 7$ AND THE NUMBER OF FILTERS $L_1 = L_2 = 8$ ARE SET, AND THE LINEAR SVM IS USED IN RandNet, PCANet, AND LDANet UNLESS OTHERWISE SPECIFIED. HERE, $L'_1 = L_1 L_2$ AND $k'_i = 2k_i - 1,\ i = 1, 2$

| Methods | *basic* | *rot* | *bg-rand* | *bg-img* | *bg-img-rot* | *rect* | *rect-img* | *convex* |
|---|---|---|---|---|---|---|---|---|
| CAE-2 [48] | 2.48 | 9.66 | 10.90 | 15.50 | 45.23 | 1.21 | 21.54 | - |
| TIRBM [49] | - | **4.20** | - | - | 35.50 | - | - | - |
| PGBM + DN-1 [43] | - | - | **6.08** | 12.25 | 36.76 | - | - | - |
| ScatNet-2 [6] | 1.27 | 7.48 | 12.30 | 18.40 | 50.48 | **0.01** | **8.02** | 6.50 |
| RandNet-1 | 1.86 | 14.25 | 18.81 | 15.97 | 51.82 | 0.21 | 15.94 | 6.78 |
| RandNet-2 | 1.25 | 8.47 | 13.47 | 11.65 | 43.69 | 0.09 | 17.00 | 5.45 |
| PCANet-1 | 1.44 | 10.55 | 6.77 | 11.11 | 42.03 | 0.15 | 25.55 | 5.93 |
| PCANet-2 | 1.06 | 7.37 | 6.19 | **10.95** | **35.48** | 0.24 | 14.08 | **4.36** |
| LDANet-1 | 1.61 | 11.40 | 7.16 | 13.03 | 43.86 | 0.15 | 23.63 | 6.89 |
| LDANet-2 | **1.05** | 7.52 | 6.81 | 12.42 | 38.54 | 0.14 | 16.20 | 7.22 |
| PCANet-1 ($L'_1 = 64,\ k'_1 = k'_2 = 13$) | 1.21 | 8.30 | 6.88 | 11.97 | 39.06 | 0.03 | 13.94 | 6.75 |
| PCANet-2 (NN) | 2.77 | 11.51 | 12.91 | 33.55 | 56.83 | 0.49 | 18.89 | 19.66 |
| PCANet-2 (softmax) | 1.40 | 8.52 | 6.85 | 11.55 | 35.86 | 0.49 | 13.39 | 4.19 |


*basic*

*rot*

bg-rand

*bg-img*

*bg-img-rot*

*rect*

*rect-img*

*convex*

Fig. 10. The PCANet filters learned on various MNIST datasets. For each dataset, the top row shows the filters of the first stage; the bottom row shows the filters of the second stage.

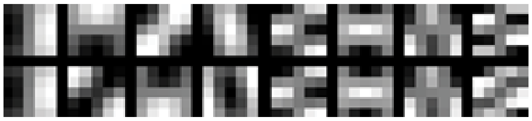


Fig. 11. The PCANet filters learned on the CUReT database. Top row: the first stage. Bottom row: the second stage.

### *B. Texture Classification on CUReT Dataset*

In this experiment, a subset of the dataset with azimuthal viewing angle of less than 60 degrees is selected, therein yielding 92 images in each class. A central 200 × 200 region is cropped from each of the selected images. The dataset is randomly split into a training and testing set, with 46 training images for each class, as in [39]. The PCANet is trained with filter size $k_1 = k_2 = 5$, number of filters $L_1 = L_2 = 8$, and block size 50×50. We use the linear SVM classifier. The testing error rates averaged over 10 random splits are shown in Table XII. We see that PCANet-1 outperforms ScatNet-1, but the improvement from PCANet-1 to PCANet-2 is not as large as that of ScatNet. Note that ScatNet-2 followed by a PCA-based classifier provides the best result [6].

### *C. Object Recognition on CIFAR10*

Finally, we evaluate the performance of the PCANet on the CIFAR10 database for object recognition. The motivation here is to explore the limitation of such a simple PCANet on a relatively complex database compared to the databases composed of faces, digits, and textures that we have experimented with, which could somehow be roughly aligned or prepared. First, we extend PCA filter learning so as to accommodate the RGB images in object databases. In the same spirit of constructing the data matrix $\boldsymbol{X}$ in (1), we gather the same individual matrix for RGB channels of the images, denoted by $\boldsymbol{X}_{\mathrm{r}},\ \boldsymbol{X}_{\mathrm{g}},\ \boldsymbol{X}_{\mathrm{b}} \in \mathbb{R}^{k_1 k_2 \times N\tilde{m}\tilde{n}}$. Following the key steps

TABLE XII

COMPARISON OF ERROR RATES (%) ON CUReT

| Methods | Error rates |
|---|---|
| Textons [50] | 1.50 |
| BIF [51] | 1.40 |
| Histogram [52] | 1.00 |
| ScatNet-1 (PCA) [6] | 0.50 |
| ScatNet-2 (PCA) [6] | **0.20** |
| RandNet-1 | 0.61 |
| RandNet-2 | 0.46 |
| PCANet-1 | 0.45 |
| PCANet-2 | 0.39 |
| LDANet-1 | 0.69 |
| LDANet-2 | 0.54 |

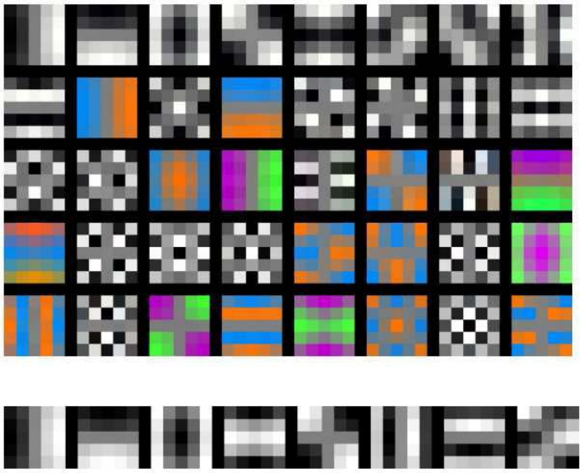


Fig. 12. The PCANet filters learned on the Cifar10 database. Top: the first stage. Bottom: the second stage.

TABLE XIII

COMPARISON OF ACCURACY (%) OF THE METHODS ON THE CIFAR10 DATASET WITH NO DATA AUGMENTATION

| Methods | Accuracy |
|---|---|
| Tiled CNN [57] | 73.10 |
| Improved LCC [58] | 74.50 |
| KDES-A [59] | 76.00 |
| K-means (Triangle, 4000 features) [60] | 79.60 |
| Cuda-convnet2 [61] | 82.00 |
| Stochastic pooling ConvNet [47] | 84.87 |
| CNN + Spearmint [62] | 85.02 |
| Conv. Maxout + Dropout [3] | 88.32 |
| NIN [63] | 89.59 |
| PCANet-2 | 77.14 |
| PCANet-2 (combined) | 78.67 |

in Section II-A1, the multichannel PCA filters can be easily verified as

$$\boldsymbol{W}_l^{\mathrm{r,g,b}} \doteq \mathrm{mat}_{k_1,k_2,3}(\boldsymbol{q}_l(\widetilde{\boldsymbol{X}}\widetilde{\boldsymbol{X}}^T)) \in \mathbb{R}^{k_1 \times k_2 \times 3}, \qquad (14)$$

where $\widetilde{\boldsymbol{X}} = [\boldsymbol{X}_{\mathrm{r}}^T, \boldsymbol{X}_{\mathrm{g}}^T, \boldsymbol{X}_{\mathrm{b}}^T]^T$ and $\mathrm{mat}_{k_1,k_2,3}(\boldsymbol{v})$ is a function that maps $\boldsymbol{v} \in \mathbb{R}^{3k_1k_2}$ to a tensor $\boldsymbol{W} \in \mathbb{R}^{k_1 \times k_2 \times 3}$. An example of the learned multichannel PCA filters is demonstrated in Figure 12. In addition to the modification above, we also connect spatial pyramid pooling (SPP) [53]–[56] to the output layer of PCANet, with the objective of extracting information invariant to large poses and complex backgrounds, usually seen in object databases. Another advantage of equipping SPP with the PCANet is that it can generate a fixed-length representation regardless of image size/scale. This strategy has been explored in [55] and [56], where the SPP-net clearly increases the accuracy of various no-SPP counterparts. We also observe that the PCANet with SPP essentially improves the accuracy of object recognition on CIFAR10! However, no such significant improvement was found in the previous experiments on images of faces, digits and textures.

We use the linear SVM classifier in the experiments. In the first experiment, we train the PCANet on CIFAR10 with a filter size $k_1 = k_2 = 5$, number of filters $L_1 = 40$, $L_2 = 8$, and block size equal to $8 \times 8$. In addition, we set the overlapping region between blocks to half of the block size and connected SPP to the output layer of the PCANet, i.e., the maximum response in each bin of the block histograms is pooled in a pyramid of 4×4, 2×2, and 1×1 subregions. This yields the 21 pooled histogram features of dimension $L_1 2^{L_2}$. The dimension of each pooled feature is reduced to 1280 by PCA.

In the second experiment, we concatenate PCANet features learned with different filter sizes of $k_1 = k_2 = 3$ and $k_1 = k_2 = 5$. All of the processes and model parameters are fixed to be identical to the single descriptor mentioned in the last paragraph, except $L_1 = 12$ and $L_1 = 28$ are set to make the filter size equal to 3 and 5, respectively. This is performed to ensure that the combined features are of the same dimension as the single descriptor for fairness.

The results are shown in Table XIII. PCANet-2 achieves an accuracy of 77.14% and obtains a 1.5% improvement when combining two features learned with different filter sizes (marked with "combined" in parenthesis). Although PCANet-2 experiences an approximately 11% accuracy degradation in comparison to the state-of-the-art method (with no data augmentation), the performance of the fully unsupervised and extremely simple PCANet-2 shown here remains encouraging.

## V. CONCLUSION

In this paper, we proposed arguably the simplest unsupervised convolutional deep learning network— PCANet. The network processes input images using cascaded PCA, binary hashing, and block histograms. Like most ConvNet models, the network parameters, such as the number of layers, the filter size, and the number of filters, must be given to the PCANet. Once the parameters are fixed, training the PCANet is extremely simple and efficient because the filter learning in the PCANet does not involve regularized parameters or require numerical optimization solvers. Moreover, building the PCANet consists of only a cascaded linear map followed by a nonlinear output stage. Such simplicity offers an alternative yet refreshing perspective on convolutional deep learning networks and could further facilitate mathematical analysis and justification of their effectiveness.

Two simple extensions of PCANet, i.e., RandNet and LDANet, have been introduced and tested together with PCANet on many image classification tasks, including faces, hand-written digits, textures, and objects. The extensive experimental results have consistently shown that the PCANet outperforms RandNet and LDANet and that it is generally on par with ScatNet and variations of ConvNet. Furthermore, the performance of the PCANet is very similar and often better than highly engineered hand-crafted features (such as LBP and LQP). In tasks such as face recognition, the PCANet also demonstrates remarkable robustness to corruption and the ability to transfer to new datasets.

The experiments also demonstrate that as long as the images in the databases are somehow well prepared, i.e., the images are roughly aligned and do not exhibit diverse scales or poses, the PCANet is able to eliminate the image variability and provides reasonably competitive accuracy. In challenging image databases such as PASCAL VOC [64] and ImageNet [65], the PCANet might not be sufficient to address the variability given its extremely simple structure and unsupervised learning method. An intriguing research direction will then be how to construct a more complicated (e.g., more sophisticated filters possibly with discriminative learning) or deeper (greater number of stages) PCANet that could address the aforementioned issues. Some preprocessing of pose alignment and scale normalization might be essential to guaranteeing good performance. The current bottleneck that keeps PCANet from becoming deeper (e.g., to more than two stages) is that the dimension of the resulting feature would increase exponentially with the number of stages. This fortunately seems able to be fixed by replacing the 2D convolution filters with tensor-like filters, as in (14); this will be the topic of future study. Furthermore, we will also leave as future work augmenting PCANet with a simple, scalable baseline classifier, readily applicable to much larger scale datasets and problems.

Regardless, the extensive experiments given in this paper sufficiently demonstrate two facts: 1) the PCANet is a very simple deep learning network that effectively extracts useful information for the classification of faces, digits, and texture images, and 2) the PCANet can be a valuable baseline for studying advanced deep learning architectures for large-scale image classification tasks.

## References

[1] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[2] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[3] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, "Maxout networks," in *Proc. 30th ICML*, 2013, pp. 1–9.

[4] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[5] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *Proc. IEEE 12th ICCV*, Sep./Oct. 2009, pp. 2146–2153.

[6] J. Bruna and S. Mallat, "Invariant scattering convolution networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1872–1886, Aug. 2013.

[7] H. Lee, R. Grosse, R. Rananth, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proc. 26th Annu. ICML*, 2009, pp. 609–616.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural network," in *Proc. NIPS*, 2012, pp. 1097–1105.

[9] K. Kavukcuoglu, P. Sermanet, Y. Boureau, K. Gregor, M. Mathieu, and Y. LeCun, "Learning convolutional feature hierarchies for visual recognition," in *Proc. NIPS*, 2010, pp. 1090–1098.

[10] L. Sifre and S. Mallat, "Rotation, scaling and deformation invariant scattering for texture discrimination," in *Proc. IEEE Conf. CVPR*, Jun. 2013, pp. 1233–1240.

[11] C. J. C. Burges, J. C. Platt, and S. Jana, "Distortion discriminant analysis for audio fingerprinting," *IEEE Trans. Speech Audio Process.*, vol. 11, no. 3, pp. 165–174, May 2003.

[12] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[13] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. CVPR*, Jun. 2005, pp. 886–893.

[14] L. Fei-Fei and P. Perona, "A Bayesian hierarchical model for learning natural scene categories," in *Proc. IEEE Comput. Soc. Conf. CVPR*, Jun. 2005, pp. 524–531.

[15] C. Liu and H. Wechsler, "Gabor feature based classification using the enhanced Fisher linear discriminant model for face recognition," *IEEE Trans. Image Process.*, vol. 11, no. 4, pp. 467–476, Apr. 2002.

[16] H. Yu and J. Yang, "A direct LDA algorithm for high-dimensional data—With application to face recognition," *Pattern Recognit.*, vol. 34, no. 10, pp. 2067–2069, 2001.

[17] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-PIE," in *Proc. 8th IEEE Conf. Autom. Face Gesture Recognit.*, Sep. 2008, pp. 1–8.

[18] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 6, pp. 643–660, Jun. 2001.

[19] A. Martínez and R. Benavente, "The AR face database," CVC, Barcelona, Spain, Tech. Rep. #24, 1998.

[20] P. J. Phillips, H. Wechsler, J. Huang, and P. J. Rauss, "The FERET database and evaluation procedure for face-recognition algorithms," *Image Vis. Comput.*, vol. 16, no. 5, pp. 295–306, 1998.

[21] 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," Univ. Massachusetts, Amherst, Amherst, MA, USA, Tech. Rep. 07-49, 2007.

[22] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face description with local binary patterns: Application to face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 12, pp. 2037–2041, Dec. 2006.

[23] Y. Jia. (2013). *Caffe: An Open Source Convolutional Architecture for Fast Feature Embedding*. [Online]. Available: http://caffe.berkeleyvision.org/

[24] X. Tan and B. Triggs, "Enhanced local texture feature sets for face recognition under difficult lighting conditions," *IEEE Trans. Image Process.*, vol. 19, no. 6, pp. 1635–1650, Jun. 2010.

[25] W. Deng, J. Hu, and J. Guo, "Extended SRC: Undersampled face recognition via intraclass variant dictionary," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 9, pp. 1864–1870, Sep. 2012.

[26] J. Lu, Y.-P. Tan, and G. Wang, "Discriminative multimanifold analysis for face recognition from a single training sample per person," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 39–51, Jan. 2013.

[27] N.-S. Vu and A. Caplier, "Enhanced patterns of oriented edge magnitudes for face recognition and image matching," *IEEE Trans. Image Process.*, vol. 21, no. 3, pp. 1352–1368, Mar. 2012.

[28] S. U. Hussain, T. Napoléon, and F. Jurie, "Face recognition using local quantized patterns," in *Proc. BMVC*, 2012, p. 11.

[29] S. Xie, S. Shan, X. Chen, and J. Chen, "Fusing local patterns of Gabor magnitude and phase for face recognition," *IEEE Trans. Image Process.*, vol. 19, no. 5, pp. 1349–1361, May 2010.

[30] N.-S. Vu, "Exploring patterns of gradient orientations and magnitudes for face recognition," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 2, pp. 295–304, Feb. 2013.

[31] Z. Chai, Z. Sun, H. Méndez-Vázquez, R. He, and T. Tan, "Gabor ordinal measures for face recognition," *IEEE Trans. Inf. Forensics Security*, vol. 9, no. 1, pp. 14–26, Jan. 2014.

[32] L. Wolf, T. Hassner, and Y. Taigman, "Effective unconstrained face recognition by combining multiple descriptors and learned background statistics," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 10, pp. 1978–1990, Oct. 2011.

[33] O. Barkan, J. Weill, L. Wolf, and H. Aronowitz, "Fast high dimensional vector multiplication face recognition," in *Proc. IEEE ICCV*, Dec. 2013, pp. 1960–1967.

[34] Y. Taigman, M. Yang, M. A. Ranzato, and L. Wolf, "DeepFace: Closing the gap to human-level performance in face verification," in *Proc. IEEE Conf. CVPR*, Jun. 2014, pp. 1701–1708.

[35] H. Fan, Z. Cao, Y. Jiang, and Q. Yin. (2014). "Learning deep face representation." [Online]. Available: http://arxiv.org/abs/1403.2802

[36] D. Chen, X. Cao, F. Wen, and J. Sun, "Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification," in *Proc. IEEE Conf. CVPR*, Jun. 2013, pp. 3025–3032.

[37] 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 *Proc. IEEE CVPR*, Jun. 2013, pp. 3554–3561.

[38] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation," in *Proc. 24th ICML*, 2007, pp. 473–480.

[39] M. Varma and A. Zisserman, "A statistical approach to material classification using image patch exemplars," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 11, pp. 2032–2047, Nov. 2009.

[40] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009.

[41] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *J. Mach. Learn. Res.*, vol. 9, pp. 1871–1874, Jul. 2008.

[42] D. Ciresan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Proc. IEEE Conf. CVPR*, Jun. 2012, pp. 3642–3649.

[43] K. Sohn, G. Zhou, C. Lee, and H. Lee, "Learning and selecting features jointly with point-wise gated Boltzmann machines," in *Proc. 30th ICML*, 2013, pp. 217–225.

[44] K. Yu, Y. Lin, and J. Lafferty, "Learning image representations from the pixel level via hierarchical sparse coding," in *Proc. IEEE Conf. CVPR*, Jun. 2011, pp. 1713–1720.

[45] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 4, pp. 509–522, Apr. 2002.

[46] D. Keysers, T. Deselaers, C. Gollan, and H. Ney, "Deformation models for image recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 8, pp. 1422–1435, Aug. 2007.

[47] M. D. Zeiler and R. Fergus, "Stochastic pooling for regularization of deep convolutional neural networks," in *Proc. ICLR*, 2013.

[48] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, "Contractive auto-encoders: Explicit invariance during feature extraction," in *Proc. 28th ICML*, 2011, pp. 1–8.

[49] K. Sohn and H. Lee, "Learning invariant representations with local transformations," in *Proc. 29th ICML*, 2012, pp. 1311–1318.

[50] E. Hayman, B. Caputo, M. Fritz, and J.-O. Eklundh, "On the significance of real-world conditions for material classification," in *Proc. 8th ECCV*, 2004, pp. 253–266.

[51] M. Crosier and L. D. Griffin, "Using basic image features for texture classification," *Int. J. Comput. Vis.*, vol. 88, no. 3, pp. 447–460, 2010.

[52] R. E. Broadhurst, "Statistical estimation of histogram variation for texture classification," in *Proc. Workshop Texture Anal. Synth.*, 2006, pp. 1–6.

[53] K. Grauman and T. Darrell, "The pyramid match kernel: Discriminative classification with sets of image features," in *Proc. 10th IEEE ICCV*, Oct. 2005, pp. 1458–1465.

[54] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proc. IEEE Comput. Soc. Conf. CVPR*, Jun. 2006, pp. 2169–2178.

[55] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *Proc. 13th ECCV*, 2014, pp. 346–361.

[56] K. He, X. Zhang, S. Ren, and J. Sun. (2015). "Spatial pyramid pooling in deep convolutional networks for visual recognition." [Online]. Available: http://arxiv.org/abs/1406.4729

[57] Q. V. Le, J. Ngiam, Z. Chen, D. Chia, P. W. Koh, and A. Y. Ng, "Tiled convolutional neural networks," in *Proc. NIPS*, 2010, pp. 1279–1287.

[58] K. Yu and T. Zhang, "Improved local coordinate coding using local tangents," in *Proc. 27th ICML*, 2010, pp. 1215–1222.

[59] L. Bo, X. Ren, and D. Fox, "Kernel descriptors for visual recognition," in *Proc. NIPS*, 2010, pp. 244–252.

[60] A. Coates, H. Lee, and A. Y. Ng, "An analysis of single-layer networks in unsupervised feature learning," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2001, pp. 215–223.

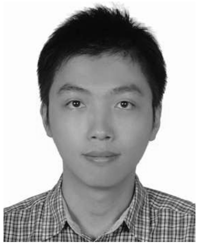
[61] A. Krizhevsky. (Jul. 18, 2014). *Cuda-Convnet*. [Online]. Available: http://code.google.com/p/cuda-convnet/

[62] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Proc. NIPS*, 2012, pp. 2951–2959.

[63] M. Lin, Q. Chen, and S. Yan. (2014). "Network in network." [Online]. Available: http://arxiv.org/abs/1312.4400

[64] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Jun. 2010.

[65] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," 2014.



**Tsung-Han Chan** (S'08–M'09) received the B.S. degree from the Department of Electrical Engineering, Yuan Ze University, Taiwan, in 2004, and the Ph.D. degree from the Institute of Communications Engineering, National Tsing Hua University, Taiwan, in 2009. He is currently a Senior Engineer with MediaTek Inc., Hsinchu, Taiwan. He was a co-recipient of the WHISPERS 2011 Best Paper Award. He was also recognized as an Outstanding Reviewer for CVPR 2014, and a Best Reviewer of the IEEE Transactions on Geoscience and Remote Sensing 2014. His research interests are in image processing and numerical optimization, with a recent emphasis on computer vision, machine learning, and hyperspectral remote sensing.

**Kui Jia** received the B.Eng. degree in marine engineering from Northwestern Polytechnic University, China, in 2001, the M.Eng. degree in electrical and computer engineering from the National University of Singapore in 2003, and the Ph.D. degree in computer science from the Queen Mary University of London, London, U.K., in 2007. He is currently a Visiting Assistant Professor with the University of Macau, Macau, China. His research interests are in computer vision, machine learning, and image processing.

**Shenghua Gao** received the B.E. degree from the University of Science and Technology of China, in 2008 (outstanding graduates), and the Ph.D. degree from Nanyang Technological University, in 2012. From 2012 to 2014, he was a Research Scientist with the Advanced Digital Sciences Center, Singapore. In 2015, he visited UC Berkeley as a Visiting Scholar. He is currently an Assistant Professor with ShanghaiTech University, China. He has authored more than 20 papers in object and face recognition related topics in many international conferences and journals, including the IEEE Transactions on Pattern Analysis and Machine Intelligence, the *International Journal of Computer Vision*, the IEEE Transactions on Image Processing, the IEEE Transactions on Neural Networks and Learning Systems, the IEEE Transactions on Multimedia, the IEEE Transactions on Circuits and Systems for Video Technology, the Conference on Computer Vision and Pattern Recognition, and the European Conference on Computer Vision. His research interests include computer vision and machine learning. He received the Microsoft Research Fellowship in 2010, and the ACM Shanghai Young Research Scientist Award in 2015.

**Jiwen Lu** (M'11–SM'15) received the B.Eng. degree in mechanical engineering and the M.Eng. degree in electrical engineering from the Xi'an University of Technology, Xi'an, China, and the Ph.D. degree in electrical engineering from Nanyang Technological University, Singapore.

He is currently a Faculty Member with the Department of Automation, Tsinghua University, China. His current research interests include computer vision, pattern recognition, and machine learning. He has authored or co-authored over 110 scientific papers in these areas, in which more than 40 papers are published in the IEEE Transactions journals and top-tier computer vision conferences. He serves as an Associate Editor of *Pattern Recognition Letters*, *Neurocomputing*, and the IEEE Biometrics Council Newsletters.

Dr. Lu was a recipient of the First-Prize National Scholarship and the National Outstanding Student Award from the Ministry of Education of China in 2002 and 2003, the Best Student Paper Award from Pattern Recognition and Machine Intelligence Association of Singapore in 2012, the Top 10% Best Paper Award from the IEEE International Workshop on Multimedia Signal Processing in 2014, and the National 1000 Young Talents Plan Program in 2015.

**Zinan Zeng** received the B.E. (Hons.) and master's degrees from the School of Computer Engineering, Nanyang Technological University, Singapore. He joined the Advanced Digital Sciences Center, Singapore, as a Senior Software Engineer. His research interests include statistical learning and optimization with application to computer vision and image processing.

**Yi Ma** (F'13) received the bachelor's degrees in automation and applied mathematics from Tsinghua University, Beijing, in 1995, and the M.S. degree in electrical engineering and computer science, the M.A. degree in mathematics, and the Ph.D. degree in electrical engineering and computer science from the University of California, Berkeley, in 1997, 2000, and 2000, in 1997, 2000, and 2000. From 2000 to 2011, he was an Associate Professor of the Electrical and Computer Engineering Department, University of Illinois at Urbana–Champaign. From 2009 to 2014, he was a Principal Researcher and the Group Manager of the Visual Computing Group with Microsoft Research Asia. He is currently a Professor and the Executive Dean of the School of Information Science and Technology with ShanghaiTech University, China. His main research interests include computer vision and data science, and has written two textbooks entitled *An Invitation to 3-D Vision* (Springer) and *Generalized Principal Component Analysis* (Springer). He has served as an Associate Editor of the International Journal on Computer Vision, the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, the IEEE TRANSACTIONS ON INFORMATION THEORY, the *IEEE Signal Processing Magazine*, the *SIAM Journal on Imaging Sciences*, and the *IMA Journal on Information and Inference*. He was a recipient of the David Marr Best Paper Prize at the International Conference on Computer Vision in 1999 and the Honorable Mention for the Longuet-Higgins Best Paper Award at the European Conference on Computer Vision in 2004. He received the CAREER Award from the U.S. National Science Foundation in 2004 and the Young Investigator Program Award from the U.S. Office of Naval Research in 2005. He has served as the Area Chair of NIPS, CVPR, and ICCV, the Program Chair of ICCV 2013, and the General Chair of ICCV 2015.