Applied Computational Intelligence and Soft Computing

Machine Learning and Visual Computing

Guest Editors: Lei Zhang, Yu Cao, Fei Yang, and Qiushi Zhao

Machine Learning and Visual Computing

Machine Learning and Visual Computing

Guest Editors: Lei Zhang, Yu Cao, Fei Yang, and Qiushi Zhao

Editorial Board

Contents

Hindawi
Applied Computational Intelligence and Soft Computing
Volume 2017, Article ID 7571043, 1 page
http://dx.doi.org/10.1155/2017/7571043

Editorial
Machine Learning and Visual Computing

Lei Zhang,[1] Yu Cao,[2] Fei Yang,[3] and Qiushi Zhao[4]

[1]*Temple University, Philadelphia, PA, USA*
[2]*Medical Sieve Radiology Grand Challenge, IBM Research, San Jose, CA, USA*
[3]*Shandong University, Weihai, China*
[4]*Harbin University of Science and Technology, Harbin, China*

Correspondence should be addressed to Lei Zhang; cszhanglei@gmail.com

Received 29 January 2017; Accepted 29 January 2017; Published 19 March 2017

With the fast development of information science, information contained in big data has raised the interest of researchers from many different disciplines. Extracting and exploring the information from big datasets are essential to applying computational intelligence and soft computing to natural and social sciences. Recent advances of machine learning (especially, deep learning) make it possible to gain insight into big data and extract meaningful information, which has accelerated the progression of computational intelligence. Computer vision techniques contribute to understanding image and high-dimensional data from the real world to produce numerical or symbolic information. Visualization methods provide various ways to demonstrate information from complex datasets. Both computer vision techniques and visualization methods can be utilized to visually demonstrate the extracted information from datasets.

This special issue is dedicated to latest developments in machine learning and visual computing. Five articles from researchers around the world contribute to further steps into the theories and applications of machine learning and visual computing. The special issue covers a widespread research topics, from theoretical investigations to real-world applications, and comprises articles in active research areas like face classification, action recognition, and deep learning.

In the applications of computer vision techniques, M. Ngadi et al. introduce a practical face detection method by using the Local Binary Patterns (LBP) for the feature extraction and a novel Neighboring Support Vector Classifier (NSVC) for classification. The experimental results on different natural images show that the proposed method can get promising results within a short detection time. G. Zhang et al. provide a novel person reidentification method which leverages the clothing features in real-life videos.

Toward improving machine learning algorithms based on theoretical finding, B. Zu et al. propose a Low-Rank Kernel-based Semi-supervised Discriminant Analysis (LRKSDA). Extensive experiments on public databases show that LRKSDA can outperform other related Kernel Semi-supervised Discriminant Analysis methods. X. Li et al. propose an action recognition method by mining key skeleton poses with latent support vector machine (latent SVM). The detailed experimental results on three benchmark action datasets demonstrate that the proposed approach achieves superior performance to the state-of-the-art skeleton-based action recognition methods.

This special issue includes one review paper. The work by D. Xie et al. provides a survey of deep learning. The authors not only reviewed typical deep learning algorithms in computer vision and signal processing but also provided detailed information on how to apply different deep learning techniques to specific areas such as road crack detection and fault diagnosis.

Acknowledgments

We would like to thank all the authors and reviewers for their great contributions to this special issue.

Lei Zhang
Yu Cao
Fei Yang
Qiushi Zhao

Hindawi
Applied Computational Intelligence and Soft Computing
Volume 2017, Article ID 1320780, 13 pages
http://dx.doi.org/10.1155/2017/1320780

Review Article

Deep Learning in Visual Computing and Signal Processing

Danfeng Xie, Lei Zhang, and Li Bai

Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA 19121, USA

Correspondence should be addressed to Danfeng Xie; danfeng.xie@temple.edu

Received 21 October 2016; Revised 15 December 2016; Accepted 15 January 2017; Published 19 February 2017

Academic Editor: Francesco Carlo Morabito

Deep learning is a subfield of machine learning, which aims to learn a hierarchy of features from input data. Nowadays, researchers have intensively investigated deep learning algorithms for solving challenging problems in many areas such as image classification, speech recognition, signal processing, and natural language processing. In this study, we not only review typical deep learning algorithms in computer vision and signal processing but also provide detailed information on how to apply deep learning to specific areas such as road crack detection, fault diagnosis, and human activity detection. Besides, this study also discusses the challenges of designing and training deep neural networks.

1. Introduction

Deep learning methods are a group of machine learning methods that can learn features hierarchically from lower level to higher level by building a deep architecture. The deep learning methods have the ability to automatically learn features at multiple levels, which makes the system be able to learn complex mapping function $f : X \rightarrow Y$ directly from data, without help of the human-crafted features. This ability is crucial for high-level feature abstraction since high-level features are difficult to be described directly from raw training data. Moreover, with the sharp growth of data, the ability to learn high-level features automatically will be even more important.

The most characterizing feature of deep learning methods is that their models all have deep architectures. A deep architecture means it has multiple hidden layers in the network. In contrast, a shallow architecture has only few hidden layers (1 to 2 layers). Deep architectures are loosely inspired by mammal brain. When given an input percept, mammal brain processes it using different area of cortex which abstracts different levels of features. Researchers usually describe such concepts in hierarchical ways, with many levels of abstraction. Furthermore, mammal brains also seem to process information through many stages of transformation and representation. A very clear example is that the information in the primate visual system is processed in a sequence of stages: edge detection, primitive shapes, and more complex visual shapes.

Inspired by the deep architecture of mammal brain, researchers investigated deep neural networks for two decades but did not find effective training methods before 2006: researchers only obtained good experimental results of neural network with one or two hidden layers but could not get good results of neural network with more hidden layers. In 2006, Hinton et al. proposed deep belief networks (DBNs) [1], with a learning algorithm that uses unsupervised learning algorithm to greedily train deep neural network layer by layer. This training method, which is called deep learning, turns out to be very effective and efficient in training deep neural networks.

Many other deep architectures, that is, autoencoder, deep convolutional neural networks, and recurrent neural networks, are successfully applied in various areas. Regression [2], classification [3–9], dimensionality reduction [10, 11], modeling motion [12, 13], modeling textures [14], information retrieval [15–17], natural language processing [18–20], robotics [21], fault diagnosis [22], and road crack detection [23] have seen increasing deep learning-related research studies. There are mainly three crucial reasons for the rapid development of deep learning applications nowadays: the big leap of deep learning algorithms, the significantly increased computational abilities, and the sharp drop of price in hardware.

This survey provides an overview of several deep learning algorithms and their emerging applications in several specific areas, featuring face recognition, road crack detection, fault diagnosis, and falls detection. As complementarity to existing review papers [24, 25], we not only review the state-of-the-art deep learning methods but also provide detailed information on how to apply deep learning to specific problems. The reminder of this paper is organized as follows. In Section 2, the two categories of deep learning algorithms are introduced: restricted Boltzmann machines (RBMs) and convolutional neural networks (CNNs). The training strategies are discussed in Section 3. In Section 4, we describe several specific deep learning applications, that is, face recognition, road crack detection, fault diagnosis, and human activity detection. In Section 5, we discuss several challenges of training and using the deep neural networks. In Section 6, we conclude the paper.

2. Deep Learning Algorithms

Deep learning algorithms have been extensively studied in recent years. As a consequence, there are a large number of related approaches. Generally speaking, these algorithms can be grouped into two categories based on their architectures: restricted Boltzmann machines (RBMs) and convolutional neural networks (CNNs). In the following sections, we will briefly review these deep learning methods and their developments.

2.1. Deep Neural Network. This section introduces how to build and train RBM-based deep neural networks (DNNs). The building and training procedures of a DNN contain two steps. First, build a deep belief network (DBN) by stacking restricted Boltzmann machines (RBMs) and feed unlabeled data to pretrain the DBN. The pretrained DBN provides initial parameters for the deep neural network. In the second step, labeled data is fed to train the DNN using back-propagation. After two steps of training, a trained DNN is obtained. This section is organized as follows. Section 2.1.1 introduces RBM, which is the basic component of DBN. In Section 2.1.2, RBM-based DNN is introduced.

2.1.1. Restricted Boltzmann Machines. RBM is an energy-based probabilistic generative model [26–29]. It is composed of one layer of visible units and one layer of hidden units. The visible units represent the input vector of a data sample and the hidden units represent features that are abstracted from the visible units. Every visible unit is connected to every hidden unit, whereas no connection exists within the visible layer or hidden layer. Figure 1 illustrates the graphical model of restricted Boltzmann machine.

As a result of the lack of hidden-hidden and input-input interactions, the energy function of a RBM is

$$\text{Energy}(\mathbf{v}, \mathbf{h}; \theta) = -\mathbf{b}^T\mathbf{v} - \mathbf{c}^T\mathbf{h} - \mathbf{h}^T\mathbf{W}\mathbf{v}, \tag{1}$$

where $\theta = \{\mathbf{W}, \mathbf{b}, \mathbf{c}\}$ are the parameters of RBM and they need to be learned during the training procedure; $\mathbf{W}$ denotes the weights between the visible layer and hidden layer; $\mathbf{b}$ and $\mathbf{c}$ are the bias of the visible layer and hidden layer, respectively; this model is called binary RBM because the vectors $\mathbf{v}$ and $\mathbf{h}$ only contain binary values (0 or 1).

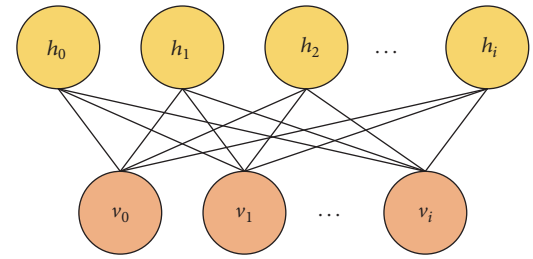


Figure 1: Restricted Boltzmann machine.

We can obtain a tractable expression for the conditional probability $P(h \mid v)$ [30]:

$$\begin{aligned} P(h \mid v) &= \frac{\exp\left(\mathbf{b}^T\mathbf{v} + \mathbf{c}^T\mathbf{h} + \mathbf{h}^T\mathbf{W}\mathbf{v}\right)}{\sum_{\tilde{\mathbf{h}}} \exp\left(\mathbf{b}^T\mathbf{v} + \mathbf{c}^T\tilde{\mathbf{h}} + \tilde{\mathbf{h}}^T\mathbf{W}\mathbf{v}\right)} \\ &= \frac{\prod_i \exp\left(\mathbf{c}_i\mathbf{h}_i + \mathbf{h}_i\mathbf{W}_i\mathbf{v}\right)}{\prod_i \sum_{\tilde{\mathbf{h}}_i} \exp\left(\mathbf{c}_i\tilde{\mathbf{h}}_i + \tilde{\mathbf{h}}_i\mathbf{W}_i\mathbf{v}\right)} \\ &= \prod_i \frac{\exp\left(\mathbf{h}_i\left(\mathbf{c}_i + \mathbf{W}_i\mathbf{v}\right)\right)}{\sum_{\tilde{\mathbf{h}}_i} \exp\left(\tilde{\mathbf{h}}_i\left(\mathbf{c}_i + \mathbf{W}_i\mathbf{v}\right)\right)} = \prod_i P\left(\mathbf{h}_i \mid v\right). \end{aligned} \tag{2}$$

For binary RBM, where $h_i \in \{0, 1\}$, the equation for a hidden unit's output given its input is

$$P(h_i = 1 \mid v) = \frac{e^{c_i + W_i v}}{1 + e^{c_i + W_i v}} = \text{sigm}\left(c_i + W_i v\right). \tag{3}$$

Because v and h play a symmetric role in the energy function, the following equation can be derived:

$$P(v \mid h) = \prod_i P\left(v_i \mid h\right), \tag{4}$$

and for the visible unit $v_j \in \{0, 1\}$, we have

$$P\left(v_j = 1 \mid h\right) = \text{sigm}\left(b_j + W_{\cdot j}^T h\right), \tag{5}$$

where $W_{\cdot j}$ is the jth column of W.

Although binary RBMs can achieve good performance when dealing with discrete inputs, they have limitations to handle continuous-valued inputs due to their structure. Thus, in order to achieve better performance on continuous-valued inputs, Gaussian RBMs are utilized for the visible layer [4, 31]. The energy function of a Gaussian RBM is

$$\text{Energy}(\mathbf{v}, \mathbf{h}) = \sum_i \frac{\left(v_i - a_i\right)^2}{2\sigma_i^2} - \sum_{ij} w_{ij} h_j \frac{v_i}{\sigma_i} - \sum_j c_j h_j, \tag{6}$$

where a_i and σ_i are the mean and the standard deviation of visible unit i. Note here that only the visible layer v is continuous-valued and hidden layer h is still binary. In practical situation, the input data is normalized, which makes $a_i = 0$ and $\sigma_i = 1$. Therefore, (6) becomes

$$\text{Energy}(\mathbf{v}, \mathbf{h}) = \frac{1}{2}\mathbf{v}^T\mathbf{v} - \mathbf{c}^T\mathbf{h} - \mathbf{h}^T\mathbf{W}\mathbf{v}. \tag{7}$$

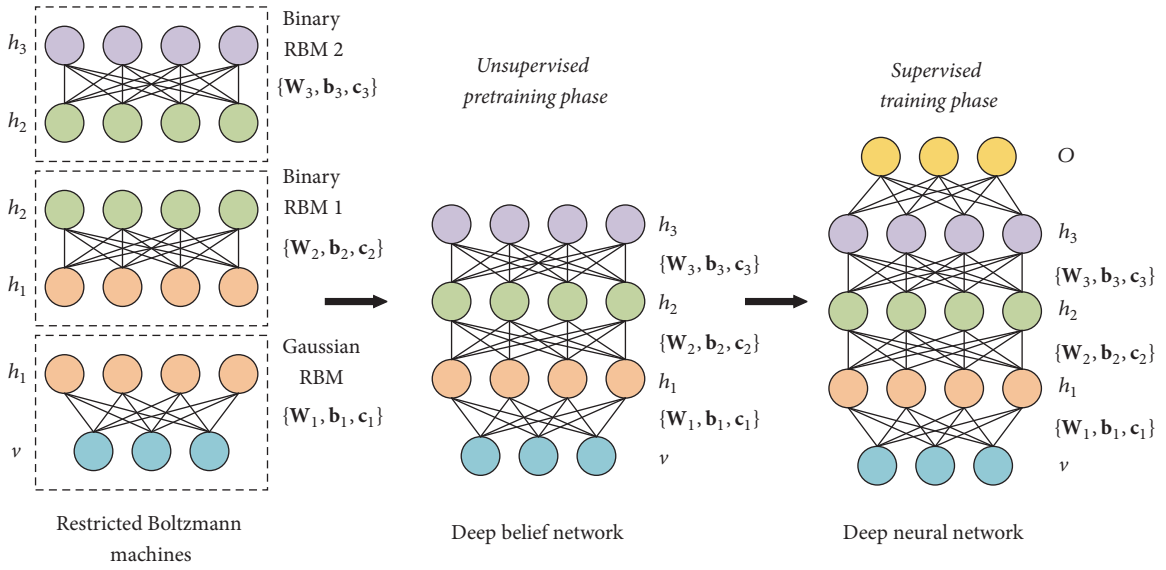


Figure 2: Deep belief network structure.

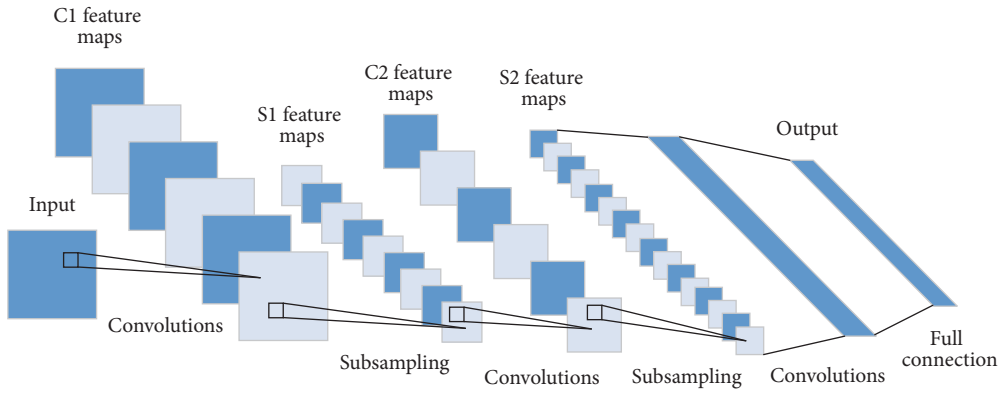


Figure 3: The architecture of convolution neural network.

2.1.2. Deep Neural Network. Hinton et al. [1] showed that RBMs can be stacked and trained in a greedy manner to form so-called deep belief networks (DBNs) [32]. DBNs are graphical models which learn to extract deep hierarchical representation of the training data. A DBN model with l layers models the joint distribution between observed vector v and ℓ hidden layers h^k as follows [30]:

$$P\left(v, h^1, \ldots, h^\ell\right) = \left(\prod_{k=0}^{\ell-2} P\left(h^k \mid h^{k+1}\right)\right) P\left(h^{\ell-1}, h^\ell\right), \quad (8)$$

where $v = h^0$, $P(h^{k-1} \mid h^k)$ is a conditional distribution for the visible units conditioned on the hidden units of the RBM at level k and $P(h^{\ell-1}, h^\ell)$ is the visible-hidden joint distribution in the top-level RBM. This is illustrated in Figure 2.

As Figure 2 shows, the hidden layer of low-level RBM is the visible layer of high-level RBM, which means that the output of low-level RBM is the input of high-level RBM. By using this structure, the high-level RBM is able to learn high-level features from low-level features generated from the low-level RBM. Thus, DBN allows latent variable space in its hidden layers. In order to train a DBN effectively, we need to train its RBM from low level to high level successively.

After the unsupervised pretraining step for DBN, the next step is to use parameters from DBN to initialize the DNN and do supervised training for DNN using back-propagation. The parameters of the N-layer DNN are initialized as follows: parameters $\{W_n, c_n\}$ $(l = 1, \ldots, N)$ except the top layer parameters are set the same as the DBN, and the top layer weights $\{W_N, c_N\}$ are initialized stochastically. After that, the whole network can be fine-tuned by back-propagation in a supervised way using labeled data.

2.2. Convolutional Neural Network. Convolutional neural network is one of the most powerful classes of deep neural networks in image processing tasks. It is highly effective and commonly used in computer vision applications [33]. The convolution neural network contains three types of layers: convolution layers, subsampling layers, and full connection layers. The whole architecture of convolutional neural network is shown in Figure 3. A brief introduction to each type of layer is provided in the following paragraphs.

3	15	64	22	55	62
92	213	7	32	145	34
17	178	86	33	12	21
231	87	48	5	23	234
59	56	55	45	3	218
82	97	94	33	238	44

1	1	1
1	0	2
1	0	1

Figure 4: Digital image representation and convolution matrix.

2.2.1. Convolution Layer. As Figure 4 shows, in convolution layer, the left matrix is the input, which is a digital image, and the right matrix is a convolution matrix. The convolution layer takes the convolution of the input image with the convolution matrix and generates the output image. Usually the convolution matrix is called filter and the output image is called filter response or filter map. An example of convolution calculation is demonstrated in Figure 5. Each time, a block of pixels is convoluted with a filter and generates a pixel in a new image.

2.2.2. Subsampling Layer. The subsampling layer is an important layer to convolutional neural network. This layer is mainly to reduce the input image size in order to give the neural network more invariance and robustness. The most used method for subsampling layer in image processing tasks is max pooling. So the subsampling layer is frequently called max pooling layer. The max pooling method is shown in Figure 6. The image is divided into blocks and the maximum value of each block is the corresponding pixel value of the output image. The reason to use subsampling layer is as follows. First, the subsampling layer has fewer parameters and it is faster to train. Second, a subsampling layer makes convolution layer tolerate translation and rotation among the input pattern.

2.2.3. Full Connection Layer. Full connection layers are similar to the traditional feed-forward neural layer. They make the neural network fed forward into vectors with a predefined length. We could fit the vector into certain categories or take it as a representation vector for further processing.

3. Training Strategy

Compared to conventional machine learning methods, the advantage of the deep learning is that it can build deep architectures to learn more multiscale abstract features. Unfortunately, the large amount of parameters of the deep architectures may lead to overfitting problem.

3.1. Data Augmentation. The key idea of data augmentation is to generate additional data without introducing extra labeling costs. In general, the data augmentation is achieved by deforming the existing ones. Mirroring, scaling, and rotation are the most common methods for data augmentation [34–36]. Wu et al. extended the deforming idea to color space, the provided color casting, vignetting, and lens distortion

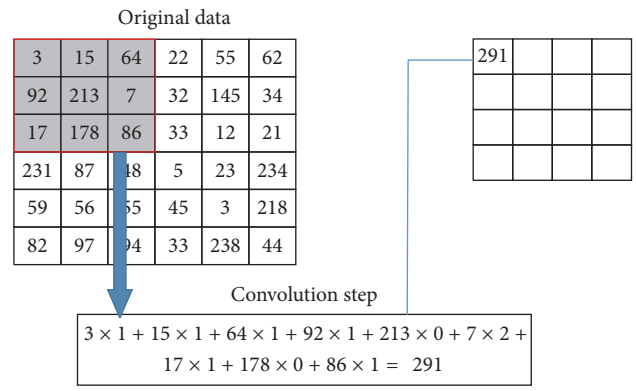


Figure 5: An example of convolution calculation.

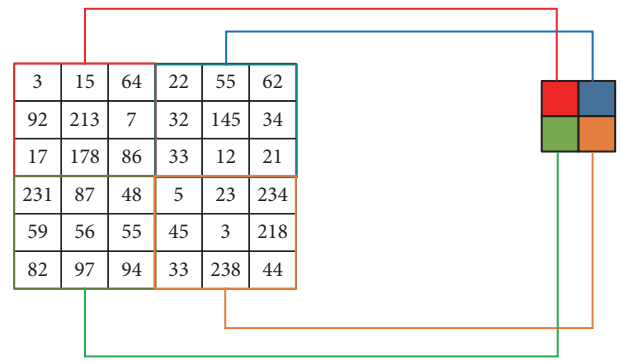


Figure 6: The example of the subsampling layer.

techniques in their work, which enlarged the training set significantly [37].

3.2. Pretraining and Fine-Tuning. Training a deep learning architecture is a time-consuming and nontrivial task. On one hand, it is difficult to obtain enough well-labeled data to train the deep learning architecture in real application, although the data augmentation can help us obtain more training data.

For visual tasks, when it is hard to get sufficient data, a recommendable way is to fine-tune the pretrained CNN by natural images (e.g., ImageNet) and then use specific data set to fine-tune the CNN [36, 38, 39]. Tajbakhsh et al. showed that, for medical applications, the use of a pretrained CNN with adequate fine-tuning outperformed or, in the worst case, performed as well as a CNN trained from scratch [38].

On the other hand, the deep learning architecture contains hundreds of thousands of parameters to be initialized even with sufficient data. Erhan et al. provided the evidence to explain that the pretraining step helps train deep architectures such as deep belief networks and stacked autoencoders [40]. Their experiments supported a regularization explanation for the effect of pretraining, which helps the deep-learned model obtain better generalization from the training data set.

4. Applications

Deep learning has been widely applied in various fields, such as computer vision [25], signal processing [24], and speech recognition [41]. In this section, we will briefly review several recently developed applications of deep learning (all the results are referred from the original papers).

4.1. CNN-Based Applications in Visual Computing. As we know, convolutional neural networks are very powerful tools for image recognition and classification. These different types of CNNs are often tested on well-known ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) data set and achieved state-of-the-art performance in recent years [42–44]. After winning the ImageNet competition in 2012 [42], the CNN-based methods have brought about a revolution in computer vision. CNNs have been applied with great success to the object detection [35, 45, 46], object segmentation [47, 48], and recognition of objects and regions in images [49–54]. Compared with hand-crafted features, for example, Local Binary Patterns (LBP) [55] and Scale Invariant Feature Transform (SIFT) [56], which need additional classifiers to solve vision problems [57–59], the CNNs can learn the features and the classifiers jointly and provide superior performance. In next subsection, we review how the deep-learned CNN is applied to recent face recognition and road crack detection problem in order to provide an overview for applying the CNN to specific problems.

4.1.1. CNN for Face Recognition. Face recognition has been one of the most important computer vision tasks since the 1970s [60]. Face recognition systems typically consist of four steps. First, given an input image with one or more faces, a face detector locates and isolates faces. Then, each face is preprocessed and aligned using either 2D or 3D modeling methods. Next, a feature extractor extracts features from an aligned face to obtain a low-dimensional representation (or embedding). Finally, a classifier makes predictions based on the low-dimensional representation. The key to get good performances for face recognition systems is obtaining an effective low-dimensional representation. Face recognition systems using hand-crafted features include [61–64]. Lawrence et al. [65] first proposed using CNNs for face recognition. Currently, the state-of-the-art performance of face recognition systems, that is, Facebook's DeepFace [66] and Google's FaceNet [67], are based on CNNs. Other notable CNN-based face recognition systems are lightened convolutional neural networks [68] and Visual Geometry Group (VGG) Face Descriptor [69].

Figure 7 shows the logic flow of CNN-based face recognition systems. Instead of using hand-crafted features, CNNs are directly applied to RGB pixel values and used as a feature extractor to provide a low-dimensional representation characterizing a person's face. In order to normalize the input image to make the face robust to different view angles, DeepFace [66] models a face in 3D and aligns it to appear as a frontal face. Then, the normalized input is fed to a single convolution-pooling-convolution filter. Next, 3 locally connected layers and 2 fully connected layers are used to make final predictions. The architecture of DeepFace is shown in Figure 8. Though DeepFace achieves the best performance on face recognition up to date, its representation is difficult to interpret and use because the faces of the same person are not clustered necessarily during the training process. In contrast, FaceNet defines a triplet loss function directly on the representation, which makes the training procedure learn to cluster face representation of the same person [70]. It should also be noted that OpenFace uses a simple 2D affine transformation to align face input.

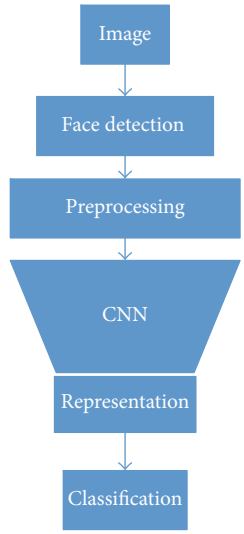


Figure 7: Logic flow of CNN-based face recognition [70].

Table 1: Experiment results on LFW benchmark [70].

Technique	Accuracy
Human-level (cropped) [74]	0.9753
FaceNet [67]	0.9964 ± 0.009
DeepFace-ensemble [66]	0.9735 ± 0.0025
OpenFace [70]	0.9292 ± 0.0134

Nowadays, face recognition in mobile computing is a very attractive topic [71, 72]. While DeepFace and FaceNet remain private and are of large size, OpenFace [70] offers a light-weighted, real-time, and open-source face recognition system with competitive accuracy, which is suitable for mobile computing. OpenFace implements FaceNet's architecture but it is one order of magnitude smaller than DeepFace and two orders of magnitude smaller than FaceNet. Their performances are compared on Labeled Faces in the Wild data set (LFW) [73], which is a standard benchmark in face recognition. The experiment results are demonstrated in Table 1. Though the accuracy of OpenFace is slightly lower than the state of the art, its smaller size and fast execution time show great potential in mobile face recognition scenarios.

4.1.2. CNN for Road Crack Detection. Automatic detection of pavement cracks is an important task in transportation maintenance for driving safety assurance. Inspired by recent success in applying deep learning to computer vision and

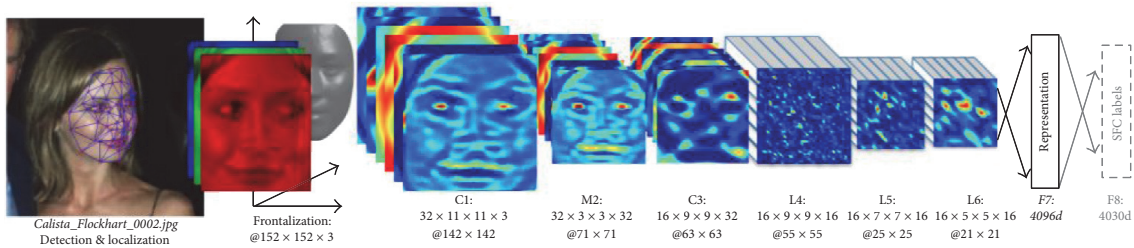


Figure 8: Outline of DeepFace architecture [66].

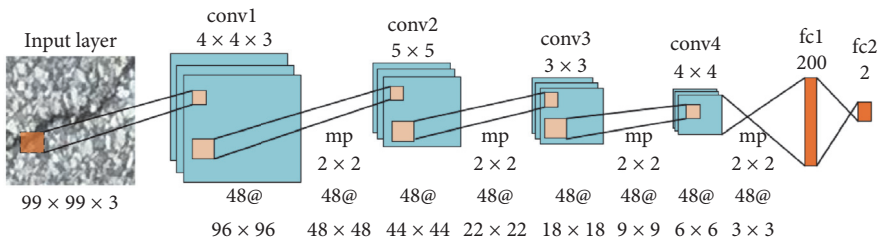


Figure 9: Illustration of the architecture of the proposed ConvNet [23].

medical problems, a deep learning based method for crack detection is proposed [23].

Data Preparation. A data set with more than 500 pavement pictures of size 3264 × 2448 is collected at the Temple University campus by using a smartphone as the data sensor. Each image is annotated by multiple annotators. Patches of size 99 × 99 are used for training and testing the proposed method. 640,000 patches, 160,000 patches, and 200,000 patches are selected as training set, validation set, and testing set, respectively.

Design and Train the CNN. A deep learning architecture is designed, which is illustrated in Figure 9 and *conv*, *mp*, and *fc* represent convolutional, max pooling, and fully connected layers, respectively. The CNNs are trained using the stochastic gradient descent (SGD) method on GPU with a batch size of 48 examples, momentum of 0.9, and weight decay of 0.0005. Less than 20 epochs are needed to reach a minimum on the validation set. The dropout method is used between two fully connected layers with a probability of 0.5 and the rectified linear units (ReLU) as the activation function.

Evaluate the Performance of the CNN. The proposed method is compared against the support vector machine (SVM) and the Boosting methods. The features for training the SVM and the Boosting method are based on color and texture of each patch which are associated with a binary label indicating the presence or absence of cracked pavement. The feature vector is 93-dimensional and is composed of color elements, histograms of textons, and LBP descriptor within the patch.

The Receiver Operating Characteristic (ROC) curves of the proposed method, the SVM, and the Boosting method are shown in Figure 10. Both the ROC curve and Area under the Curve (AUC) of the proposed method indicate that the proposed deep learning based method can outperform

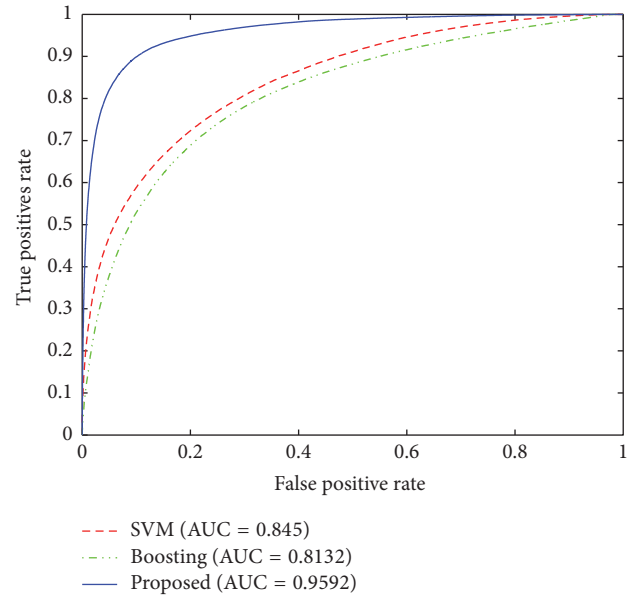
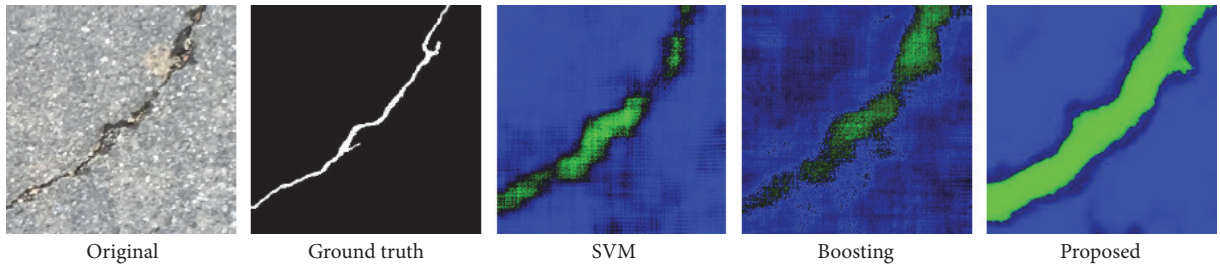


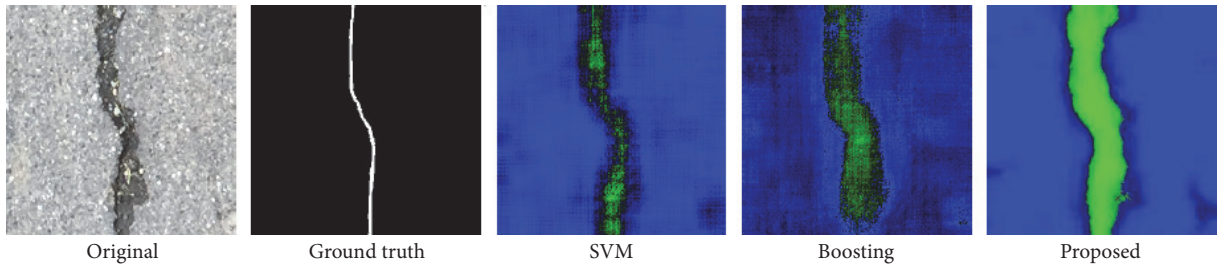
Figure 10: ROC curves [23].

the shallow structure learned from hand-crafted features. In addition, more comprehensive experiments are conducted on 300 × 300 scenes as shown in Figure 11.

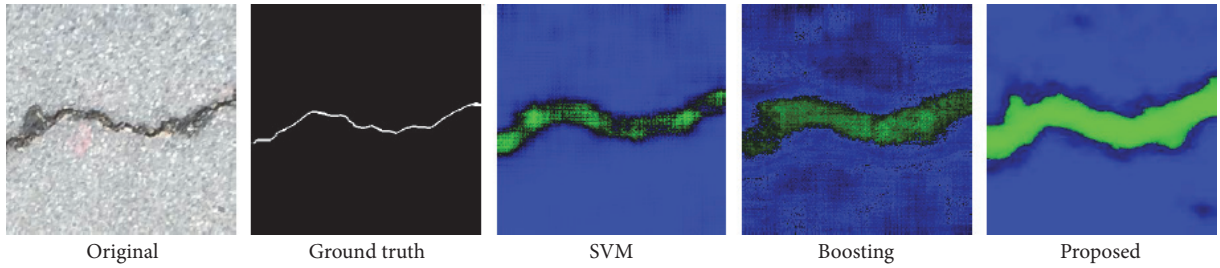
For each scene, each row shows the original image with crack, ground truth, and probability maps generated by the SVM and the Boosting methods and that by the ConvNet. The pixels in green and in blue denote the crack and the noncrack, respectively, and higher brightness means higher confidence. The SVM cannot distinguish the crack from the background, and some of the cracks have been misclassified. Compared to the SVM, the Boosting method can detect the cracks with a higher accuracy. However, some of the background



(a) Scene 1



(b) Scene 2



(c) Scene 3

Figure 11: Probability maps.

patches are classified as cracks, resulting in isolated green parts in Figure 11. In contrast to these two methods, the proposed method provides superior performance in correctly classifying crack patches from background ones.

4.2. DBN-Based Applications in Signal Processing

4.2.1. DNN for Fault Diagnosis. Plant faults may cause abnormal operations, emergency shutdowns, equipment damage, or even casualties. With the increasing complexity of modern plants, it is difficult even for experienced operators to diagnose faults fast and accurately. Thus, designing an intelligent fault detection and diagnose system to aid human operators is a critical task in process engineering. Data-driven methods for fault diagnosis are becoming very popular in recent years, since they utilize powerful machine learning algorithms. Conventional supervised learning algorithms used for fault diagnosis are Artificial Neural Networks [76–81] and support vector machines [82–84]. As one of emerging machine learning techniques, deep learning techniques are investigated for fault diagnosis in a few current studies [22, 85–88]. This subsection reviews a study which uses Hierarchical Deep Neural Network (HDNN) [22] to diagnose faults in a well-known data set called Tennessee Eastman Process (TEP).

TEP is a simulation model that simulates a real industry process. The model was first created by Eastman Chemical Company [75]. It consists of five units: a condenser, a compressor, a reactor, a separator, and a stripper. Two liquid products G and H are produced from the process with the gaseous inputs A, C, D, and E and the inert component B. The flowsheet of TEP is shown in Figure 12.

Data Preparation. The TEP is monitored by a network of M sensors that collect measurement at the same sampling time. At the ith sample, the state of mth sensor is represented by a scalar x_i^m. By combining all M sensors, the state of the whole process in ith sampling interval is represented as a row vector $x_i = [x_i^1, x_i^2, \ldots, x_i^M]$. The fault occurring at the ith sampling interval is indicated with class label $y_i \in \{1, 2, \ldots, C\}$, where value 1 to C represents one of C fault types. There are total N historical observations collected from all M sensors to form a data set $D = \{(x_i, y_i), i = 1, 2, \ldots, N, y_i \in \{1, 2, \ldots, C\}\}$. The objective of fault diagnosis is to train a classification $h : x_i \rightarrow y_i$ given data set $D = \{(x_i, y_i), i = 1, 2, \ldots, N\}$.

For each simulation run, the simulation starts without faults and the faults are introduced at sample 1. Each run collects a total of 1000 pieces of sample data. Each single fault type has 5 independent simulation runs. The Tennessee Eastman Process has 20 different predefined faults but faults

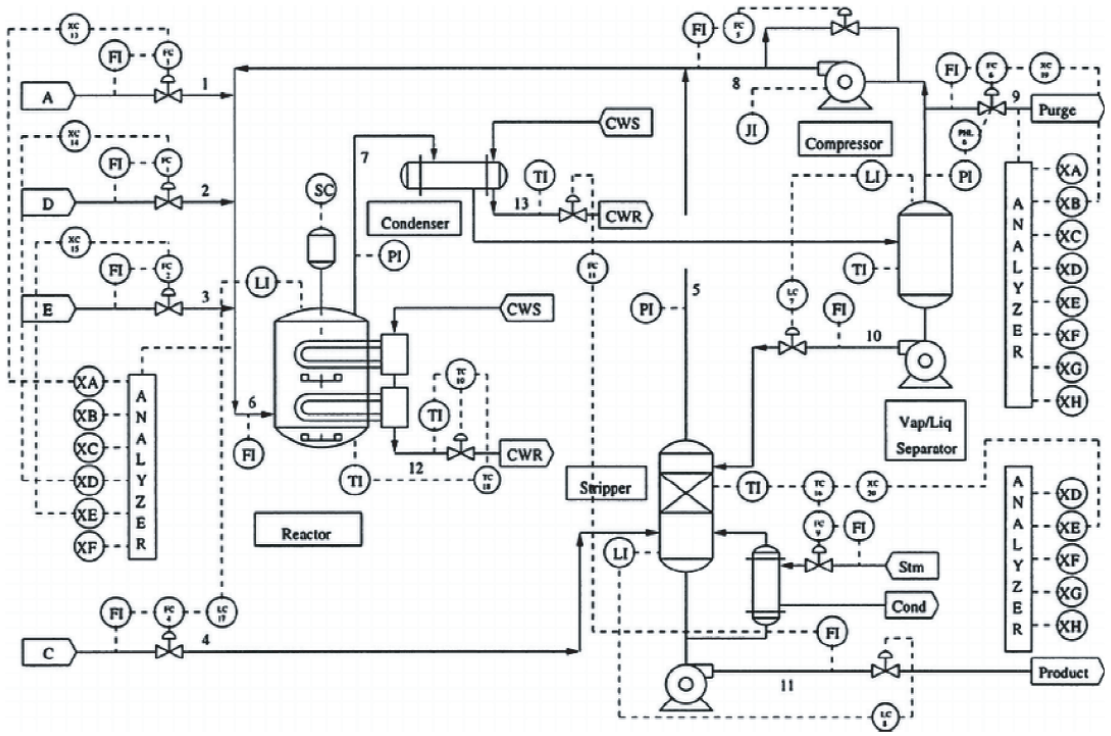


Figure 12: Tennessee Eastman Process [75].

3, 9, and 15 are excluded for fault diagnosis due to no effect or subtle effect on all the sensors [82, 84]. Thus, the training data set has a total of $N = 5 \times 17 \times 1000 = 85000$ data samples; that is, $D_{\text{train}} = \{(x_i, y_i), i = 1, 2, \ldots, N, y_i \in \{0, 1, \ldots, C\}\}$, $N = 85000$, $C = 17$. Then, test data is generated using the same method. Because only fault diagnosis methods are investigated in this work, normal operation data is not considered. Data normalization and data augmentation techniques are used to achieve better performance.

Design and Train the HDNN. The general diagnosis scheme of HDNN [22] is as follows. The symptom data generated by simulation is transmitted to a supervisory DNN. The supervisory DNN then classifies symptom data into different groups and triggers the DNN which is specially trained for that group to do further fault diagnosis. Figure 13 illustrates the fault diagnosis scheme of the HDNN, where each agent represents a DNN.

Evaluate the Performance of the DNN. The experiment result of the HDNN is compared to single neural network and Duty-Oriented Hierarchical Artificial Neural Network (DOHANN) [76] and is shown in Figure 14. 7 out of 17 faults have been diagnosed with 90% accuracy. The highest Correct Classification Rate (CCR) is 99.6% from fault 4, while the lowest CCR is 50.4% from fault 13. The average CCR of our method is 80.5%, while the average of CCRs of SNN and DOHANN is 49.7% and 70.7%, respectively. It demonstrates that the DNN-based algorithm outperforms other conventional NN-based algorithms.

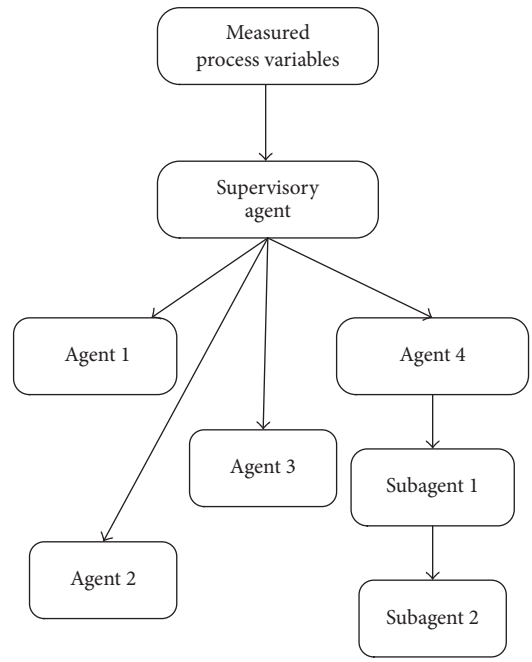


Figure 13: Schematic diagram of HDNN [22].

4.2.2. DNN for Human Activity Detection. Human activity detection has drawn much attention from researchers due to high demands for security, law enforcement, and health care [90–93]. In contrast to using cameras to detect human

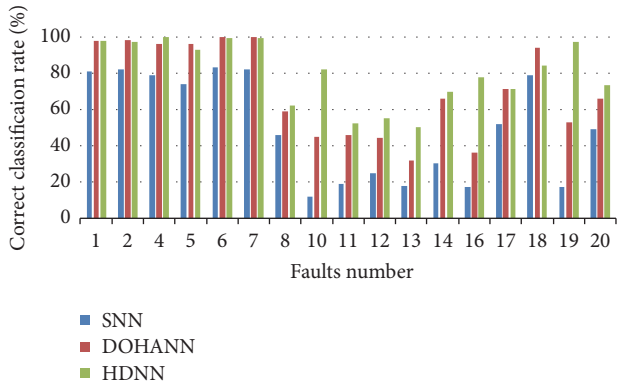


Figure 14: Correct classification rate of SNN, DOHANN [76], and HDNN [22].

activity, sensors such as worn accelerometers or in-home radar which use signals to detect human activities are robust to environmental conditions such as weather conditions and light variations [94–99]. Nowadays, there are a few emerging research works that focus on using deep learning technologies to detect human activities based on signals [89, 92, 100].

Fall detection is one of the very important human activity detection scenarios for researchers, since falls are a main cause of both fatal and nonfatal injuries for the elderly. Khan and Taati [100] proposed a deep learning method for falls detection based on signals collected from wearable devices. They propose an ensemble of autoencoders to extract features from each channel of sensing data. Unlike wearable devices which are intrusive and easily broken and must be carried, in-home radars which are safe, nonintrusive, and robust to lighting conditions show their advantages for fall detection. Jokanovic et al. [89] proposed a method that uses deep learning to detect fall motion through in-home radar. The procedure is demonstrated in Figure 15. They first denoise and normalize the spectrogram as input. Then, stacked autoencoders are performed as a feature extractor. On top of the stacked autoencoders, a softmax regression classifier is used to make predictions. The whole model is compared with a SVM model. Experiment results show that the overall correct classification rate for deep learning approach is 87%, whereas the overall correct classification rate for SVM is 78%.

5. Challenges

Though deep learning techniques achieve promising performance on multiple fields, there are still several big challenges as research articles indicate. These challenges are described as follows.

5.1. Training with Limited Data. Training deep neural network usually needs large amounts of data as larger training data set can prevent deep learning model from overfitting. Limited training data may severely affect the learning ability of a deep neural network. Unfortunately, there are many applications that lack sufficient labeled data to train a DNN. Thus, how to train DNN with limited data effectively and efficiently becomes a hot topic.

Recently, two possible solutions draw attention from researchers. One of the solutions is to generalize new training data from original training data using multiple data augmentation methods. Traditional ones include rotation, scaling, and cropping. In addition to these, Wu et al. [37] adopted vignetting, color casting, and lens distortion techniques. These techniques can further produce more different training examples. Another solution is to obtain more training data using weak learning algorithms. Song et al. [101] proposed a weakly supervised method that can label image-level object-presence. This method helps to reduce laborious bounding box annotation costs while generating training data.

5.2. Time Complexity. Training deep neural network is very time-consuming in early years. It needs a large amount of computational resources and is not suitable for real-time applications. By default, GPUs are used to accelerate training of large DNNs with the help of parallel computing technique. Thus, it is important to make the most of GPU computing ability when training DNNs. He and Sun [102] investigated training CNN under time cost constrains and proposed fast training methods for real-world applications while having similar performance as existing CNN models. Li et al. [103] remove all the redundant computations during training CNNs for pixel wise classification, which leads to a speedup of 1500 times.

5.3. Theoretical Understanding. Though deep learning algorithms achieve promising results on many tasks, the underlying theory is still not very clear. There are many questions that need to be answered. For instance, which architecture is better than other architectures in certain task? How many layers and how many nodes in each layer should be chosen in a DNN? Besides, there are a few hyperparameters such as learning rate, dropout rate, and the strength of regularizer which need to be tuned with specific knowledge.

Several approaches are developed to help researchers to get better understanding in DNN. Zeiler and Fergus [43] proposed a visualization method that illustrates features in intermediate layers. It displays intermediate features in interpretable patterns, which may help design better architectures for future DNNs. In addition to visualizing features, Girshick et al. [49] tried to discover the learning pattern of CNN by testing the performance layer by layer during the training process. It demonstrates that convolutional layers can learn more generalized features.

Although there is progress in understanding the theory of deep learning, there is still large room to improve in deep learning theory aspect.

6. Conclusion

This paper gives an overview of deep learning algorithms and their applications. Several classic deep learning algorithms such as restricted Boltzmann machines, deep belief networks, and convolutional neural networks are introduced. In addition to deep learning algorithms, their applications are

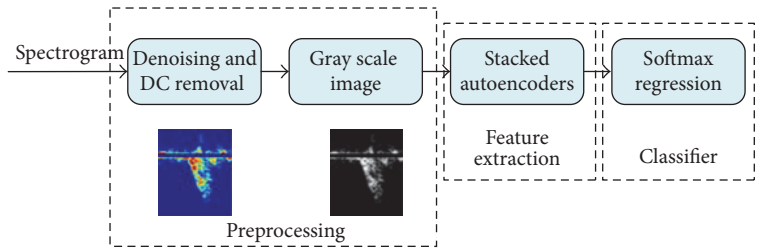


Figure 15: Block diagram of the deep learning based fall detector [89].

reviewed and compared with other machine learning methods. Though deep neural networks achieve good performance on many tasks, they still have many properties that need to be investigated and justified. We discussed these challenges and pointed out several new trends in understanding and developing deep neural networks.

Competing Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

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

[2] R. Salakhutdinov and G. E. Hinton, "Using deep belief nets to learn covariance kernels for Gaussian processes," in *Proceedings of the 21st Annual Conference on Neural Information Processing Systems (NIPS '07)*, Vcancouver, Canada, December 2007.

[3] A. Ahmed, K. Yu, W. Xu, Y. Gong, and E. Xing, "Training hierarchical feed-forward visual recognition models using transfer learning from pseudo-tasks," in *Proceedings of the in European Conference on Computer Vision*, pp. 69–82, Springer, Marseille, France, October 2008.

[4] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in Neural Information Processing Systems 19 (NIPS '06)*, pp. 153–160, MIT Press, 2007.

[5] 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 *Proceedings of the 24th International Conference on Machine Learning (ICML '07)*, pp. 473–480, ACM, Corvalis, Ore, USA, June 2007.

[6] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th Annual International Conference on Machine Learning (ICML '09)*, pp. 609–616, ACM, Quebec, Canada, June 2009.

[7] M. Ranzato, Y.-L. Boureau, and Y. LeCun, "Sparse feature learning for deep belief networks," in *Advances in Neural Information Processing Systems*, pp. 1185–1192, 2008.

[8] M. Ranzato, C. Poultney, S. Chopra, and Y. LeCun, "Efficient learning of sparse representations with an energy-based model," in *Proceedings of the 20th Annual Conference on Neural Information Processing Systems (NIPS '06)*, pp. 1137–1144, Vancouver, Canada, December 2006.

[9] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th International Conference on Machine Learning*, pp. 1096–1103, ACM, Helsinki, Finland, July 2008.

[10] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[11] R. Salakhutdinov and G. E. Hinton, "Learning a nonlinear embedding by preserving class neighbourhood structure," in *Proceedings of the 8th International Conference on Artificial Intelligence and Statistics (AISTATS '07)*, pp. 412–419, San Juan, Puerto Rico, March 2007.

[12] G. W. Taylor and G. E. Hinton, "Factored conditional restricted Boltzmann machines for modeling motion style," in *Proceedings of the 26th Annual International Conference on Machine Learning(ICML '09)*, pp. 1025–1032, ACM, Quebec, Canada, June 2009.

[13] G. W. Taylor, G. E. Hinton, and S. T. Roweis, "Modeling human motion using binary latent variables," in *Advances in Neural Information Processing Systems*, pp. 1345–1352, 2006.

[14] S. Osindero and G. E. Hinton, "Modeling image patches with a directed hierarchy of Markov random fields," in *Advances in Neural Information Processing Systems*, pp. 1121–1128, 2008.

[15] M. Ranzato and M. Szummer, "Semi-supervised learning of compact document representations with deep networks," in *Proceedings of the 25th International Conference on Machine Learning*, pp. 792–799, ACM, Helsinki, Finland, July 2008.

[16] R. Salakhutdinov and G. Hinton, "Semantic hashing," *International Journal of Approximate Reasoning*, vol. 50, no. 7, pp. 969–978, 2009.

[17] P. E. Utgoff and D. J. Stracuzzi, "Many-layered learning," *Neural Computation*, vol. 14, no. 10, pp. 2497–2529, 2002.

[18] R. Collobert and J. Weston, "A unified architecture for natural language processing: deep neural networks with multitask learning," in *Proceedings of the 25th International Conference on Machine Learning*, pp. 160–167, ACM, Helsinki, Finland, July 2008.

[19] A. Mnih and G. Hinton, "A scalable hierarchical distributed language model," in *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems (NIPS '08)*, pp. 1081–1088, British Columbia, Canada, December 2008.

[20] J. Weston, F. Ratle, H. Mobahi, and R. Collobert, "Deep learning via semi-supervised embedding," in *Neural Networks: Tricks of the Trade*, pp. 639–655, Springer, Berlin, Germany, 2012.

[21] R. Hadsell, A. Erkan, P. Sermanet, M. Scoffier, U. Muller, and Y. LeCun, "Deep belief net learning in a long-range vision system for autonomous off-road driving," in *Proceedings of the*

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '08), pp. 628–633, Nice, France, September 2008.

[22] D. Xie and L. Bai, "A hierarchical deep neural network for fault diagnosis on Tennessee-Eastman process," in *Proceedings of the IEEE 14th International Conference on Machine Learning and Applications (ICMLA '15)*, pp. 745–748, IEEE, Miami, Fla, USA, December 2015.

[23] L. Zhang, F. Yang, Y. Daniel Zhang, and Y. J. Zhu, "Road crack detection using deep convolutional neural network," in *Proceedings of the IEEE International Conference on Image Processing (ICIP '16)*, pp. 3708–3712, Phoenix, Ariz, USA, September 2016.

[24] D. Yu and L. Deng, "Deep learning and its applications to signal and information processing," *IEEE Signal Processing Magazine*, vol. 28, no. 1, pp. 145–154, 2011.

[25] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, "Deep learning for visual understanding: a review," *Neurocomputing*, vol. 187, pp. 27–48, 2016.

[26] P. Smolensky, "Information processing in dynamical systems: foundations of harmony theory," Tech. Rep. DTIC Document, 1986.

[27] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for boltzmann machines," *Cognitive Science*, vol. 9, no. 1, pp. 147–169, 1985.

[28] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang, "A tutorial on energy-based learning," in *Predicting Structured Data*, vol. 1, MIT Press, 2006.

[29] Y. LeCun and F. J. Huang, "Loss functions for discriminative training of energy-based models," in *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics (AISTATS '05)*, January 2005.

[30] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–27, 2009.

[31] M. Welling, M. Rosen-Zvi, and G. E. Hinton, "Exponential family harmoniums with an application to information retrieval," in *Advances in Neural Information Processing Systems*, pp. 1481–1488, 2004.

[32] G. Hinton, "Deep belief networks," *Scholarpedia*, vol. 4, no. 5, article no. 5947, 2009.

[33] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," *The Handbook of Brain Theory and Neural Networks*, vol. 3361, no. 10, p. 1995, 1995.

[34] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS '12)*, pp. 2843–2851, December 2012.

[35] H. R. Roth, L. Lu, J. Liu et al., "Improving computer-aided detection using convolutional neural networks and random view aggregation," *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1170–1181, 2016.

[36] S. Xie and Z. Tu, "Holistically-nested edge detection," in *Proceedings of the 15th IEEE International Conference on Computer Vision (ICCV '15)*, pp. 1395–1403, IEEE, Santiago, Chile, December 2015.

[37] R. Wu, S. Yan, Y. Shan, Q. Dang, and G. Sun, "Deep image: scaling up image recognition," https://arxiv.org/abs/1501.02876.

[38] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu et al., "Convolutional neural networks for medical image analysis: full training or fine tuning?" *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1299–1312, 2016.

[39] H.-C. Shin, H. R. Roth, M. Gao et al., "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning," *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.

[40] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, "Why does unsupervised pre-training help deep learning?" *Journal of Machine Learning Research*, vol. 11, pp. 625–660, 2010.

[41] O. Abdel-Hamid, A.-R. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '12)*, pp. 4277–4280, IEEE, Kyoto, Japan, March 2012.

[42] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS '12)*, pp. 1097–1105, December 2012.

[43] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part I*, vol. 8689 of *Lecture Notes in Computer Science*, pp. 818–833, Springer, 2014.

[44] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: integrated recognition, localization and detection using convolutional networks," https://arxiv.org/abs/1312.6229.

[45] X. Wang, L. Zhang, L. Lin, Z. Liang, and W. Zuo, "Deep joint task learning for generic object extraction," in *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS '14)*, pp. 523–531, ACM, Montreal, Canada, December 2014.

[46] J. Liu, N. Lay, Z. Wei et al., "Colitis detection on abdominal CT scans by rich feature hierarchies," in *Medical Imaging 2016: Computer-Aided Diagnosis*, vol. 9785 of *Proceedings of SPIE*, San Diego, Calif, USA, February 2016.

[47] G. Luo, R. An, K. Wang, S. Dong, and H. Zhang, "A deep learning network for right ventricle segmentation in short-axis mri," in *Proceedings of the Computing in Cardiology Conference (CinC '16)*, pp. 224–227, Vancouver, Canada, September 2016.

[48] H. R. Roth, L. Lu, A. Farag, A. Sohn, and R. M. Summers, "Spatial aggregation of holistically-nested networks for automated pancreas segmentation," https://arxiv.org/abs/1606.07830.

[49] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*, pp. 580–587, IEEE, Columbus, Ohio, USA, June 2014.

[50] R. Girshick, "Fast R-CNN," in *Proceedings of the 15th IEEE International Conference on Computer Vision (ICCV '15)*, pp. 1440–1448, December 2015.

[51] J. Liu, C. Gao, D. Meng, and W. Zuo, "Two-stream contextualized CNN for fine-grained image classification," in *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pp. 4232–4233, Phoenix, Ariz, USA, February 2016.

[52] K. Wang, L. Lin, W. Zuo, S. Gu, and L. Zhang, "Dictionary pair classifier driven convolutional neural networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '16)*, pp. 2138–2146, Las Vegas, Nev, USA, June 2016.

[53] L. Lin, K. Wang, W. Zuo, M. Wang, J. Luo, and L. Zhang, "A deep structured model with radius-margin bound for 3D human

activity recognition," *International Journal of Computer Vision*, vol. 118, no. 2, pp. 256–273, 2016.

[54] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: surpassing human-level performance on imagenet classification," in *Proceedings of the 15th IEEE International Conference on Computer Vision (ICCV '15)*, pp. 1026–1034, IEEE, Santiago, Chile, December 2015.

[55] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[56] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[57] W. Lu, M. Li, and L. Zhang, "Palm vein recognition using directional features derived from local binary patterns," *International Journal of Signal Processing, Image Processing and Pattern Recognition*, vol. 9, no. 5, pp. 87–98, 2016.

[58] D. Xie, Z. Huang, S. Wang, and H. Liu, "Moving objects segmentation from compressed surveillance video based on motion estimation," in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR '12)*, pp. 3132–3135, IEEE, Tsukuba, Japan, November 2012.

[59] S. Wang, H. Liu, D. Xie, and B. Zeng, "A novel scheme to code object flags for video synopsis," in *Proceedings of the IEEE Visual Communications and Image Processing (VCIP '12)*, pp. 1–5, November 2012.

[60] T. Kanade, *Picture processing system by computer complex and recognition of human faces [Ph.D. thesis]*, Kyoto University, 3952, 1973.

[61] D. Chen, X. Cao, F. Wen, and J. Sun, "Blessing of dimensionality: high-dimensional feature and its efficient compression for face verification," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '13)*, pp. 3025–3032, June 2013.

[62] X. Cao, D. Wipf, F. Wen, G. Duan, and J. Sun, "A practical transfer learning algorithm for face verification," in *Proceedings of the 14th IEEE International Conference on Computer Vision (ICCV '13)*, pp. 3208–3215, December 2013.

[63] T. Berg and P. N. Belhumeur, "Tom-vs-Pete classifiers and identity-preserving alignment for face verification," in *Proceedings of the 23rd British Machine Vision Conference (BMVC '12)*, BMVA Press, September 2012.

[64] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun, "Bayesian face revisited: a joint formulation," in *Computer Vision—ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7–13, 2012, Proceedings, Part III*, vol. 7574 of *Lecture Notes in Computer Science*, pp. 566–579, Springer, Berlin, Germany, 2012.

[65] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: a convolutional neural-network approach," *IEEE Transactions on Neural Networks*, vol. 8, no. 1, pp. 98–113, 1997.

[66] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "DeepFace: closing the gap to human-level performance in face verification," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*, pp. 1701–1708, June 2014.

[67] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: a unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '15)*, pp. 815–823, IEEE, Boston, Mass, USA, June 2015.

[68] X. Wu, R. He, Z. Sun, and T. Tan, "A light CNN for deep face representation with noisy labels," https://arxiv.org/abs/1511.02683.

[69] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *Proceedings of the British Machine Vision Conference*, vol. 1, p. 6, 2015.

[70] B. Amos, B. Ludwiczuk, and M. Satyanarayanan, "Openface: a general-purpose face recognition library with mobile applications," Tech. Rep. CMU-CS-16-118, CMU School of Computer Science, 2016.

[71] T. Soyata, R. Muraleedharan, C. Funai, M. Kwon, and W. Heinzelman, "Cloud-vision: real-time face recognition using a mobile-cloudlet-cloud acceleration architecture," in *Proceedings of the 17th IEEE Symposium on Computers and Communication (ISCC '12)*, pp. 59–66, July 2012.

[72] H.-J. Hsu and K.-T. Chen, "Face recognition on drones: issues and limitations," in *Proceedings of the 1st Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use (DroNet '15)*, pp. 39–44, ACM, Florence, Italy, 2015.

[73] 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," Tech. Rep. 07-49, University of Massachusetts, Amherst, Mass, USA, 2007.

[74] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, "Attribute and simile classifiers for face verification," in *Proceedings of the 12th International Conference on Computer Vision (ICCV '09)*, pp. 365–372, IEEE, Kyoto, Japan, October 2009.

[75] J. J. Downs and E. F. Vogel, "A plant-wide industrial process control problem," *Computers & Chemical Engineering*, vol. 17, no. 3, pp. 245–255, 1993.

[76] R. Eslamloueyan, "Designing a hierarchical neural network based on fuzzy clustering for fault diagnosis of the Tennessee-Eastman process," *Applied Soft Computing Journal*, vol. 11, no. 1, pp. 1407–1415, 2011.

[77] V. Venkatasubramanian and K. Chan, "A neural network methodology for process fault diagnosis," *AIChE Journal*, vol. 35, no. 12, pp. 1993–2002, 1989.

[78] K. Watanabe, I. Matsuura, M. Abe, M. Kubota, and D. M. Himmelblau, "Incipient fault diagnosis of chemical processes via artificial neural networks," *AIChE Journal*, vol. 35, no. 11, pp. 1803–1812, 1989.

[79] J. Y. Fan, M. Nikolaou, and R. E. White, "An approach to fault diagnosis of chemical processes via neural networks," *AIChE Journal*, vol. 39, no. 1, pp. 82–88, 1993.

[80] K. Watanabe, S. Hirota, L. Hou, and D. M. Himmelblau, "Diagnosis of multiple simultaneous fault via hierarchical artificial neural networks," *AIChE Journal*, vol. 40, no. 5, pp. 839–848, 1994.

[81] R. Eslamloueyan, M. Shahrokhi, and R. Bozorgmehri, "Multiple simultaneous fault diagnosis via hierarchical and single artificial neural networks," *Scientia Iranica*, vol. 10, no. 3, pp. 300–310, 2003.

[82] L. H. Chiang, M. E. Kotanchek, and A. K. Kordon, "Fault diagnosis based on Fisher discriminant analysis and support vector machines," *Computers and Chemical Engineering*, vol. 28, no. 8, pp. 1389–1401, 2004.

[83] M. Ge, R. Du, G. Zhang, and Y. Xu, "Fault diagnosis using support vector machine with an application in sheet metal stamping operations," *Mechanical Systems and Signal Processing*, vol. 18, no. 1, pp. 143–159, 2004.

[84] M. Grbovic, W. Li, P. Xu, A. K. Usadi, L. Song, and S. Vucetic, "Decentralized fault detection and diagnosis via sparse PCA

based decomposition and maximum entropy decision fusion," *Journal of Process Control*, vol. 22, no. 4, pp. 738–750, 2012.

[85] W. Sun, S. Shao, R. Zhao, R. Yan, X. Zhang, and X. Chen, "A sparse auto-encoder-based deep neural network approach for induction motor faults classification," *Measurement*, vol. 89, pp. 171–178, 2016.

[86] M. Gan, C. Wang, and C. Zhu, "Construction of hierarchical diagnosis network based on deep learning and its application in the fault pattern recognition of rolling element bearings," *Mechanical Systems and Signal Processing*, vol. 72-73, pp. 92–104, 2016.

[87] P. Jiang, Z. Hu, J. Liu, S. Yu, and F. Wu, "Fault diagnosis based on chemical sensor data with an active deep neural network," *Sensors*, vol. 16, no. 10, p. 1695, 2016.

[88] H. J. Steinhauer, A. Karlsson, G. Mathiason, and T. Helldin, "Root-cause localization using restricted Boltzmann machines," in *Proceedings of the 19th International Conference on Information Fusion (FUSION '16)*, pp. 248–255, ISIF, 2016.

[89] B. Jokanovic, M. Amin, and F. Ahmad, "Radar fall motion detection using deep learning," in *Proceedings of the IEEE Radar Conference (RadarConf '16)*, IEEE, Philadelphia, Pa, USA, May 2016.

[90] L. M. Frazier, "MDR for law enforcement," *IEEE Potentials*, vol. 16, no. 5, pp. 23–26, 1997.

[91] E. F. Greneker, "Radar flashlight for through the wall detection of humans," in *SPIE Proceedings of the Targets and Backgrounds: Characterization and Representation IV*, pp. 280–285, SPIE, Orlando, Fla, USA, April 1998.

[92] J. Park, R. Javier, T. Moon, and Y. Kim, "Micro-doppler based classification of human aquatic activities via transfer learning of convolutional neural networks," *Sensors*, vol. 16, no. 12, p. 1990, 2016.

[93] J. Sung, C. Ponce, B. Selman, and A. Saxena, "Unstructured human activity detection from RGBD images," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '12)*, pp. 842–849, St Paul, Minn, USA, May 2012.

[94] J. R. Smith, K. P. Fishkin, B. Jiang et al., "RFID-based techniques for human-activity detection," *Communications of the ACM*, vol. 48, no. 9, pp. 39–44, 2005.

[95] P. Van Dorp and F. C. A. Groen, "Human walking estimation with radar," *IEE Proceedings: Radar, Sonar and Navigation*, vol. 150, no. 5, pp. 356–366, 2003.

[96] R. J. Javier and Y. Kim, "Application of linear predictive coding for human activity classification based on micro-doppler signatures," *IEEE Geoscience and Remote Sensing Letters*, vol. 11, no. 10, pp. 1831–1834, 2014.

[97] Y. Kim and H. Ling, "Human activity classification based on micro-doppler signatures using a support vector machine," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, no. 5, pp. 1328–1337, 2009.

[98] R. Igual, C. Medrano, and I. Plaza, "Challenges, issues and trends in fall detection systems," *BioMedical Engineering Online*, vol. 12, no. 1, article 66, 2013.

[99] P. Rashidi and A. Mihailidis, "A survey on ambient-assisted living tools for older adults," *IEEE Journal of Biomedical and Health Informatics*, vol. 17, no. 3, pp. 579–590, 2013.

[100] S. S. Khan and B. Taati, "Detecting unseen falls from wearable devices using channel-wise ensemble of autoencoders," https://arxiv.org/abs/1610.03761.

[101] H. O. Song, Y. J. Lee, S. Jegelka, and T. Darrell, "Weakly-supervised discovery of visual pattern configurations," in *Proceedings of the 28th Annual Conference on Neural Information Processing Systems (NIPS '14)*, pp. 1637–1645, Québec, Canada, December 2014.

[102] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '15)*, pp. 5353–5360, Boston, Mass, USA, June 2015.

[103] H. Li, R. Zhao, and X. Wang, "Highly efficient forward and backward propagation of convolutional neural networks for pixelwise classification," https://arxiv.org/abs/1412.4526.

Hindawi
Applied Computational Intelligence and Soft Computing
Volume 2017, Article ID 5861435, 11 pages
http://dx.doi.org/10.1155/2017/5861435

Research Article

Mining Key Skeleton Poses with Latent SVM for Action Recognition

Xiaoqiang Li,[1] Yi Zhang,[1] and Dong Liao[2]

[1]*School of Computer Engineering and Science, Shanghai University, Shanghai, China*
[2]*School of Mathematic and Statistics, Nanyang Normal University, Nanyang, China*

Correspondence should be addressed to Xiaoqiang Li; xqli@i.shu.edu.cn and Dong Liao; liaodong@nynu.edu.cn

Received 23 August 2016; Revised 8 November 2016; Accepted 15 December 2016; Published 23 January 2017

Academic Editor: Lei Zhang

Human action recognition based on 3D skeleton has become an active research field in recent years with the recently developed commodity depth sensors. Most published methods analyze an entire 3D depth data, construct mid-level part representations, or use trajectory descriptor of spatial-temporal interest point for recognizing human activities. Unlike previous work, a novel and simple action representation is proposed in this paper which models the action as a sequence of inconsecutive and discriminative skeleton poses, named as key skeleton poses. The pairwise relative positions of skeleton joints are used as feature of the skeleton poses which are mined with the aid of the latent support vector machine (latent SVM). The advantage of our method is resisting against intraclass variation such as noise and large nonlinear temporal deformation of human action. We evaluate the proposed approach on three benchmark action datasets captured by Kinect devices: MSR Action 3D dataset, UTKinect Action dataset, and Florence 3D Action dataset. The detailed experimental results demonstrate that the proposed approach achieves superior performance to the state-of-the-art skeleton-based action recognition methods.

1. Introduction

The task of automatic human action recognition has been studied over the last few decades as an important area of computer vision research. It has many applications including video surveillance, human computer interfaces, sports video analysis, and video retrieval. Despite remarkable research efforts and many encouraging advances in the past decade, accurate recognition of the human actions is still a quite challenging task [1].

In traditional RGB videos, human action recognition mainly focuses on analyzing spatiotemporal volumes and representation of spatiotemporal volumes. According to the variety of visual spatiotemporal descriptors, human action recognition work can be classified into three categories. The first category is local spatiotemporal descriptors. An action recognition method first detects interesting points (e.g., STIPs [2] or trajectories [3]) and then computes descriptors (e.g., HOG/HOF [2] and HOG3D [4]) based on the detected local motion volumes. These local features are then combined (e.g., bag-of-words) to represent actions. The second category is global spatiotemporal templates that represent the entire action. A variety of image measurements have been proposed to populate such templates, including optical flow and spatiotemporal orientations [5, 6] descriptors. Except the local and holistic representational method, the third category is mid-level part representations which model moderate portions of the action. Here, parts have been proposed which capture a neighborhood of spacetime [7, 8] or a spatial key frame [9]. These representations attempt to balance the trade-off between generality exhibited by small patches, for example, visual words, and the specificity by large ones, for example, holistic templates. In addition, with the advent of inexpensive RGB-depth sensors such as Microsoft Kinect [10], a lot of efforts have been made to extract features for action recognition in depth data and skeletons. Reference [11] represents each depth frame as a bag of 3D points along the human silhouette and utilizes HMM to model the temporal dynamics. Reference [12] learns semilocal features automatically from the data with an efficient random sampling approach. Reference [13] selects most informative joints based on the

Figure 1: Two athletes perform the same action (diving water) in different way.

discriminative measures of each joint. Inspired by [14], Seidenari et al. model the movements of the human body using kinematic chains and perform action recognition by Nearest-Neighbor classifier [15]. In [16], skeleton sequences are represented as trajectories in an n-dimensional space; then these trajectories are then interpreted in a Riemannian manifold (shape space). Recognition is finally performed using kNN classification on this manifold. Reference [17] extracts a sparse set of active joint coordinates and maps these coordinates to lower-dimensional linear manifold before training an SVM classifier. The methods above generally extract the spatial-temporal representation of the skeleton sequences with well-designed handcrafted features. Recently, with the developing of deep learning, several Recurrent Neural Networks (RNN) models have been proposed for action recognition. In order to recognize actions according to the relative motion between limbs and the trunk, [18] uses an end-to-end hierarchical RNN for skeleton-based action recognition. Reference [19] uses skeleton sequences to regularize the learning of Long Short Term Memory (LSTM), which is grounded via deep Convolutional Neural Network (DCNN) onto the video for action recognition.

Most of the above methods relied on entire video sequences (RGB or RGBD) to perform action recognition, in which spatiotemporal volumes were always selected as representative feature of action. These methods will suffer from sensitivity to intraclass variation such as temporal scale or partial occlusions. For example, Figure 1 shows that two athletes perform some different poses when diving water, which makes the spatiotemporal volumes different. Motivated by this case, the question we seek to answer in this paper is whether a few inconsecutive key skeleton poses are enough to perform action recognition. As far as we know, this is an unresolved issue, which has not yet been systematically investigated. In our early work [20], it has been proven that some human actions could be recognized with only a few inconsecutive and discriminative frames for RGB video sequences. Related to our work, very short snippets [9] and discriminative action-specific patches [21] are proposed as representation of specific action. However, in contrast to our method, these two methods focused on consecutive frame.

In this paper, a novel framework is proposed for action recognition in which key skeleton poses are selected as representation of action in RGBD video sequences. In order to make our method more robust to translation, rotation, and scaling, Procrustes analysis [22] is conducted on 3D skeleton joint data. Then, the pairwise relative positions of the 3D skeleton joints are computed as discriminative features to represent the human movement. Finally, key skeleton poses, defined as the most representative skeleton model of the action, are mined from the 3D skeleton videos with the help of latent support vector machine (latent SVM) [23]. In early exploration experiments, we noticed that the number of the inconsecutive key skeleton poses is no smaller than 4. During testing, the temporal position and similarity of each of the key poses are compared with the model of the action. The proposed approach has been evaluated on three benchmark datasets: MSR Action 3D [24] dataset, UTKinect Action dataset [25], and Florence 3D Action dataset [26]; all are captured with Kinect devices. Experimental results demonstrate that the proposed approach achieves better recognition accuracy than a few existing methods. The remainder of this paper is organized as follows. The proposed approach is elaborated in Section 2 including the feature extracting, key poses mining, and action recognizing. Experimental results are shown and analyzed in Section 3. Finally, we conclude this paper in Section 4.

2. Proposed Approach

Due to the large performance variation of an action, the appearance, temporal structure, and motion cues exhibit large intraclass variability. So selecting the inconsecutive and discriminative key poses is a promising method to represent the action. In this section, we answer the question of what are and how to find the discriminative key poses.

2.1. Definition of the Key Poses and Model Structure. The structure of the proposed approach is shown in Figure 2. Each action model is composed of a few key poses, and each key pose in the model will be represented by three parts: (1) a linear classifier $g_i(x)$ which can discriminate the key

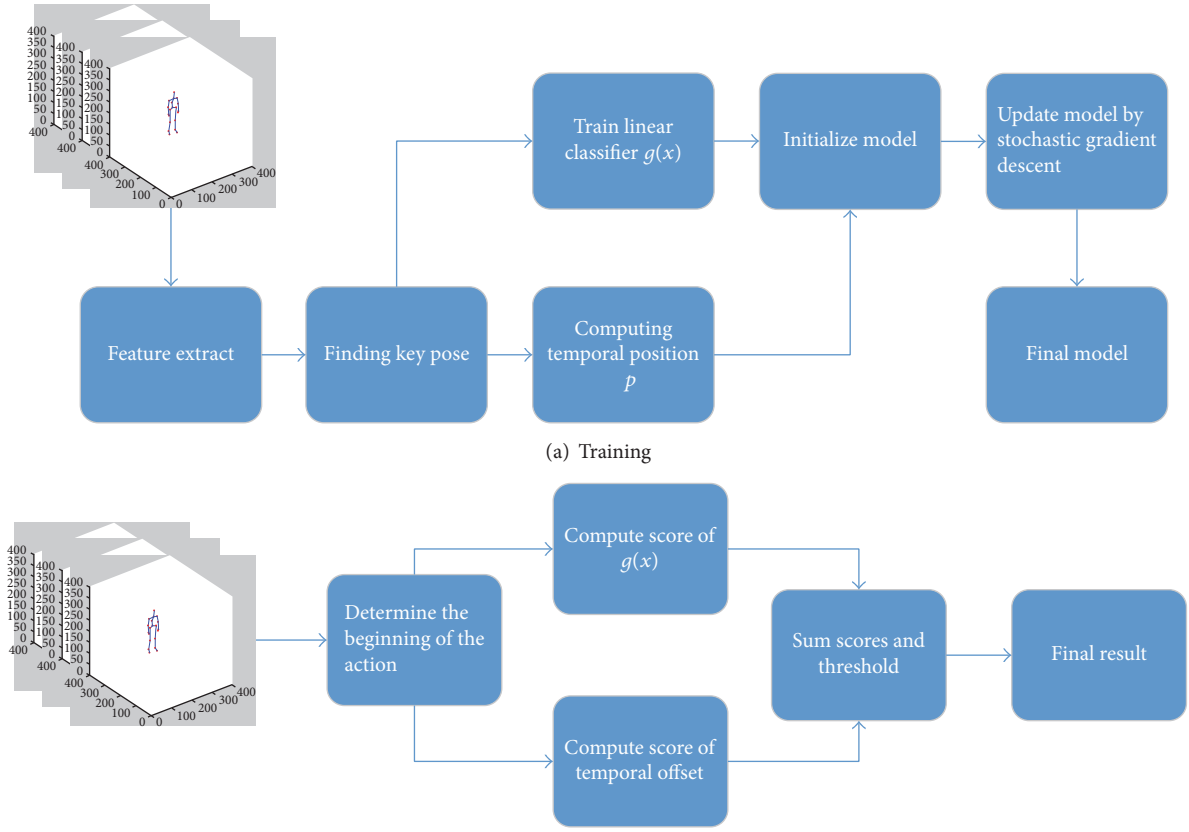


FIGURE 2: Structure of our model.

pose from the others, (2) the temporal position p_i and offset o_i, where the key poses i are most likely to appear in the neighborhood of p_i with radius o_i, and (3) the weight of linear classifier w_{g_i} and weight of the temporal information w_{p_i}.

Given is a video that contains m frames $X = \{x_1, \ldots, x_m\}$, where x_i is the i-th frame of the video. The score will be computed as follows:

$$f\left(X_{T^n}\right) = \max_{t \in T^n} \sum_{i=1}^{n} \left(w_{g_i} \times g\left(x_{t_i}\right) + w_{p_i} \times \Delta t_i\right), \quad (1)$$

in which X_{T^n} is the set of key poses of video X, $T^n = \{t \mid t = (t_1, \ldots, t_n),\ 1 \le t_i \le m\}$, and $x_{t_i} \in X_{T^n}$. For example, T^n is $\{1, 9, 10, 28\}$ in Figure 3(a). n is the total number of key poses in the action model; in our following experiment, n is ranging from 1 to 20. t_i is the serial number of the key pose in the sequence of frames of video. And Δt_i is defined as follows:

$$\Delta t_i = \frac{1}{2\pi o_i} \exp\left(\frac{-\left(t_i - t_0 - p_i\right)^2}{2{o_i}^2}\right), \quad (2)$$

in which t_0 is the frame at which action begins. Δt is a Gaussian function and reaches peak when $t_i - t_0 = p_i$. t_0 has been manually labeled on the training set. The method of finding t_0 in a testing will be discussed in Section 2.4.

2.2. Feature Extracting and Linear Classifier. With the help of real-time skeleton estimation algorithm, the 3D joint positions are employed to characterize the motion of the human body. Following the methods [1], we also represent the human movement as the pairwise relative positions of the joints.

For a human skeleton, joint positions are tracked by the skeleton estimation algorithm and each joint j has 3 coordinates at each frame. The coordinates are normalized based on Procrustes analysis [22], so that the motion is invariant to the initial body orientation and the body size. For a given frame $x = \{j_{1_x}, j_{1_y}, j_{1_z}, \ldots, j_{n_x}, j_{n_y}, j_{n_z}\}$, n is the number of joints. The feature of this frame $\varphi(x)$ is

$$\begin{aligned} \varphi(x) &= \left\{j_{a,b} \mid j_{a,b} = j_a - j_b,\ 1 \le a < b \le n\right\} \\ &\quad + \left\{j_{1_x}, j_{1_y}, j_{1_z}, \ldots, j_{n_x}, j_{n_y}, j_{n_z}\right\} \\ j_a - j_b &= \left\{j_{a_x} - j_{b_x}, j_{a_y} - j_{b_y}, j_{a_z} - j_{b_z}\right\}. \end{aligned} \quad (3)$$

And the feature is a 630-dimension (570 pairwise relative positions of the joint and 60 joint position coordinates) vector for MSR Action 3D and UTKinect Action dataset. AS for Florence 3D Action dataset, it is a 360-dimension vector. (The

(a) Key poses of drink are 1, 9, 10, and 28 (subject 1, action drink, video 4, total 32 frames)

(b) Key poses of stand up are 1, 3, 10, and 11 (subject 1, action stand up, video 21, total 29 frames)

(c) Key poses of wave are 4, 7, 8, and 12 (subject 2, action wave, video 1, total 13 frames)

(d) Key poses of drink are 5, 7, 9, and 11 (subject 2, action drink, video 3, total 14 frames)

Figure 3: Key poses for different action in Florence 3D Actions dataset.

selection of alternative feature representations will be discussed in Experiment Result.) Then, we train a linear classifier for each key pose according to the following equation:

$$g(x) = w \cdot \varphi(x). \tag{4}$$

The question of which frame should be used for training $g(x)$ will be discussed in Section 2.3.

2.3. Latent Key Poses Mining. It is not easy to decide which frames contain the key poses, because key poses' space T^n is too large to enumerate all the possible poses. Enlightened by [23], since the key pose positions are not observable in the training data, we formulate the learning problem as a latent structural SVM, regarding the key pose positions as the latent variable.

Rewrite (1) as follows:

$$\begin{aligned} f(X) &= \max_{t \in T^n} W \cdot \Phi(X, t) \\ W &= \left(w_{g_1}, w_{p_1}, \ldots, w_{g_n}, w_{p_n}\right) \\ \Phi(X, t) &= \left(g\left(x_{t_1}\right), \Delta t_1, \ldots, g\left(x_{t_n}\right), \Delta t_n\right), \end{aligned} \tag{5}$$

in which $t = (t_1, \ldots, t_n)$ is treated as the latent variable. Given a labeled set $D = \{\langle X_1, Y_1\rangle, \ldots, \langle X_i, Y_i\rangle, \ldots\}$, where $X_i = \{x_1, \ldots, x_m\}$ and $Y_i \in \{-1, +1\}$, the objective is to minimize the objective function:

$$L_D(W) = \frac{1}{2}\|W\|^2 + C\sum_{i=1}^{n} \max\left(0, 1 - Y_i f(X_i)\right), \tag{6}$$

Require:
D_p, D_n, N;
$pos = 1$, $neg\_pose = \{x_i \mid i = random(),\ x_i \in X,\ X \in D_n\}$;
for $i = 1 \ldots N$ **do**
$o_i = 5$
$\varphi^{exp} = \varphi(x_{pos})$
where x_{pos} is the pos-th frame of the first video in D_p
for $X \in D_p$ **do**

$$pos\_pose = \left\{ pos\_pose,\ \arg\min_{x_j} \left(Euclidean\left(\varphi^{exp}, \varphi\left(x_j\right)\right)\right)\right\}$$

where $pos - o_i < j < pos + o_i,\ x_j \in X$
end for
Train $g_i(x)$ with pos_pose and neg_pose
$p_i = average\ \{j \mid x_j \in pos\_pose\}$
for $X \in D_p$ **do**
For each frame $x_j \in X$, $s[j] = s[j] + g_i(x_j)$
end for
$pos = \arg\min_j(s[j])$
end for
Training w_{g_i} and w_{p_i} with linear SVM

Algorithm 1

in which C is the penalty parameter. Following [23], the model is first initialized: D_p and D_n are the positive and negative subsets of D, and the model is initialized with N key frames as shown in Algorithm 1. In Algorithm 1, *pos_pose* and *neg_pose* are the positive frame set and the negative frame set, respectively. They are used to train the linear classifier $g(x)$. In order to initialize our model, we firstly compute $\varphi(x_{pos})$, the feature of the pos-th frame which belongs to the first video sample in D_p. Then the Euclidean distance between $\varphi(x_{pos})$ and the feature of the frames in other samples in the neighborhood of temporal position pos with radius o_i in D_p is computed. The frame which has the minimum Euclidean distance from $\varphi(x_{pos})$ in each sample is added in *pos_pose*. Then *pos_pose* is used to train the linear classifier $g_i(x)$ and choose p_i as the average of frame number in *pos_pose*. To select the next key pose, pos chose j with the minimum score based on $g_i(x)$ for next loop; in other words, the j-th frame which is most different from previous key pose is selected in the next loop. Finally, all w_{g_i} and w_{p_i} are trained with the linear SVM when Algorithm 1 is completed.

Once the initialization is finished, the model will be iteratively trained as follows. First, to find the optimal value t subjected to $t^{\text{opt}} \in T^n$ where $t^{\text{opt}} = \arg\max_t(W \cdot \Phi(X, t))$ for each positive video example and update p with the average value of all t^{opt}, the new linear classifier $g(x)$ is trained with modified p for each key pose. Second, (6) is optimized over W, where $f(x) = W \cdot \Phi(X, t^{\text{opt}})$ with stochastic gradient descent. Thus, the models are modified to better capture skeleton characteristics for each action.

2.4. Action Recognition with Key Poses. The key technical issue in action recognition in real-world video is that we do not know where the action starts, and searching start position in all possible places takes a lot of time. Fortunately, the score of each possible start position can be computed, respectively. So a parallel tool such as OpenMP or CUDA might be helpful.

Given a test video X with m frames, first, the skeleton feature score $g(x)$ of each frame has been computed in advance so we could reuse them later. Then for each possible action start position t_0, we compute the score of each key pose x_{t_i} according to the following equation:

$$\text{score} = \max_{t_i \geq t_0} \left(w_{g_i} \times g\left(x_{t_i}\right) + w_{p_i} \times \Delta t_i \right). \tag{7}$$

These scores are summed together as the final score of t_0. If the final score is bigger than the threshold, then an action beginning at t_0 has been detected and recognized. Figure 3 shows key poses for different actions in Florence 3D Action dataset.

3. Experiment Result

This section presents all experimental results. First, trying to eliminate the noise generated by translation, scale, and rotation changes of skeleton poses, we preprocess the dataset with Procrustes analysis [22]. And we conduct the experiment for action recognition with or without Procrustes analysis on UTKinect dataset to demonstrate effectiveness of Procrustes analysis. Second, the appropriate feature extraction was selected from four existing feature extraction methods according to experimental result on Florence 3D Action dataset. Third, quantitative experiment is conducted to select the number of inconsecutive key poses. Last, we evaluate our model and compare it with some state-of-the-art method on three benchmark datasets: MSR Action 3D dataset, UTKinect Action dataset, and Florence 3D Action dataset.

3.1. Datasets

(1) Florence 3D Action Dataset. Florence 3D Action dataset [26] was collected at the University of Florence during 2012 and captured using a Kinect camera. It includes 9 activities; 10 subjects were asked to perform the above actions for two or three times. This resulted in a total of 215 activity samples. And each frame contains 15 skeleton joints.

(2) MSR Action 3D Dataset. MSR Action 3D dataset [11] consists of the skeleton data obtained by depth sensor similar to the Microsoft Kinect. The data was captured at a frame rate of 15 frames per second. Each action was performed by 10 subjects in an unconstrained way for two or three times. The set of actions included *high arm wave*, *horizontal arm wave*, *hammer*, *hand catch*, *forward punch*, *high throw*, *draw x*, *draw tick*, *draw circle*, *hand clap*, *two-hand wave*, *side boxing*, *forward kick*, *side kick*, *jogging*, *tennis swing*, and *tennis serve*.

(3) UTKinect Action Dataset. UTKinect Action dataset [24] was captured using a single stationary Kinect and contains 10 actions. Each action is performed twice by 10 subjects in indoor setting. Three synchronized channels (RGB, depth, and skeleton) are recorded with a frame rate of 30 frames per second. The 10 actions are *walk*, *sit down*, *stand up*, *pick up*, *carry*, *throw*, *push*, *pull*, *wave hands*, and *clap hands*. It is a challenging dataset due to the huge variations in view point and high intraclass variations. So, this dataset is used to validate the effectiveness of Procrustes analysis [22].

3.2. Data Preprocessing with Procrustes Analysis.

Skeleton data in each frame of a given video usually consists of a fixed number of predefined joints. The position of joint is determined by three coordinates (x, y, z). Figure 4 shows the skeleton definition in MSR Action 3D dataset. It contains 20 joints which could be represented by their coordinates. Regarding raw human skeleton in the video as the features is not a good choice in consideration of the nature of skeleton—rotation, scaling, and translation. So, before the experiment, we should normalize the datasets by Procrustes analysis.

In statistics, Procrustes analysis is a form of statistical shape analysis used to analyze the distribution of a set of shapes and is widely applied to the field of computer vision such as face detection. In this paper, it is used to align the skeleton joints and eliminate the noise owed to rotation, scaling, or translation. Details of Procrustes analysis will be depicted next.

Given a skeleton data with k joints $((x_1, y_1, z_1), (x_2, y_2, z_2), \ldots, (x_k, y_k, z_k))$, the first step is to process the joints with translation transformation. We compute the mean coordinates $(\overline{x}, \overline{y}, \overline{z})$ of all joints and put them on the origin of coordinates. The translation is completed after each joint coordinate subtracting the mean coordinate, denoted as equation $(x_i, y_i, z_i) = (x_i - \overline{x}, y_i - \overline{y}, z_i - \overline{z})$. The purpose of scaling is making mean square root of all joint coordinates equivalent to 1. For the skeleton joints, we compute s according to the following equation:

$$s = \sqrt{\frac{x_1^2 + y_1^2 + z_1^2 + \cdots + x_k^2 + y_k^2 + z_k^2}{k}}. \quad (8)$$

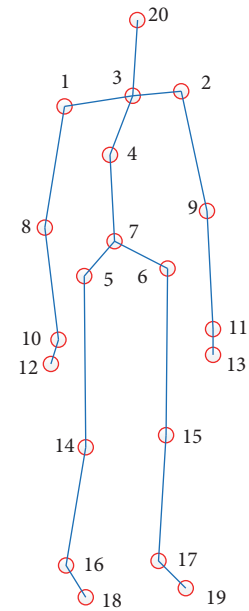


Figure 4: Skeleton of MSR Action 3D.

And the scaling result is calculated as follows: $(x_i, y_i, z_i) = (x_i/s, y_i/s, z_i/s)$. The rotation of skeleton is the last step of Procrustes analysis. Removing the rotation is more complex, as standard reference orientation is not always available. Given is a group of standard skeleton joint points $A = ((u_1, v_1, w_1), (u_2, v_2, w_2), \ldots, (u_k, v_k, w_k))$, which represent an action *stand* facing positive direction of x-coordinate axis. The mean coordinate of A is put on the origin of coordinate and the mean square root of coordinate is 1. Then we compute the rotation matrix R for skeleton $B = ((x_1, y_1, z_1), (x_2, y_2, z_2), \ldots, (x_k, y_k, z_k))$ which has been scaled and transformed as aforementioned method by (9), in which M is $3 * 3$ matrix. $U\Sigma V^T$ is the singular value decomposition with orthogonal U and V and diagonal Σ. And the rotation matrix R is equal to matrix V multiplied by the matrix transform of U. At last, skeleton joint points B can be aligned with A through computing R multiplied by B.

$$\begin{aligned} M &= B^T A \\ M &= U\Sigma V^T \\ R &= VU^T. \end{aligned} \quad (9)$$

We followed the cross-subject test setting of [30] on UTKinect dataset to test the validity of Procrustes analysis. Result is shown in Table 1. It is easy to see that the recognition rate of almost all actions is improved after preprocessing skeleton joint point with Procrustes analysis. In particular, the recognition rate of action *walk* is improved by 10%. It turned out that the translation, scaling, and rotation of human action skeleton in the video affect the recognition accuracy and Procrustes analysis is an effective method to eliminate the influence of geometry transformation.

Table 1: Results of action recognition with or without Procrustes analysis.

	Walk	Sit down	Stand up	Pick up	Carry	Throw	Push	Pull	Wave	Clap
with PA	**93%**	**86%**	**91%**	**97%**	**92%**	**88%**	87%	**91%**	**99%**	**91%**
without PA	83%	84%	85%	92%	89%	83%	87%	82%	93%	87%

3.3. Feature Extraction Method Selection. With the deep research on action recognition based on skeleton, there are many efficient feature representations. We select four of them (Pairwise [1], the most informative sequences of joint angles (MIJA) [31], histograms of 3D joints (HOJ3D) [24], and sequence of the most informative joints (SMIJ) [13]) as alternative feature representations.

Given is a skeleton $x = \{j_1, j_2, \ldots, j_n\}$, in which $j_i = (j_{x_i}, j_{y_i}, j_{z_i})$. The Pairwise representation is computed as follows: for each joint a, we extract the pairwise relative position features by taking the difference between the position of joint a and the position of another joint b: $j_{ab} = j_a - j_b$, so the feature of x is $\varphi(x) = \{j_{ab} \mid j_{ab} = j_a - j_b,\ 1 \le a < b \le n\}$. Due to the informativeness of the original joints, we made an improvement on this representation by concatenating $\varphi(x)$ and x. Then the new feature is $\varphi(x) = \{j_{a,b} \mid j_{a,b} = j_a - j_b,\ 1 \le a < b \le n\} + \{j_1, j_2, \ldots, j_n\}$.

The most informative sequences of joint angles (MIJA) representation regards joint angle as features. The shape of trajectories of joints encodes local motion patterns for each action. It chooses to use 11 out of the 20 joints capturing information for an action and center the skeleton, using the hip center joint as the origin $(0, 0, 0)$ of the coordinate system. From this origin, vectors to the 3D position of each joint are calculated. For each vector, it computes the angle θ_1 of its projection onto the x-z plane with the positive x-axis and the angle θ_2 between the vector and y-axis. The feature consists of the 2 angles of each joint.

Histograms of 3D joints (HOJ3D) representation chooses 12 discriminative joints of 20 skeletal joints. It takes the hip center as the center of the reference coordinate system and defines x-direction according to left and right hip. The remaining 8 joints are used to compute the 3D spatial histogram. The Spherical Coordinates space is partitioned to 84 bins. And for each joint location, a Gaussian weight function is used for the 3D bins. Counting the votes in each bin and concatenating them, we can get an 84-dimension feature vector.

Sequence of the most informative joints (SMIJ) representation also takes the joint angle as feature but it is different from MIJA. It partitions the joint angle time series of an action sequence into a number of congruent temporal segments and computes the variance of the joint angle time series of each joint over each temporal segment. The top 6 most variable joints in each temporal segment are selected to extract features with mapping function Φ. Here $\Phi(a) : \mathbb{R}^{|a|} \to \mathbb{R}$ is a function that maps a time series of scalar values to a single scalar value.

In order to find the optimal feature, we conduct an experiment on Florence 3D Action dataset, in which each video is short. And we estimate other 5 joints coordinates from original 15 joints of each frame in Florence dataset to make the same joints number of each frame as MSR Action 3D or UTKinect dataset. The experiment takes cross-subject test settings; one half of the dataset is used to train the key pose model and the other is used for testing. The model has 4 key poses and Procrustes analysis has been done before the feature extracting. Results are shown in Figure 5. The overall accuracy of Pairwise feature across 10 actions is better than SMIJ and MIJA. And it is observed that, for all actions except sit down and stand up, the Pairwise representation shows promising results. So, in following experiment, we select Pairwise feature to conduct action recognition experiment. The estimated joints coordinates generate more noise, so the accuracy is lower than the results on original Florence 3D Action dataset (shown in Table 6).

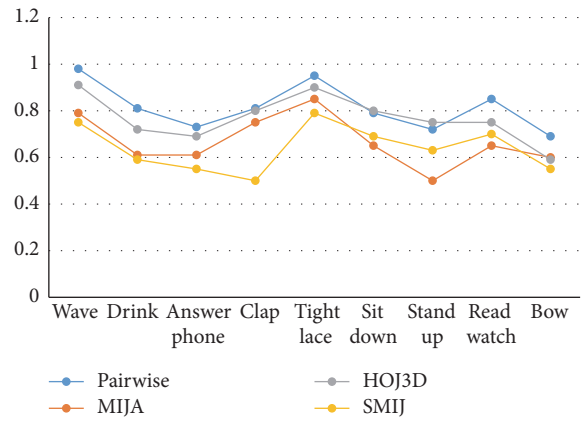


Figure 5: Features selection.

3.4. Selection of Key Pose Numbers. In this section, we implement some experiments to determine how many key poses are necessary for action recognition. The experimental results are shown in Figure 6; the horizontal axis denotes the number of key poses, and the vertical axis denotes recognition accuracy of the proposed approach. The number of key poses ranges from 1 to 20. We can see that the accuracy increases with the number of key poses when the number is less than 4. The accuracy almost achieves maximum values when the number of key poses equals 4, and the accuracy does not increase when the number of key poses is more than 4. To consider the accuracy and computation time, 4 is selected as the number of key poses for recognition action in our following experiment.

Table 2 only enumerates recognition accuracy for each action in UTKinect Action dataset when the number of key poses ranges from 4 to 8. It can be seen that the recognition accuracy varies with different key poses number for one action. However, the average recognition accuracy is nearly

Table 2: Recognition accuracy on different number of key poses.

Number	Carry	Clap	Pick	Pull	Push	Sit	Stand	Throw	Walk	Wave	Average
4	0.960	0.870	0.900	0.980	0.930	0.850	0.890	0.890	0.970	0.920	0.915
5	0.910	0.860	0.910	0.970	0.920	0.840	0.900	0.910	0.980	0.930	0.913
6	0.910	0.890	0.920	0.970	0.920	0.910	0.880	0.890	0.980	0.960	0.923
7	0.920	0.870	0.890	0.970	0.940	0.900	0.910	0.890	0.980	0.940	0.921
8	0.900	0.860	0.900	0.990	0.920	0.900	0.920	0.900	0.980	0.940	0.921

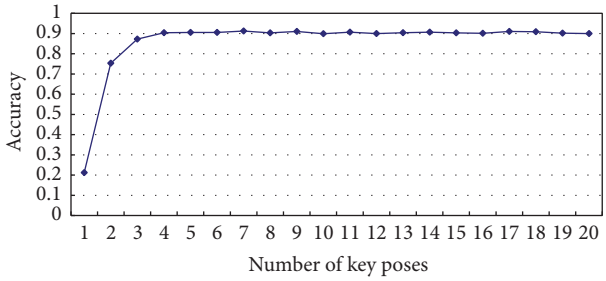


Figure 6: How many key poses does the model need?

Table 3: The three subsets of actions used in the experiments.

AS1	AS2	AS3
Bend	Draw circle	Forward kick
Forward punch	Draw tick	Golf swing
Hammer	Draw X	High throw
Hand clap	Forward kick	Jogging
High throw	Hand catch	Pick & throw
Horizontal arm wave	High arm wave	Side kick
Pickup & throw	Side boxing	Tennis serve
Tennis serve	Two-hand wave	Tennis swing

the same with different key poses number, so 4 is the high cost-effective choice.

3.5. Results on MSR Action 3D Dataset. According to the standard protocol provided by Li et al. [11], the dataset was divided into three subsets, shown in Table 3. AS1 and AS2 were intended to group actions with similar movement, while AS3 was intended to group complex actions together. For example, action *hammer* is likely to be confused with *forwardpunch* in AS1 and action *pickup* & *throw* in AS3 is a composition of *bend* and *high throw* in AS1.

We evaluate our method using a cross-subject test setting: videos of 5 subjects were used to train our model and videos of other 5 subjects were used for test procedure. Table 4 illustrates results for AS1, AS2, and AS3. We compare our performance with Li et al. [11], Xia et al. [24], and Yang and Tian [25]. We can see that our algorithm achieves considerably higher recognition rate than Li et al. [11] in all the testing setups on AS1, AS2, and AS3. For AS2, the accuracy rate of the proposed method is the highest. For AS1 or AS3, our recognition rate is only slightly lower than Xia et al. [24] or Yang and Tian [25], respectively. However, the average accuracy of our method on all three subsets is higher than the other methods.

Table 4: Comparison of our method with the others on AS1, AS2, and AS3.

Action subset	Li et al. [11]	Xia et al. [24]	Yang and Tian [25]	Ours
AS1	72.9%	**89.8**%	80.5%	89.1%
AS2	71.9%	85.5%	73.9%	**88.7**%
AS3	79.2%	63.5%	**95.5**%	94.9%
Average	74.7%	79.6%	83.3%	**90.9**%

Table 5: Comparison of our method with the others on MSR Action 3D.

MSR Action 3D	
Histogram of 3D joints [24]	78.97%
EigenJoints [25]	82.30%
Angle similarities [27]	83.53%
Actionlet [1]	88.20%
Spatial and temporal part-sets [28]	90.22%
Covariance descriptors [29]	90.53%
Our approach	**90.94**%

Table 5 shows the results on MSR Action 3D dataset. The average accuracy of the proposed method achieves 90.94%. It is easy to see that our method performs better than the other six methods.

3.6. Results on UTKinect Action Dataset. On UTKinect dataset, we followed the cross-subject test setting of [30], in which one half of the subjects is used for training our model and the other is used to evaluate the model. And we compare our model with Xia et al. [24] and Gan and Chen [30]. Figure 7 summarizes the results of our model along with competing approaches on UTKinect dataset. We can see that our method achieves the best performance on three actions such as pull, push, and throw. And the most important thing is that the average accuracy of our method achieves 91.5% and is better than the other two methods (90.9% and 91.1% for Xia et al. [24] and Gan and Chen [30], resp.). The accuracy of actions such as *clap hands* and *wave hands* is not so good; the reason may be the fact that the skeleton joint movement ranges of these actions are not large enough and the skeleton data contain more noise. So, it hinders our method from finding the optimal key poses and degrades the accuracy.

3.7. Result on Florence 3D Actions Dataset. We follow the leave-one-actor-out protocol which is suggested by dataset

Table 6: Results on Florence 3D Actions dataset.

Subject	1	2	3	4	5	6	7	8	9	10	Average
Wave	0.79	0.83	0.81	0.95	0.95	0.78	0.95	0.83	0.90	0.87	0.87
Drink	0.66	0.83	0.48	0.84	0.70	0.68	0.68	0.87	0.85	0.82	0.74
Answer	0.79	1.00	0.68	0.89	0.65	0.86	0.94	0.96	0.80	0.78	0.84
Clap	1.00	1.00	0.95	0.84	1.00	0.91	1.00	0.92	1.00	0.78	0.94
Tight	0.97	0.94	0.95	1.00	0.95	0.86	0.95	0.92	1.00	0.95	0.95
Sit down	0.72	0.89	0.90	0.90	0.76	0.86	0.79	1.00	0.80	0.91	0.85
Stand up	1.00	0.83	1.00	0.90	0.90	0.90	0.84	0.88	0.95	0.96	0.92
Read watch	0.59	0.89	0.90	0.84	0.75	0.82	0.68	0.75	0.85	0.73	0.78
Bow	0.86	0.89	0.86	1.00	0.85	1.00	1.00	0.96	0.90	0.74	0.91
Average	0.82	0.90	0.84	0.91	0.83	0.85	0.87	0.90	0.89	0.84	0.87

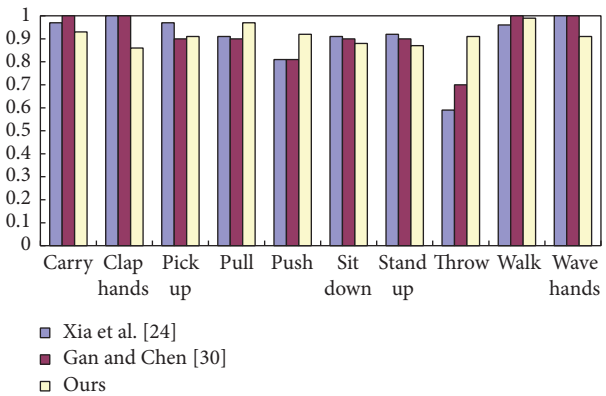


Figure 7: Results on UTKinect Action dataset.

collector on original Florence 3D Action dataset. All the sequences from 9 out of 10 subjects are used for training, while the remaining one is used for testing. For each subject, we repeat the procedure and average the 10 classification accuracy values at last. For comparison with other methods, average action recognition accuracy is also computed. The experimental results are shown in Table 6. In each column, the data represent each action's recognition accuracy, while the corresponding subject is used for testing. The challenges of this dataset are the human-object interaction and the different ways of performing the same action. By analyzing the experiment result of our method, we can notice that the proposed approach obtains high accuracies for most of the actions. Our method overcomes the difficulty of intraclass variation such as bow and clap. The proposed approach gets lower accuracies for the actions such as answer the phone and read watch; this can be explained by the fact that these actions are human-object interaction with small range of motion and the Pairwise feature could not well reflect the motion. Furthermore, results compared with other methods are listed in Table 7. It is clear that our average accuracy is better than Seidenari et al. [15] and is the same as Devanne et al. [16].

Table 7: Comparing of our method with the others on Florence 3D Actions dataset.

Florence 3D Actions		
Seidenari et al. [15]	Devanne et al. [16]	Our approach
82%	**87**%	**87**%

4. Conclusion

In this paper, we presented an approach for action recognition based on skeleton by mining the key skeleton poses with latent SVM. Experimental results demonstrated that human actions can be recognized by only a few frames with key skeleton pose; in other words, a few inconsecutive and representative skeleton poses can describe the video action. Starting from feature extraction using the pairwise relative positions of the joints, the positions of key poses are found with the help of latent SVM. Then the model is iteratively trained with positive and negative video examples. In test procedure, a simple method is given by computing the score of each start position to recognize the action.

We validated our model on three benchmark datasets: MSR Action 3D dataset, UTKinect Action dataset, and Florence 3D Action dataset. Experimental results demonstrated that our method outperforms all other methods. Because our method relies on extracting descriptors of simple relative positions of the joints, its performance degrades when the actions are little varied and uninformative, for instance, those actions that were performed only by forearm gestures such as *clap hands* in UTKinect Action dataset. In the future, we will explore the other local features reflecting minor motion for better understanding human action.

Competing Interests

The authors declare that there are no competing interests regarding the publication of this paper.

References

[1] J. Wang, Z. Liu, Y. Wu, and J. Yuan, "Learning actionlet ensemble for 3D human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 5, pp. 914–927, 2014.

[2] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '08)*, pp. 1–8, June 2008.

[3] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Dense trajectories and motion boundary descriptors for action recognition," *International Journal of Computer Vision*, vol. 103, no. 1, pp. 60–79, 2013.

[4] A. Klaser, M. Marszalek, and C. Schmid, "A spatio-temporal descriptor based on 3d-gradients," in *Proceedings of the 19th British Machine Vision Conference (BMVC '08)*, p. 275, British Machine Vision Association, 2008.

[5] K. G. Derpanis, M. Sizintsev, K. J. Cannons, and R. P. Wildes, "Action spotting and recognition based on a spatiotemporal orientation analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 3, pp. 527–540, 2013.

[6] S. Sadanand and J. J. Corso, "Action bank: a high-level representation of activity in video," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 1234–1241, IEEE, Providence, RI, USA, June 2012.

[7] A. Fathi and G. Mori, "Action recognition by learning mid-level motion features," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '08)*, pp. 1–8, IEEE, June 2008.

[8] M. Raptis, I. Kokkinos, and S. Soatto, "Discovering discriminative action parts from mid-level video representations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, June 2012.

[9] K. Schindler and L. Van Gool, "Action snippets: how many frames does human action recognition require?" in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '08)*, pp. 1–8, June 2008.

[10] "kinect—australia," http://www.xbox.com/en-AU/Kinect.

[11] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3D points," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW '10)*, pp. 9–14, IEEE, San Francisco, Calif, USA, June 2010.

[12] J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu, "Robust 3D action recognition with random occupancy patterns," in *Computer Vision—ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7–13, 2012, Proceedings, Part II*, pp. 872–885, Springer, Berlin, Germany, 2012.

[13] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy, "Sequence of the most informative joints (SMIJ): a new representation for human skeletal action recognition," *Journal of Visual Communication and Image Representation*, vol. 25, no. 1, pp. 24–38, 2014.

[14] M. Müller and T. Röder, "Motion templates for automatic classification and retrieval of motion capture data," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '06)*, pp. 137–146, Vienna, Austria, September 2006.

[15] L. Seidenari, V. Varano, S. Berretti, A. Del Bimbo, and P. Pala, "Recognizing actions from depth cameras as weakly aligned multi-part bag-of-poses," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW '13)*, pp. 479–485, Portland, Ore, USA, June 2013.

[16] M. Devanne, H. Wannous, S. Berretti, P. Pala, M. Daoudi, and A. Del Bimbo, "3-D human action recognition by shape analysis of motion trajectories on riemannian manifold," *IEEE Transactions on Cybernetics*, vol. 45, no. 7, pp. 1340–1352, 2015.

[17] T. Batabyal, T. Chattopadhyay, and D. P. Mukherjee, "Action recognition using joint coordinates of 3D skeleton data," in *Proceedings of the IEEE International Conference on Image Processing (ICIP '15)*, pp. 4107–4111, IEEE, Québec, Canada, September 2015.

[18] Y. Du, Y. Fu, and L. Wang, "Representation learning of temporal dynamics for skeleton-based action recognition," *IEEE Transactions on Image Processing*, vol. 25, no. 7, pp. 3010–3022, 2016.

[19] B. Mahasseni and S. Todorovic, "Regularizing long short term memory with 3D human-skeleton sequences for action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '16)*, pp. 3054–3062, Las Vegas, NV, USA, June 2016.

[20] X. Li and Q. Yao, "Action detection based on latent key frame," in *Biometric Recognition*, pp. 659–668, Springer, Berlin, Germany, 2015.

[21] W. Zhang, M. Zhu, and K. G. Derpanis, "From actemes to action: a strongly-supervised representation for detailed action understanding," in *Proceedings of the 14th IEEE International Conference on Computer Vision (ICCV '13)*, Sydney, Australia, December 2013.

[22] J. C. Gower, "Generalized procrustes analysis," *Psychometrika*, vol. 40, no. 1, pp. 33–51, 1975.

[23] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[24] L. Xia, C.-C. Chen, and J. K. Aggarwal, "View invariant human action recognition using histograms of 3D joints," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW '12)*, pp. 20–27, June 2012.

[25] X. Yang and Y. Tian, "Effective 3D action recognition using EigenJoints," *Journal of Visual Communication and Image Representation*, vol. 25, no. 1, pp. 2–11, 2014.

[26] "Florence 3d actions dataset," http://www.micc.unifi.it/vim/datasets/3dactions/.

[27] C. Wang, Y. Wang, and A. L. Yuille, "An approach to pose-based action recognition," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '13)*, pp. 915–922, IEEE, Portland, Ore, USA, June 2013.

[28] C. Wang, Y. Wang, and A. L. Yuille, "An approach to pose-based action recognition," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '13)*, pp. 915–922, Portland, Ore, USA, June 2013.

[29] M. E. Hussein, M. Torki, M. A. Gowayyed, and M. El-Saban, "Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations," in *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI '13)*, pp. 2466–2472, Beijing, China, August 2013.

[30] L. Gan and F. Chen, "Human action recognition using APJ3D and random forests," *Journal of Software*, vol. 8, no. 9, pp. 2238–2245, 2013.

[31] H. Pazhoumand-Dar, C.-P. Lam, and M. Masek, "Joint movement similarities for robust 3D action recognition using skeletal data," *Journal of Visual Communication and Image Representation*, vol. 30, article no. 1493, pp. 10–21, 2015.

Hindawi
Applied Computational Intelligence and Soft Computing
Volume 2017, Article ID 5834846, 9 pages
http://dx.doi.org/10.1155/2017/5834846

Research Article

Reidentification of Persons Using Clothing Features in Real-Life Video

Guodong Zhang,[1] Peilin Jiang,[2] Kazuyuki Matsumoto,[1] Minoru Yoshida,[1] and Kenji Kita[1]

[1]*Faculty of Engineering, Tokushima University, Tokushima 7708506, Japan*
[2]*Xian Jiao Tong University, No. 28, Xianning West Road, Xian, China*

Correspondence should be addressed to Guodong Zhang; zhang-g@hotmail.co.jp

Received 16 August 2016; Revised 6 November 2016; Accepted 24 November 2016; Published 11 January 2017

Academic Editor: Qiushi Zhao

Person reidentification, which aims to track people across nonoverlapping cameras, is a fundamental task in automated video processing. Moving people often appear differently when viewed from different nonoverlapping cameras because of differences in illumination, pose, and camera properties. The color histogram is a global feature of an object that can be used for identification. This histogram describes the distribution of all colors on the object. However, the use of color histograms has two disadvantages. First, colors change differently under different lighting and at different angles. Second, traditional color histograms lack spatial information. We used a perception-based color space to solve the illumination problem of traditional histograms. We also used the spatial pyramid matching (SPM) model to improve the image spatial information in color histograms. Finally, we used the Gaussian mixture model (GMM) to show features for person reidentification, because the main color feature of GMM is more adaptable for scene changes, and improve the stability of the retrieved results for different color spaces in various scenes. Through a series of experiments, we found the relationships of different features that impact person reidentification.

1. Introduction

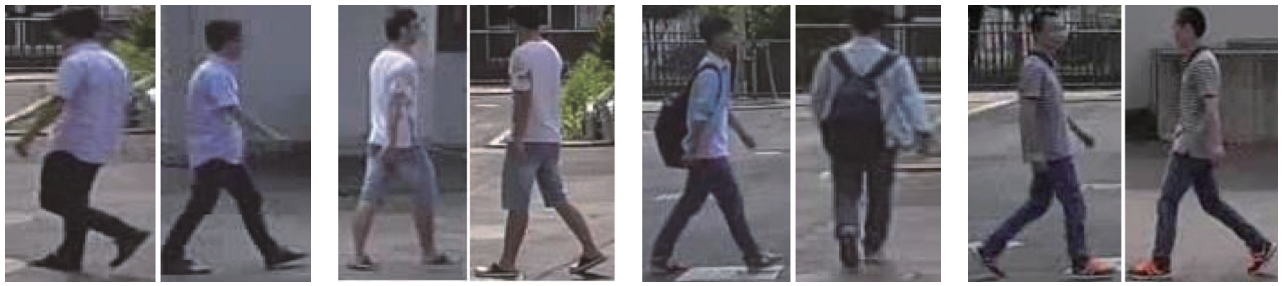
As public security technology has become increasingly intelligent, surveillance cameras have been set up in public places such as airports and supermarkets. These cameras provide huge amounts of nonoverlapping video data. It is often necessary to track an object or person of interest that appears on video from multiple cameras under different illumination conditions [1–3]. When searching for moving people in surveillance video data, object retrieval systems for intelligent video surveillance experience the following problems.

(1) Object retrieval results in video surveillance depend on motion segmentation and video analysis. Digital video is a series of images, constituted by frames that contain rich information. If an image frame contains moving objects, then object retrieval detection can be used to segment a moving target [4]. Object retrieval results depend on the object segmentation. If video analysis cannot separate the foreground and moving objects, the target object cannot be retrieved from the many irrelevant foreground objects. A good object retrieval system should adapt to various levels of video quality for foreground detection, which could eliminate unrelated objects and retrieve the target [5].

(2) Specific object retrieval in video surveillance faces technical limitations. The moving objects of interest in surveillance video are often persons and cars. Facial features are the most distinctive elements for person recognition, and relatively mature methods are available for this process. However, low camera resolution often makes it difficult to extract perceivable information about facial expression [6]. The mature technology of video object retrieval based on facial features should receive more technical exploration.

(3) External factors greatly influence objects appearance under video surveillance. A robust object retrieval system should be able to compensate for the following factors.

    (i) Person pose variation: a moving person may have arbitrary poses (Figure 1(a)).



(a) (b) (c) (d)

Figure 1: Images showing the same person in different camera views: (a) pose change, (b) illumination change, (c) occlusion, and (d) low resolution.

(ii) Varying illumination conditions: illumination conditions usually differ between camera views (Figure 1(b)).

(iii) Occlusion: a person body parts may be occluded by other subjects, such as a carried bag, in one camera view (Figure 1(c)).

(iv) Low image resolution: due to surveillance camera performance, images of a moving person often have low resolution (Figure 1(d)).

The color histogram is a tool used to describe the color composition of an image [7]. The histogram shows the appearance of different colors and the number of pixels for each color in an image. Colors possess better immunity to the noise jamming of images and are robust against image degradation and scaling. We selected a global color approach to body features for person reidentification in surveillance video. Extracting the color information of the person makes the method clear and simple. Because color statistic features lose information about color spatial distribution, we combined this approach with the spatial pyramid matching (SPM) model. We tested our method in the RGB, HSV, and UVW color spaces using real video images. We present related work on person reidentification and feature analysis in Section 2. We offer details on our proposed method in Section 3. We report and discuss the experimental results in Section 4, and we give conclusions and suggestions for future work in Section 5.

2. Related Works

For the past few years, object retrieval techniques using content-based video retrieval have received significant theoretical and technological support. Many researchers have examined person reidentification, and the related literature is extensive [8, 9]. This section discusses feature modeling and effective matching strategies, which are important methods for person reidentification.

2.1. Color Feature. Color features are one of the low-level feature types that have been widely used in content-based image retrieval (CBIR). Compared with other features, color exhibits little dependence on image rotation, translation, scale change, and even the shape change. Color is thus thought of as almost independent of the images dimensions, direction, and view angles. Most representations in previous approaches are based on appearance. Gray and Tao [10] used a similarity function that was trained from a set of data. These authors focused on the problems of unknown viewpoint and pose. The method is robust to viewpoint change because it is based on the ensemble of localized features (ELF). Farenzena et al. [11] presented an appearance-based method based on the localization of perceptually relevant human parts. The information features contain three parts: overall chromatic content, the spatial arrangement of colors into stable regions, and the presence of recurrent local motifs with high entropy. The method is robust to pose, viewpoint, and illumination variations. Zhao et al. [12] transformed person reidentification into a distance learning problem. Using the relative distance comparison model to compute the distance of a pair of views, these authors considered a likely true match pair to have a smaller distance than that of a wrong match pair. These authors also used a new relative distance comparison model to measure the distance between pairs of person images and judge the pairs of true matches and wrong matches. Angela et al. proposed a new feature based on the definition of the probabilistic color histogram and trained fuzzy k-nearest neighbors (KNN) classifier based on an ad hoc dataset. The method is effective at discriminating and reidentifying people across two different video cameras regardless of viewpoint change. Metternich et al. [13] used a global color histogram and shape information to track people in real-life surveillance data, finding that the appearance of the subject impacted the tracking results. These authors also focused on the performance of matching techniques over cameras with different fields of view.

2.2. Metric Learning. Hirzer et al. [14] focused the matching method of metric learning on person reidentification. These authors accomplished metric learning from pairs of samples from different cameras. The method benefits from the advantages of metric learning and reduces the required computational effort. Good performance can be achieved even using less color and texture information. Khedher et al. [15] proposed a new automatic statistical method that could accept and reject SURF correspondence based on the

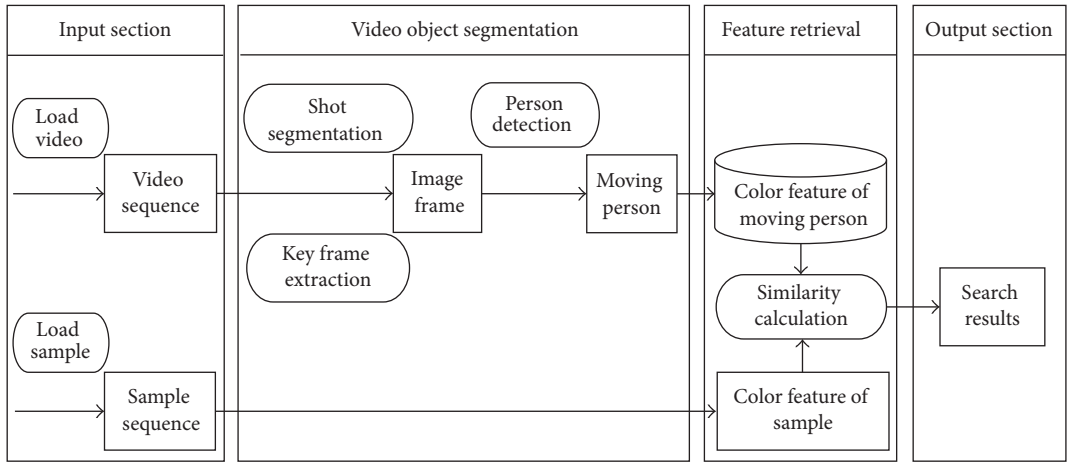


Figure 2: Overview of the system.

likelihood ratio of two Gaussian mixed models (GMMs) learned on a reference set. The method does not need to select the matching SURF pairs by empirical means. Instead, interest point matching over whole video sequences is used to judge the person identity. Matsukawa et al. [16] focused on the problem of overfitting and proposed a discriminative accumulation method of local histograms for person reidentification. The proposed method jointly learns pairs of a weight map for the accumulations and employs a distance metric that emphasizes discriminative histogram dimensions. This method can achieve better reidentification accuracy than other typical metric learning methods on various sizes of datasets.

3. System Description

3.1. An Overview of the Proposed System. The techniques of moving person retrieval information from a video database include shot segmentation, person detection, scene segmentation, feature extraction, and similarity calculation. As shown in Figure 2, shot segmentation refers to automatically segmenting video clips into shots as the basic unit for indexing. One second of video contains about 20–30 video frames, and neighboring frames are very similar to each other. There is no need to perform retrieval and matching for each frame, and frame differentiation is used to detect and extract the moving person. Frame differentiation relies on the change of pixel value between neighboring key frames. A change value greater than the established threshold value marks the pixel position of the moving person. This step is important in video parsing and directly affects the effectiveness of moving person retrieval.

The measurement method for similarity calculation influences the results ranking of object retrieval. Essentially, image similarity calculation computes the content of feature vectors from the objects. Each feature attribute selection can employ a different similarity computing method [17]. Frequently, image features are extracted in the form of feature vectors that can be regarded as points in multidimensional space. The most common similarity measure method uses the distance between two spots in feature space. We also use distance measurement and correlativity calculation to scale the comparability between images.

Our proposed method is presented in Figure 3. We use traditional histogram and SPM histogram to retrieve the object. The traditional histogram method contains three parts, the color histogram feature extraction, color histogram distance computing, and outputting. The difference between SPM histogram and traditional histogram is the histogram distance computing part. The sample image and matching image are segmented into three parts, the upper, middle, and lower part. The three parts then separately computed the color histogram distance and use average distance to evaluate the results. Then the system uses GMM model to filter the top 20 results, extracts the GMM main color feature, and computes the similarity of them. Finally, the system outputs the rank of top 10 results.

3.2. Perception-Based Color Space Histogram Feature. Computations in the RGB and HSV color spaces cannot solve the problem of background illumination sensitivity. The color spaces always affect the computing accuracy of the color histogram [18]. We attempted to use perception-based color space, which exhibits good performance in image processing [19]. As the name suggests, the perception-based color space associated metric approximates perceived distances and color displacements, capturing relationships that are robust to spectral changes in illumination [20]. RGB color space can be transformed to perception-based color space through the following steps.

RGB color space can be transformed to perception-based color space through the following steps.

(1) Transform RGB to XYZ color space using the following formula (1):

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \frac{1}{0.177} \begin{pmatrix} 0.49 & 0.361 & 0.20 \\ 0.177 & 0.0812 & 0.011 \\ 0.00 & 0.01 & 0.99 \end{pmatrix} \begin{bmatrix} G(R) \\ G(G) \\ G(B) \end{bmatrix}, \quad (1)$$

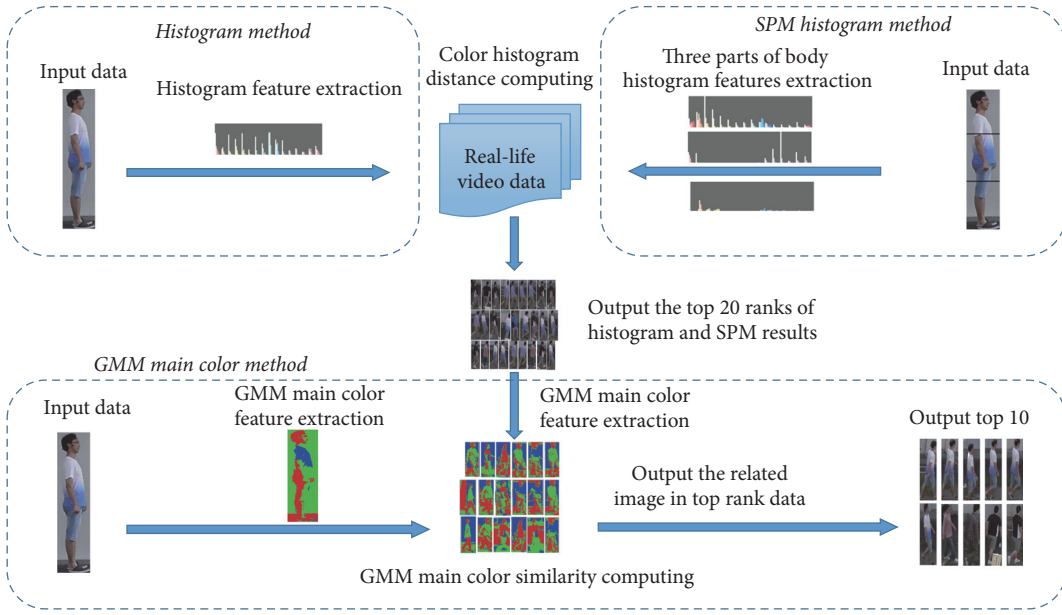


Figure 3: Overview of proposed method.

where $G(\,)$ is the gamma correction function and equals 2.0. The gamma correction function addresses color distortion and rediscovers the real environment to a certain extent.

(2) Transform XYZ to UVW color space. In UVW color space, the influence of lighting conditions is simulated by the tristimulus multiplication values and scale factor, as shown in the following formula (2):

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \longrightarrow \begin{bmatrix} U \\ V \\ W \end{bmatrix} = B^{-1}DB \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}, \tag{2}$$

where D is a diagonal matrix, accounting only for illumination, and independent of the material. B is the transfer matrix from the current color space coordinates to the base coordinates. The nonlinear transfer uses the following formula (3):

$$\begin{bmatrix} U \\ V \\ W \end{bmatrix} = A \left(\widehat{\ln} \left(B \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \right) \right), \tag{3}$$

where A and B are invertible 3×3 matrices and denote the component-wise natural logarithm. Matrix B transforms the color coordinates to the basis in which relighting best corresponds to multiplication by a diagonal matrix, while matrix A provides degrees of freedom that can be used to match perceptual distances. Based on similar color experiments in the database, A and B matrix-value formulas are shown as (4) and (5), respectively.

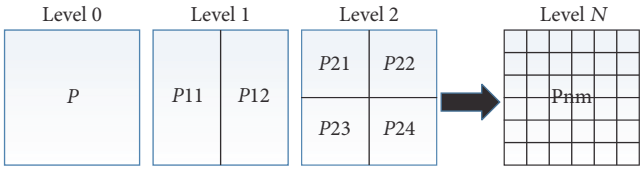


Figure 4: The method of SPM.

$$A = \begin{pmatrix} 27.07439 & -22.80783 & -1.806681 \\ -5.646736 & -7.722125 & 12.86503 \\ -4.163133 & -4.579428 & -4.576049 \end{pmatrix}, \tag{4}$$

$$B = \begin{pmatrix} 0.9465229 & 0.2946927 & -0.1313419 \\ -0.117917 & 0.9929960 & 0.007371554 \\ 0.0923046 & -0.046457 & 0.9946464 \end{pmatrix}. \tag{5}$$

3.3. SPM Model. Lazebnik et al. [21] proposed the Spatial Pyramid Matching (SPM) in 2006. SPM model contains broad space information, with which the color histogram information will be encoded orderly in space. The model divides the image into different levels, which can then be further refined. The SPM model space is shown in Figure 4. The level 0 image P is based on the original image feature information. But the image feature is based on the global unordered color information. Level 1 shows image separated as space geometry. $P11$ and $P12$ are expressed by a spatial order that contains simple space information.

P11 and P12, which also lack internal space information, are in level 1. If internal space information is necessary in P11 and P12, they must be separated using the same process. The level $i + 1$ feature is divided by level i. The levels of division are decided by the actual situation.

3.3.1. The SPM Histogram Feature. Image similarity is computed by the levels corresponding to parts in SPM model. For two images P and Q, the formula is as follows:

$$d(P,Q) = \sum k_{ij} d\left(p_{ij}, q_{ij}\right), \tag{6}$$

where P_{ij} is the image P histogram feature of the part j in level i; $d(p_{ij}, q_{ij})$ is the feature similarity degree images P and Q; and K_{ij} is the weight of the similarity calculation. In this case, we focus on part j of level i. The weight of calculation should be set high.

3.4. Gaussian Color Model. Gaussian color model (GMM) is constantly used for color image segmentation according to the classification and clustering of image characteristics [22]. The image is divided into different parts based on pixel classification. We considered the main part of person identification to be based on minutia matching and ignored details. The retrieval of similar objects in a video system prioritizes the main part of similarity matching and does not emphasize accurate detail matching, so we considered the main colors as the features of the Gaussian color model.

3.4.1. Gaussian Distribution. The Gaussian distribution is a parametric probability density function that is a mean value and variance continuous distribution maximum information entropy [23]. As shown in (7), when distributing a unit value that fits the normal distribution random variable, the frequency of the variable that follows the Gaussian distribution is entirely determined by the mean value μ and variance σ^2. As x approaches μ, probability increases. σ means the dispersion, and the value of σ is a much greater degree of dispersion.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}. \tag{7}$$

For an image, the Gaussian distribution describes the distribution of specific pixel brightness that reflects the frequency of some gray numerical value [24]. A single-mode Gaussian distribution cannot represent a multicolored image. Therefore, we used a multiplicity of Gaussian models to show different pixel distributions that approximately simulate a multicolored image. Theoretically, we could increase the numbers of models to improve the descriptive ability.

Every pixel of the color image could be represented as a d dimensional vector x_i (color image $d = 3$ and gray image $d = 1$). The whole image could be represented as $X = (x_1^T, x_2^T, \ldots, x_N^T)$, where N is the sum of all pixels in a picture, X is represented as M states in GMM, and the value of M is usually restricted from 3 to 5. The linear stacking of the M Gaussian distributions could show the GMM of the probability density function, as shown in (8): x is the pixel sampling of a picture.

$$P(x) = \sum_{k=1}^{M} p(k)\, p(x \mid k) = \sum_{k=1}^{M} \pi(k)\, N\left(x \mid \mu_k, \sum k\right). \tag{8}$$

$N(x \mid \mu_k, \sum k)$ is the single Gaussian density function. As shown in (8), $k = 1, \ldots, M$ indicates the Gaussian density function of *No.k*. μ_k is the sample mean vector, $\sum k$ is sample covariance matrix, and π_k is the nonnegative coefficient of weight that describes the proportion of *No.k* data in the total data.

3.5. Color Histogram Feature Extraction. The histogram of an image is related to the probability distribution function of the images pixel density. When this concept is extended to a color image, it is necessary to obtain the joint probability distribution value for multiple channels [25]. In general, a color histogram is defined by the following equation (9):

$$h_{A,B,C} = N \bullet \text{Prob}(A = a,\ B = b,\ C = c), \tag{9}$$

where A, B, and C indicate three color channels (R, G, and B or H, S, and V) and N is the sum of all pixels in the image. In terms of computing, the first step is to discretize the pixel values of the image, creating statistics for the number of pixels of each color for color histogram.

3.6. Histogram of Color Feature Similarity Measurement. Several methods exist to calculate and weigh the similarity measurement of the histogram. The distance formula of the similarity measure between images is based on the color content. Euclidean distance, histogram intersection, and histogram quadratic distance are widely used in image retrieval.

The Euclidean distance of the histogram between two images is given by the following equation (10):

$$d^2(h,g) = \sum_A \sum_B \sum_C \left(h(a,b,c) - g(a,b,c)\right)^2, \tag{10}$$

where h and g are two histograms and a, b, and c are the color channels. The formula subtracts the pixel value in the same bin of histograms h and g.

The formula for histogram intersection distance is as follows:

$$d(h,g) = \frac{\sum_A \sum_B \sum_C \min\left(h(a,b,c) - g(a,b,c)\right)}{\min\left(|h|, |g|\right)}, \tag{11}$$

where $|h|$ and $|g|$ stand for the pixel values of image sampling in histograms h and g, respectively.

3.7. Evaluation Method. (1) We focused on the degree of search result accuracy using evaluation parameters for precision. Precision reflects the capability of filtering irrelevant content. These video retrieval system performance criteria reference the evaluation method for information search systems. For a retrieval object, the retrieval system returns a

sort of search results. The precision rate expresses the number of correct relevant retrieval results divided by the number of total retrieval results.

$$\text{Precision}\,(\%) = \frac{A}{A+B} \times 100,$$
$$\text{AveragePrecision}\,(\%) = \frac{1}{n}\sum_{i=1}^{n}\text{Precision}\,(i)\,. \tag{12}$$

In formula (12), A is the number of correct relevant retrieval examples, B is the number of irrelevant video retrieval examples, and C is the number of missing correct relevant retrieval examples.

(2) Cumulative Match Characteristic (CMC) curve is employed to evaluate the performance of the reidentification system. The CMC curve is used when the full gallery is available. It depicts the relationship between the accuracy and the threshold of rank. Most of the existing pedestrian reidentification algorithms use the CMC curve to evaluate the algorithm performance. Given a probe set and a pedestrian gallery set, the experimental result of CMC analysis describes what is the percentage of probe searches in the pedestrian dataset that returns the probes gallery mate within the top r rank-ordered results.

4. Experiment

We evaluate our reidentification method on three datasets, that is, the multicamera video data, the VIPeR data, and the SARC3D data. We examine our proposed SPM histogram + GMM main color method, the SPM histogram method, and the traditional histogram method on three datasets and further compare our method with the Symmetry-Driven Accumulation of Local Features (SDALF) method on the public VIPeR and SARC3D datasets. The code of SDALF could be downloaded on https://github.com/lorisbaz/sdalf. All the experiments are run on a desktop computer with an i7-3.4 GHz CPU.

4.1. Experiment on Multicamera Videos. We evaluated the performance of different color spaces for real-life video data. Uneven illumination distribution should affect person reidentification results in color images. Therefore, we created a video data set to test the validity and robustness of our method. We recorded the video data on a school campus. Six pedestrians walked from left to right in order under a surveillance camera, as shown in Figures 5 and 6. Our real-life video data consists of two videos that were recorded simultaneously at different locations. Location 1 was bright and location 2 was dark. The videos were recorded at 25 frames per second. Pictures of the side viewpoints of the six pedestrians were used as the retrieval samples, as shown in Figure 7. The six pedestrians were without a hat, bag, or other accessories. The RGB results are based on machine vision, while the HSV results are closer to human visual perception. As shown in Table 1, our proposed method outperforms the traditional histogram method and the SPM histogram method. We find that although the RGB color space reflects all sorts of colors from the images, the background color which is mixed in these channels has affected the reidentification result. This problem is even severe in the SPM method, in which the lower part of the separated image contains a greater part of the background color than the body color. As shown in Table 2, the performance of UVW is better than HSV and RGB. The reason is that the results were affected mostly by the color transfer. In different illumination, the color histogram of one's clothes would be transferred to another color. For example, the red color in a dark environment seems like a black or gray color. The UVW color space is aimed at this problem. In the GMM color modeling, to solve the color transfer problem in low resolution images, we employ the primary colors of red, blue, and green as the dominant colors. However, for the dark background images, the GMM method generates a poor result.

Table 1: The average precision for persons retrieval in location 1.

Method	RGB	HSV	UVWS
Histogram	75	73.33	80
SPM histogram	71.66	75	70
GMM	86.66	85	88.33

Table 2: The average precision for persons retrieval in location 2.

Method	RGB	HSV	UVWS
Histogram	73.33	75	81.66
SPM histogram	83.33	85	80
GMM	81.66	83.33	85

4.2. Experiment on VIPeR Dataset. We examine the appearance model for person reidentification based on the VIPeR dataset, which consists of 632 pedestrian image pairs taken from arbitrary viewpoints under varying illumination conditions. Each image is scaled to 128 × 48 pixels.

As shown in Figure 8, our proposed method outperforms the histogram-based methods in the RGB color space, and the traditional histogram and the SPM histogram methods generate very similar results. We also observe that the proposed method in the HSV space performs better than in the RGB space, as shown in Figure 9. This is because that the image illumination in the VIPeR dataset varies significantly. The SDALF method renders a slightly better result than our proposed method, while our method has a great advantage on the calculation cost. Specifically, the SDALF takes about 3850 seconds to extract its features from 1264 images in the VIPeR dataset, while our proposed method takes only 40 seconds to extract and calculate the color histogram features. In addition, the SDALF method needs about 4260 seconds to compare all 399424 pairs of images, while our method needs only 610 seconds to calculate the GMM similarity for comparison in 1264 images. This result suggests that in terms of computational cost our approach significantly outperforms the SDALF method.

4.3. Experiment on SARC3D Dataset. The SARC3D dataset consists of short video clips of 50 people which have been

Figure 5: Location 1.

Figure 6: Location 2.

Figure 7: Example of placing a figure with experimental results.

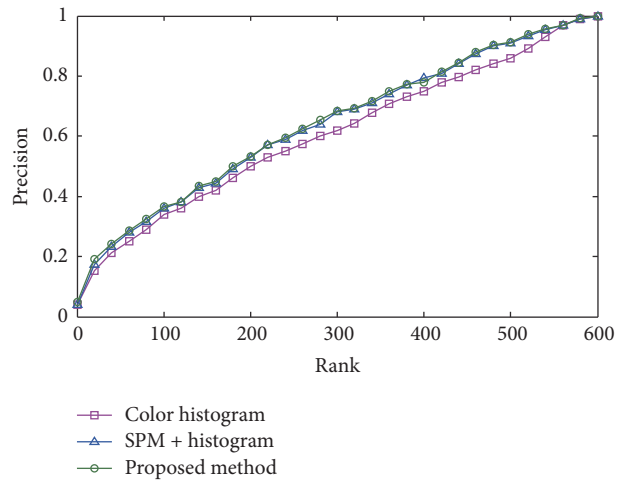


Figure 8: CMC curves on the VIPeR dataset for the proposed method and histogram methods in RGB space.

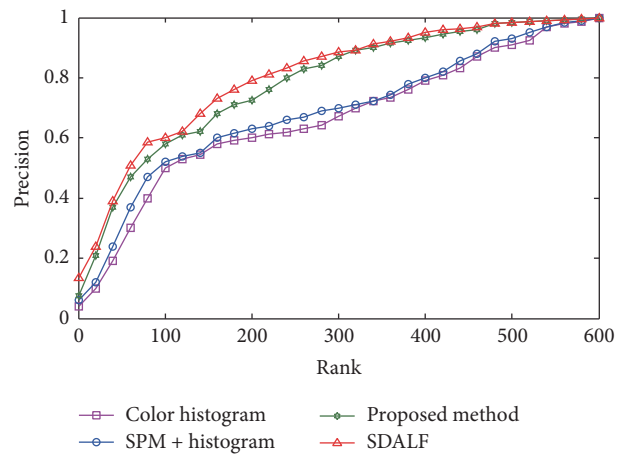


Figure 9: CMC curves on the VIPeR dataset for the proposed method and the other methods in HSV space.

captured with a calibrated camera. We employ the SARC3D dataset to effectively evaluate different person reidentification methods. To simplify the image alignment process, we manually select four frames for each clip which correspond to the predefined positions and postures, that is, back, front, left, and right, of these people. The selected dataset consists of 200 snapshots with four views for each person. For person reidentification, we randomly choose one of the four views for each person, calculate the similarity scores with all other images, and find the most similar images by sorting their similarities with the chosen image. The images of the same person with different positions and postures should be ranked higher than the other images. In the dataset, 6 people are not fully visible in their images and 2 people are observed with the same dressing, that is, colors and combinations, except for the waling postures. We remove images of these people to avoid the different size of their masks form in the original images. All methods in the experiment are based on the RGB color space. Figure 10 shows the average CMC curves for the person reidentification under different methods. Our method significantly outperforms the SDALF method in recognition rate because the backward information in GMM matching has been filtered out given the people annotation template in the dataset. In the meantime, our method significantly outperforms the SDALF method in calculation cost, with only 30 seconds for color histogram feature extraction and image matching in 126 images, while the latter takes about 440 seconds for feature extraction and 70 more seconds for image matching.

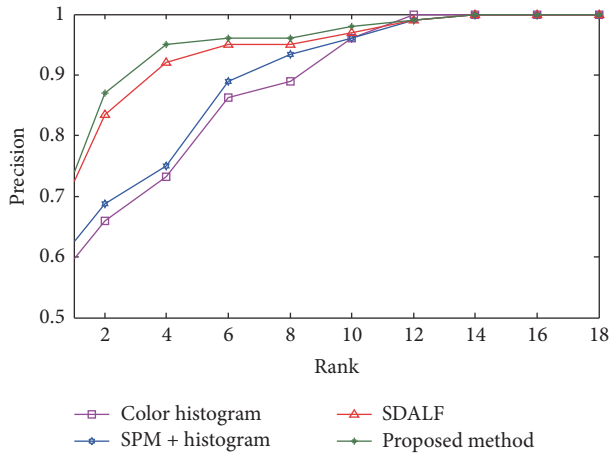


Figure 10: CMC curves on the VIPeR dataset for the proposed method and the other methods in RGB space.

5. Conclusion

Person reidentification in multicamera videos often has some problems that contain person pose variation, varying illumination, and low image resolution. We propose to solve two common problems in person reidentification, which are the varying illumination and low image resolution. Varying illumination conditions usually occur because of the difference between camera views. For example, the same people in different camera video have a color transfer. The low resolution image often contains high noise. It is difficult to extract the robust feature from the low resolution image. In order to improve the illumination problem in histogram methods, we introduce the perception-based color space which has been successfully employed in the image segmentation research into the person identification method. Secondly, for the low resolution images we incorporate spatial pyramid matching (SPM) method into the main color extraction method, which has shown great improvement in our experiment. In addition, our method has shown significant advantage in the computation cost compared with the traditional methods. In this paper we just extract the main color feature by the GMM model. We did not analyse the feature information from the mean value parameter and variance in the GMM. The main color feature also used the global object color; we could combine the SPM model with GMM main color local feature to retrieve the object from the video data.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This research was partially supported by JSPS KAKENHI Grant nos 15K00425 and 15K00309.

References

[1] R. Satta, "Appearance descriptors for person reidentification: a comprehensive review," https://arxiv.org/abs/1307.5748.

[2] A. Dangelo and J.-L. Dugelay, "People re-identification in camera networks based on probabilistic color histograms," in *Visual Information Processing and Communication II*, vol. 7882 of *Proceedings of SPIE*, January 2011.

[3] G. Doretto, T. Sebastian, P. Tu, and J. Rittscher, "Appearance-based person reidentification in camera networks: problem overview and current approaches," *Journal of Ambient Intelligence and Humanized Computing*, vol. 2, no. 2, pp. 127–151, 2011.

[4] M. A. Saghafi, A. Hussain, H. B. Zaman, and M. H. Md Saad, "Review of person re-identification techniques," *IET Computer Vision*, vol. 8, no. 6, pp. 455–474, 2014.

[5] S. Lee, N. Kim, K. Jeong, I. Paek, H. Hong, and J. Paik, "Multiple moving object segmentation using motion orientation histogram in adaptively partitioned blocks for high-resolution video surveillance systems," *Optik—International Journal for Light and Electron Optics*, vol. 126, no. 19, pp. 2063–2069, 2015.

[6] A. Bedagkar-Gala and S. K. Shah, "A survey of approaches and trends in person re-identification," *Image and Vision Computing*, vol. 32, no. 4, pp. 270–286, 2014.

[7] R. Vezzani, D. Baltieri, and R. Cucchiara, "People reidentification in surveillance and forensics: a survey," *ACM Computing Surveys*, vol. 46, no. 2, article no. 29, 2013.

[8] X. Wang and R. Zhao, "Person re-identification: system design and evaluation overview," in *Person Re-Identification*, pp. 351–370, Springer, 2014.

[9] B. Ma, Q. Li, and H. Chang, "Gaussian descriptor based on local features for person re-identification," in *Proceedings of the Asian Conference on Computer Vision (ACCV '14)*, pp. 505–518, Springer, Singapore, November 2014.

[10] D. Gray and H. Tao, "Viewpoint invariant pedestrian recognition with an ensemble of localized features," in *Computer Vision—ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12–18, 2008, Proceedings, Part I*, vol. 5302 of *Lecture Notes in Computer Science*, pp. 262–275, Springer, Berlin, Germany, 2008.

[11] M. Farenzena, L. Bazzani, A. Perina, V. Murino, and M. Cristani, "Person re-identification by symmetry-driven accumulation of local features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '10)*, pp. 2360–2367, IEEE, San Francisco, Calif, USA, June 2010.

[12] R. Zhao, W. Ouyang, and X. Wang, "Unsupervised salience learning for person re-identification," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '13)*, pp. 3586–3593, IEEE, Portland, Ore, USA, June 2013.

[13] M. J. Metternich, M. Worring, and A. W. M. Smeulders, "Color based tracing in real-life surveillance data," in *Transactions on Data Hiding and Multimedia Security V*, vol. 6010 of *Lecture Notes in Computer Science*, pp. 18–33, Springer, Berlin, Germany, 2010.

[14] M. Hirzer, P. M. Roth, M. Köstinger, and H. Bischof, "Relaxed pairwise learned metric for person re-identification," in *Proceedings of the European Conference on Computer Vision*, pp. 780–793, Springer, Florence, Italy, October 2012.

[15] M. I. Khedher, M. A. El-Yacoubi, and B. Dorizzi, "Probabilistic matching pair selection for SURF-based person re-identification," in *Proceedings of the International Conference of the Biometrics Special Interest Group (BIOSIG '12)*, pp. 1–6, Darmstadt, Germany, September 2012.

[16] T. Matsukawa, T. Okabe, and Y. Sato, "Person re-identification via discriminative accumulation of local features," in *Proceedings of the 22nd International Conference on Pattern Recognition*

(ICPR '14), pp. 3975–3980, IEEE, Stockholm, Sweden, August 2014.

[17] W.-S. Zheng, S. Gong, and T. Xiang, "Reidentification by relative distance comparison," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 3, pp. 653–668, 2013.

[18] H. Y. Chong, S. J. Gortler, and T. Zickler, "A perception-based color space for illumination-invariant image processing," *ACM Transactions on Graphics*, vol. 27, no. 3, article 61, 2008.

[19] K.-J. Yoon and I.-S. Kweon, "Human perception based color image quantization," in *Proceedings of the 17th International Conference on Pattern Recognition (ICPR '04)*, pp. 664–667, August 2004.

[20] L. Shamir, "Human perception-based color segmentation using fuzzy logic," *IPCV*, vol. 2, pp. 96–502, 2006.

[21] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: spatial pyramid matching for recognizing natural scene categories," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '06)*, vol. 2, pp. 2169–2178, New York, NY, USA, June 2006.

[22] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital Signal Processing*, vol. 10, no. 1, pp. 19–41, 2000.

[23] C. E. Rasmussen, "The infinite gaussian mixture model," *NIPS*, vol. 12, pp. 554–560, 1999.

[24] Z. Zivkovic, "Improved adaptive gaussian mixture model for background subtraction," in *Proceedings of the 17th International Conference on Pattern Recognition (ICPR '04)*, vol. 2, pp. 28–31, Cambridge, UK, August 2004.

[25] C. Liu, S. Gong, C. C. Loy, and X. Lin, "Person re-identification: what features are important?" in *Computer Vision—ECCV 2012. Workshops and Demonstrations: Florence, Italy, October 7–13, 2012, Proceedings, Part I*, vol. 7583 of *Lecture Notes in Computer Science*, pp. 391–401, Springer, Berlin, Germany, 2012.

Hindawi Publishing Corporation
Applied Computational Intelligence and Soft Computing
Volume 2016, Article ID 8272796, 10 pages
http://dx.doi.org/10.1155/2016/8272796

Research Article

The Performance of LBP and NSVC Combination Applied to Face Classification

Mohammed Ngadi, Aouatif Amine, Bouchra Nassih, Hanaa Hachimi, and Adnane El-Attar

Systems Engineering Laboratory, National School of Applied Sciences, Ibn Tofail University, Kenitra, Morocco

Correspondence should be addressed to Mohammed Ngadi; ngadi.mohammed@univ-ibntofail.ac.ma

Received 15 October 2016; Accepted 10 November 2016

Academic Editor: Lei Zhang

The growing demand in the field of security led to the development of interesting approaches in face classification. These works are interested since their beginning in extracting the invariant features of the face to build a single model easily identifiable by classification algorithms. Our goal in this article is to develop more efficient practical methods for face detection. We present a new fast and accurate approach based on local binary patterns (LBP) for the extraction of the features that is combined with the new classifier Neighboring Support Vector Classifier (NSVC) for classification. The experimental results on different natural images show that the proposed method can get very good results at a very short detection time. The best precision obtained by LBP-NSVC exceeds 99%.

1. Introduction

Researchers have shown that, to recognize a face, human uses different features such as geometry, texture, and colors of different parts of the face: eyes, mouth, nose, the front, and the cheeks. Based on this observation, several studies have been developed to verify whether it was possible to model this behavior in a computational way.

This article is devoted to the problem of computer based face classification [1], which became a popular and important research topic in recent years thanks to its many applications such as indexing and searching for image and video, security access control, video surveillance. Despite many efforts and progress that have been made during recent years, it remains an open problem and is still considered one of the most difficult problems in the community of computer vision, mainly due to the similarities between the classes and class variations such as occlusion, background clutter, perspective changes, poses scaling, and lighting. Nowadays popular detection approaches are based on descriptors and classifiers, which generally extract visual descriptors in the pictures and videos and then perform the classification using machine learning algorithms based on the extracted features.

Generally, the size of the data can be measured by two dimensions: the number of variables and the number of examples. These two dimensions can take very high values, which can be a problem with the exploration and analysis of these data. In this context, it is essential to implement some data processing tools that allow us to better understand the information contained in our dataset. Dimensionality reduction is one of the oldest approaches that answers this problem. Its objective is to select or retrieve an optimal subset of relevant characteristics according to a previously fixed criterion. This selection/extraction allows reducing the dimension of the space of the examples and making all the data more representative of the problem.

This reduction has a dual purpose, the first is to reduce redundancy, and the second allows facilitating subsequent treatments (feature extraction reduces required storage space and accordingly reduces the classification learning time and accelerates the pattern recognition process) and therefore the data interpretation.

The first step aims to select the best feature extraction [2, 3] method for the context of face classification. In this context, we found that the LBP descriptor gives the most optimal representation of the image. The principle of this

descriptor is to compare each pixel (considered as central pixel in a window of radius R and containing P points) to its neighbors and generate a binary code based on this comparison [4]. For the validation of our choice, we make a comparison with other feature extractions descriptors such as Discrete Wavelet Transform (DWT) [5] and Histogram of Oriented Gradients (HOG) [6].

The second step concerns the selection of the classification function. We have chosen to use a method placed in the context of the semisupervised classifiers, the NSVC. It is based on the combination of two classifiers belonging to two different families (nonsupervised classification: Fuzzy C-Means and supervised one: SVM). The basic idea of the Neighboring Support Vector Classifier (NSVC) is to build new vicinal kernel functions, obtained by supervised clustering in feature space. These vicinal kernel functions are then used for learning.

Finally, our experiments show that LBP-NSVC outperforms all other feature selection and classification algorithms. The main criteria used for comparison are the accuracy of the classification and the execution time, without forgetting the ability of the classifier to effectively manage practical applications where the training data may come from different environments.

The rest of the paper is organized as follows. A brief description of LBP is given in Section 2. Section 3 introduces the NSVC based on supervised partitioning of features space. Experimental results are presented in Section 4, while Section 5 concludes the article.

2. Local Binary Patterns (LBP)

The LBP method can be regarded as a unifying statistical and structural approach to texture analysis. Instead of trying to explain the formation of the texture on the pixel level, local models are formed around each pixel. Each pixel is labeled with the code of the texture that is best at the local level in his neighborhood. Thus, each LBP code can be regarded as the code that best represents the local vicinity of the pixel. The LBP distribution therefore has both structural properties: primitives of textures and the rules for placement of these primitives. For these reasons, the LBP method can be used successfully to recognize a variety of textures, in which structural and statistical methods have been traditionally applied separately.

Local binary patterns were originally proposed by Ojala et al. in 1996 [7]. The concept of the LBP is simple; it proposes assigning a binary code to a pixel based on its neighborhood. This code describing the local texture of a region is calculated by thresholding of a neighborhood with the gray of the central pixel level. In order to generate a binary pattern, all the neighbors will then take a value "1" if their value is greater than or equal to the current pixel and "0" otherwise. This binary pattern's pixels are then multiplied by weights and summoned to obtain a current pixel LBP code. We thus obtain, for any image, pixels with intensity between 0 and 255 as in an ordinary 8-bit image. Rather than describing the image by the sequence of the LBP codes,

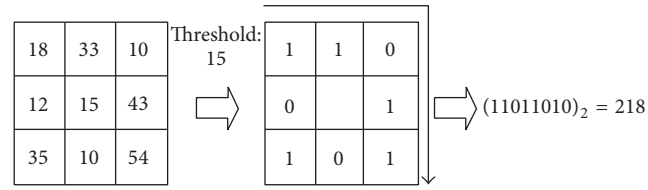


Figure 1: Example of LBP calculation.

one can choose as texture descriptor using a 255-dimension histogram (Figure 1).

The LBP was extended later by using different neighborhood sizes [8–10]. In this case, a circle of radius R around the central pixel is considered. Values of P points sampled on the edge of the circle are taken and compared with the value of the central pixel. To obtain the values for P points sampled in the vicinity for any radius R, an interpolation is necessary. The notation (P, R) is adopted to define the vicinity of P points of radius R of a pixel. LBPP, R is the LBP code for the radius R and the number of neighbors P. The main difference is that the pixels must be interpolated to obtain the values of the points on the circle. The important property of the LBP code is that this code is invariant to uniform illumination changes because the LBP for a pixel depends only on differences between its gray-level and that of its neighbors.

To calculate LBP code in a neighborhood of P pixels in RADIUS R, one simply counts the occurrences of pixels g_i superior or equal to the central value:

$$\mathrm{LBP}_{m,R} = \sum_{i=0}^{m-1} u\left(g_i - g_c\right) \cdot 2^i, \tag{1}$$

where $u(\cdot)$ is the sign function and where g_i and g_c are, respectively, a nearby pixel and the central pixel grayscale

$$u(x) = \begin{cases} 1, & \text{si } x \geq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

The concept of the multiscale LBP is based on the choice of the vicinity in order to calculate LBP code to process textures at different scales [11, 12]. A neighborhood for a central pixel is distributed on a circle and built from two parameters: the number of neighbors "P" on the circle and radius "R" to define a distance between a central pixel and its neighbors (Figure 2). The texture T of an image I is categorized by the combined distribution of gray values of $n + 1$ pixels (where $n > 0$): $T = t(g_c, g_0, \ldots, g_{p-1})$, and g_c corresponds to the value of the central pixel and g_p, with $p = 0, \ldots, P - 1$, corresponds to the level of P pixels regularly spaced on a circle of radius R. If g_c coordinates are equal to $(0, 0)$, then g_p coordinates are given by the following equation:

$$\left(x_p, y_p\right) = \left(x_c + R\cos\left(\frac{2\Pi p}{P}\right), y_c - R\sin\left(\frac{2\Pi p}{P}\right)\right). \tag{3}$$

From the definition of neighborhood, the authors define, first, a local binary pattern that is invariant to any monotonic transformation of grayscale, LBPP, R. For each pixel

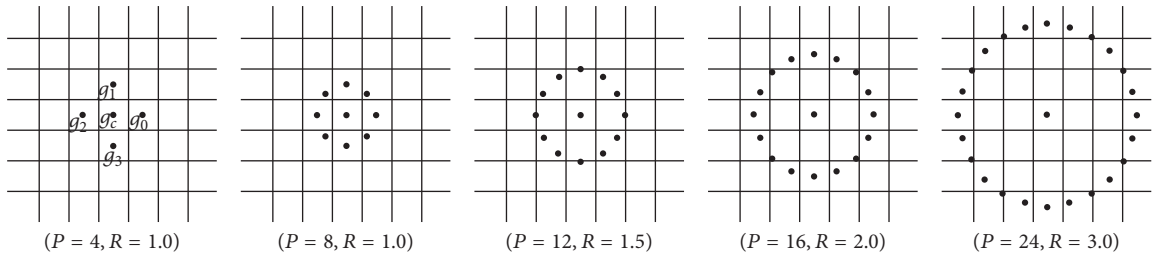


Figure 2: Multiscale LBP. Examples of neighborhoods obtained for various values of (P, R).

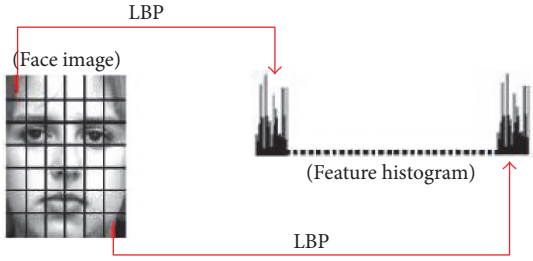


Figure 3: LBP-based facial representation.

(x, y) ($g_c = g(x, y)$), the central pixel is not used for the characterization of textures. Indeed, regardless of g_p vicinity, that pixel only describes a light intensity which is not necessarily useful [11]. Subsequently, g_c is used as a threshold in the following manner:

$$T = t\left(u\left(g_0 - g_c\right), \ldots, u\left(g_0 - g_{p-1}\right)\right). \tag{4}$$

Accordingly, the calculation of the LBP code can be obtained in the same way as the basic LBP (see (1)).

LBP-based face representation: each face image can be considered to be a composition of micropatterns which can be effectively detected by the LBP operator. Hadid et al. [13] introduced LBP-based face representation for facial recognition. To examine the face shape information, they divided the images of face to M small nonoverlapping areas $R0, R1, \ldots, RM$ (as shown in Figure 3).

The NSVC is a classifier adaptive to different datasets. It is based, on one hand, on a nonsupervised approach such as K-means or FCM and, on the other hand, on a supervised approach: SVM.

3. Neighboring Support Vector Classifier (NSVC)

Support Vector Machines, first introduced by Vapnik and colleagues for the problems of classification and regression, can be seen as a new training technique based on traditional polynomial and radial basis function (RBF). As discussed before, SVMs have attracted considerable attention because of their high generalization ability and higher classification performance relative to other pattern recognition algorithms.

However, the assumption that the training data are identically generated from unknown probability distributions may limit the application of SVM to the problems of everyday life [14].

To relax the assumption of identical distribution, the NSVC [15–17] uses a set of vicinal cores functions built based on supervised clustering in the feature space induced by the kernel. The basic idea of the NSVC is to build new vicinal core functions obtained by supervised clustering in the feature space. These vicinal core functions are then used to SVM training.

This approach consists of two steps:

(i) Supervised clustering step based on SKDA algorithm (for supervised kernel-based deterministic annealing, used to partition the training data in different vicinal areas).

(ii) A training step where the SVM technique is used to minimize the vicinal risk function (VRM) under the constraints defined in clustering step based on SKDA.

Consider the following input output data together:

$$\left(x_i, y_i\right)_{i=1}^{l}, \quad x_i \in R^n, \ y_i \in \{-1, 1\}, \tag{5}$$

where l is the number of input data points and n is the dimension of the input space.

The vicinity functions $v(x_i)$ of x_i data points are built if test data points satisfy two assumptions:

(i) The unknown density function is smooth in the neighborhood of each point x_i.

(ii) The function minimizing the functional risk is also smooth and symmetric in the neighborhood of each point x_i.

The optimization problem based on the principle of VRM named vicinal linear SVM [18, 19] can then be formulated as

$$\begin{aligned} \text{minimize:} \quad & \phi(w) = \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \\ \text{subject to:} \quad & y_i \int_{V(x_i)} \left([\langle x, w \rangle + b] \, p\left(x \mid V\left(x_i\right)\right) \right) dx \\ & \geq 1 - \xi_i \\ & \xi_i \geq 0, \ i = 1, \ldots, l, \end{aligned} \tag{6}$$

where w is a weight, C is a punishment constant for ξ_i, b is the offset, $v(x_i)$ is the vicinity associated with the test point x_i, and

$p(x \mid V(x_i))$ is the conditional probability of the respective vicinity in the input space.

The following theorem for the vicinal SVM solution is true (see [18] for a proof):

$$f(x) = \sum_{i=1}^{l} y_i \beta_i L(x, x_i) + b, \tag{7}$$

where to define the coefficients β_i one has to maximize

$$W(\beta) = \sum_{i=1}^{l} \beta_i - \frac{1}{2} \sum_{i,j=1}^{l} \beta_i \beta_j y_i y_j M(x_i, x_j)$$

$$\text{subject to} \quad \sum_{i=1}^{l} \beta_i y_i = 0 \tag{8}$$

$$\beta_i \geq 0,$$

where $L(x, x_i)$ is called the monovicinal kernel and $M(x_i, x_j)$ is the bivicinal kernel of the vicinal SVM [18].

3.1. Supervised Kernel-Based Deterministic Annealing for NSVC. The clustering of training data in the feature space is a well-documented subject [20, 21]. It consists of nonlinearly mapping the observed data of an input low-dimensional space to a high-dimensional feature space using a kernel function, which facilitates the separation of linear data, denoting a nonlinear transformation of the input space X to a high-dimensional space using a kernel function as

$$\Phi : \Re^n \longrightarrow F$$

$$x_i \longrightarrow \Phi(x_i), \tag{9}$$

$$j = 1, \ldots, l,$$

where $\Phi(x_i)$ is the transformed point x_i.

All training data points are distributed in c vicinities/clusters in the feature space, where $\phi_k(z)$ is the center of mass of the kth vicinity residing in F. This is a similar representation to clustering based on the characteristic space of k-means:

$$\phi_k = \sum_{i=1}^{l} \alpha_{ki} z_i, \quad k = 1, 2, \ldots, c, \tag{10}$$

where c is the number of clusters, α_{ki} are the parameters to be defined by the clustering technique (SKDA), and $z_i = y_i \phi(x_i)$ denotes the data points labeled in the feature space.

The classification problem is usually defined mathematically by a cost function to be minimized; for NSVC case, this function is the distortion function. Similar to the notation used in [22], we let $p(\phi_k \mid z_i)$ denote the probability of association of points z_i mapped to the cluster center ϕ_k. Using the square distance $D_k(z_i)$ [15] between the center ϕ_k and the training vector z_i, the distortion function in the function space becomes

$$J_\phi = \sum_{i=1}^{l} \sum_{k=1}^{c} p(z_i)\, p(\phi_k \mid z_i)\, D_k(z_i). \tag{11}$$

Since no a priori knowledge of the distribution of data is assumed, over all possible distributions which give a given value of J_ϕ we choose the one that maximizes the conditional Shannon entropy in the characteristic space:

$$H_\phi = -\sum_{i=1}^{l} \sum_{k=1}^{c} p(z_i)\, p(\phi_k \mid z_i) \log p(\phi_k \mid z_i). \tag{12}$$

The optimization problem can be reformulated as the minimization of the Lagrangian:

$$F_\phi = J_\phi - T H_\phi, \tag{13}$$

where T is the Lagrange multiplier.

To determine α_{ki} parameter, we minimize the free energy function F with respect to the likelihood of association [22], which is related to the Gibbs distribution as

$$p(\phi_k \mid z_i) = \frac{p(\phi_k)\, e^{-(D_k(z_i)/T)}}{\sum_{m=1}^{c} p(\phi_m)\, e^{-(D_m(z_i)/T)}}, \tag{14}$$

where $p(\phi_k)$ is the mass probability for kth cluster:

$$p(\phi_k) = \sum_{i=1}^{l} p(z_i)\, p(\phi_k \mid z_i). \tag{15}$$

And so the energy function is

$$F_\phi^* = \min_{p(\phi_k \mid z_i)} \left(J_\phi - T H_\phi \right)$$

$$= -T \sum_{i=1}^{l} p(z_i) \log \sum_{k=1}^{c} p(\phi_k)\, e^{-(D_k(z_i)/T)}. \tag{16}$$

The partial derivative of F with respect to ϕ_k is

$$\frac{\partial \left(F_\phi^* \right)}{\partial (\phi_k)} = 0. \tag{17}$$

Accordingly

$$\sum_{i=1}^{l} p(z_i)\, p(\phi_k)\, e^{-(D_k(z_i)/T)} \left[z_i - \phi_k \right] = 0. \tag{18}$$

By dividing by the normalization factor

$$Z_{z_i} = \sum_{m=1}^{c} p(\phi_m)\, e^{-(D_m(z_i)/T)}. \tag{19}$$

And, so,

$$\sum_{i=1}^{l} \frac{p(z_i)\, p(\phi_k)\, e^{-(D_k(z_i)/T)}}{Z_{z_i}} z_i$$

$$= \sum_{i=1}^{l} \frac{p(z_i)\, p(\phi_k)\, e^{-(D_k(z_i)/T)}}{Z_{z_i}} \phi_k. \tag{20}$$

Using (14) leads to

$$\sum_{i=1}^{l} p(z_i)\, p(\phi_k \mid z_i)\, z_i = \sum_{i=1}^{l} p(z_i)\, p(\phi_k \mid z_i)\, \phi_k, \tag{21}$$

$$\begin{aligned} \phi_k &= \sum_{i=1}^{l} \frac{p(z_i)\, p(\phi_k \mid z_i)}{\sum_{i=1}^{l} p(z_i)\, p(\phi_k \mid z_i)} z_i \\ &= \sum_{i=1}^{l} \alpha_{ki} z_i. \end{aligned} \tag{22}$$

Finally, we obtain the expression of α_{ki} that will be used to construct the vicinal kernel for NSVC functions:

$$\alpha_{ki} = \frac{p(z_i)\, p(\phi_k \mid z_i)}{\sum_{j=1}^{l} p(z_j)\, p(\phi_k \mid z_j)}. \tag{23}$$

3.2. NSVC with the Feature Space Partitioning. The optimization problem based on feature space partitioning is formulated as follows [18]:

$$\begin{aligned} \text{minimise:} \quad & \phi(w) = \frac{1}{2} w^T w + C \sum_{k=1}^{K} \xi_k \\ \text{subject to:} \quad & y_k \int_{V(\phi_k)} [\langle z, w \rangle + b]\, p(z \mid \phi_k)\, dz \\ & \geq 1 - \xi_k, \\ & i = 1, \ldots, l \\ & \xi_k \geq 0, \ k = 1, \ldots, K, \end{aligned} \tag{24}$$

where $v(\phi_k)$ represents the kth vicinity associated with the mass center ϕ_k in the feature space and $p(z \mid \phi_k)$ is the conditional probability of respective vicinity in the feature space. According to Bayes theorem, we have

$$\begin{aligned} p(z_i \mid \phi_k) &= \frac{p(z_i)\, p(\phi_k \mid z_i)}{p(\phi_k)} \\ &= \frac{p(z_i)\, p(\phi_k \mid z_i)}{\sum_{j=1}^{l} p(z_j)\, p(\phi_k \mid z_j)}. \end{aligned} \tag{25}$$

By comparing (22) and (25), we get

$$\phi_k = \sum_{i=1}^{l} p(z_i \mid \phi_k)\, z_i. \tag{26}$$

And the optimization constraint becomes

$$\begin{aligned} & y_k \int_{V(\phi_k)} [\langle z, w \rangle + b]\, p(z \mid \phi_k)\, dz \\ &= y_k \left[\left\langle \int_{V(\phi_k)} p(z \mid \phi_k)\, z\, dz, w \right\rangle \right. \\ &\quad \left. + \int_{V(\phi_k)} bp(z \mid \phi_k)\, dz \right] \\ &= y_k \left[\left\langle \sum_{i=1}^{l} p(z_i \mid \phi_k)\, z_i, w \right\rangle + \sum_{i=1}^{l} bp(z_i \mid \phi_k) \right] \\ &= y_k [\langle \phi_k, w \rangle + b]. \end{aligned} \tag{27}$$

Let one define the mono- and bivicinal kernels as

$$\begin{aligned} L_k(x) &= \sum_{i=1}^{l} y_i \alpha_{ki} K(x, x_i), \quad k = 1, 2, \ldots, K, \\ M_{km}(x) &= \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_{ki} \alpha_{mj} K(x_i, x_j), \\ & \qquad k, m = 1, 2, \ldots, K, \end{aligned} \tag{28}$$

where α_{ki} parameters are obtained from the SKDA clustering step. The decision boundary is

$$f(x) = \sum_{k=1}^{c} \beta_k y_k L_k(x) + b, \tag{29}$$

where β_k is the coefficient that maximizes the dual function:

$$\begin{aligned} \text{maximize} \quad & W(\beta) \\ & = \sum_{k=1}^{c} \beta_k - \frac{1}{2} \sum_{k,m=1}^{c} \beta_k \beta_m y_k y_m M_{km}(x) \\ \text{subject to} \quad & \sum_{k=1}^{c} \beta_k y_k = 0, \\ & \beta_k \geq 0. \end{aligned} \tag{30}$$

In order to obtain a sparse solution at the cost of the extra clustering procedure, a good selection of the number of clusters is required.

4. Experimental Results

We will now carry out a deep evaluation of the classifiers mentioned in previous sections. We start by a detailed description of the dataset and then present the classification results of all classifiers.

4.1. Dataset. Among the factors that influence or affect the performance of face detection system are scale, pose, lighting conditions, facial expression, and occlusion. For this reason we established a robust database based on diverse illumination conditions and different color and texture variations and, then, under various emotional facial expressions such as neutral expression, anger, scream, sadness, sleepiness, being surprised, wink, frontal smile, frontal smile with teeth, open or closed eyes, and facial details (glasses/no glasses, hats/no hats, and caps/no caps). Then, to use our database in the context of face classification, the different facial images were

Figure 4: Typical people images of database in varied environment.

Table 1: Accuracy of the method of extraction of features with polynomial kernel of NSVC.

	SLBP	ULBP	HOG	DWT
Accuracy %	99.99	99.47	95.73	91.90
Parameter of kernel	2	3	6	3

ULBP: Uniform Local Binary Pattern. SLBP: Simple Local Binary Pattern.

Table 2: Accuracy of the method of extraction of features with polynomial kernel of NSVC.

	SLBP + DWT	SLBP + HOG	ULBP + DWT
Accuracy %	99.53	99.50	99.57
Parameter of kernel	2	3	2

taken at different lighting conditions to make the classification model invariant to illumination. The images were adopted under divers unconstrained environment (Figure 4).

We detect people's faces in our database using the cascade detected of Viola-Jones algorithm and normalized the detected faces with a fixed size of $30 * 30$ pixels. Figure 5 presents some typical face images of database.

4.2. Results of NSVC. The basic idea of Neighboring Support Vector Classifier (NSVC) is to build new neighboring kernel functions, obtained by supervised clustering in feature space. These neighboring kernel functions are then used in SVM based learning.

When using polynomial and RBF kernels, we have used cross-validation in order to compute optimal learning parameters for both kernels.

We evaluate the accuracy of each feature extraction method with NSVC. The results obtained are shown in Tables 1 and 2.

To test the performance of the proposed approach, we compare the precision of the LBP-NSVC algorithm with other combinations such as HOG-NSVC and DWT-NSVC.

Figure 5: Typical face images of database.

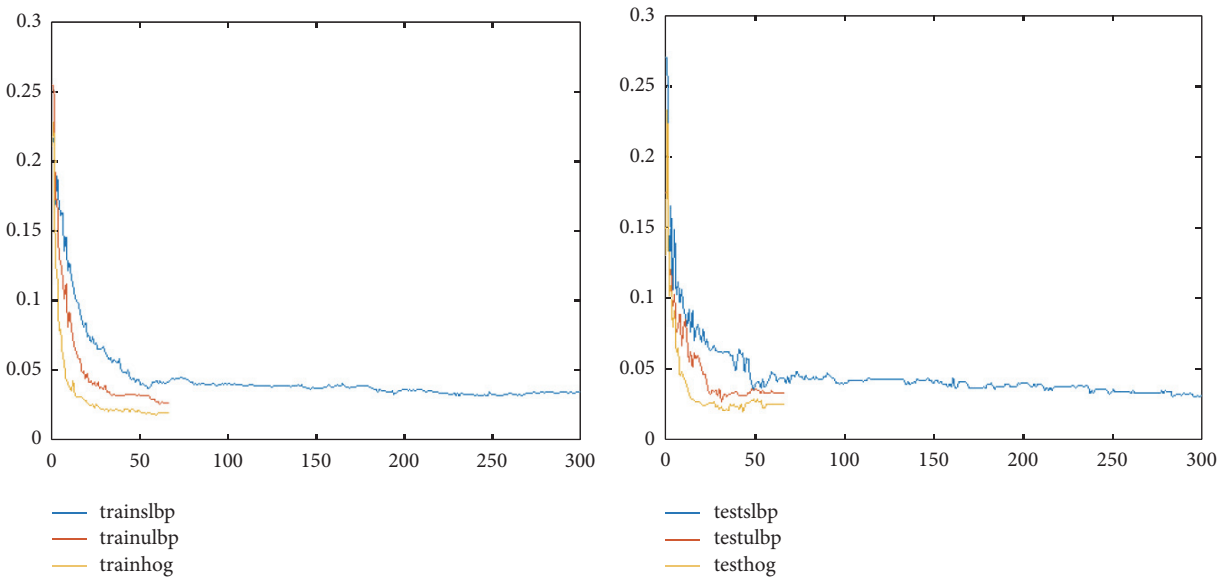


Figure 6: Classification error with respect to the number of weak classifiers.

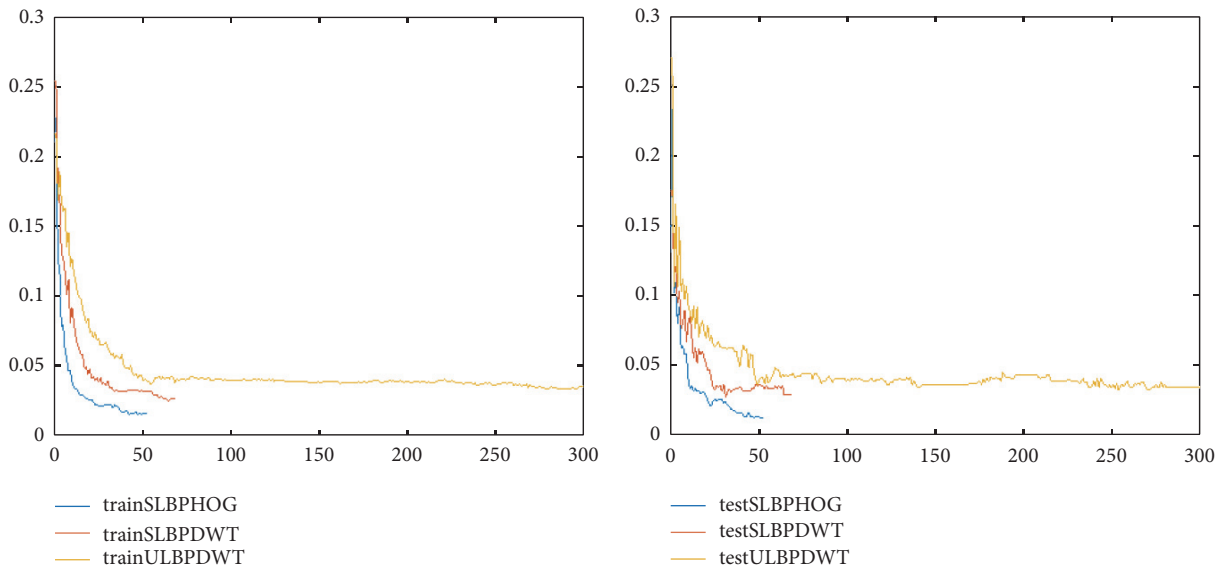


Figure 7: Classification error with respect to the number of weak classifiers.

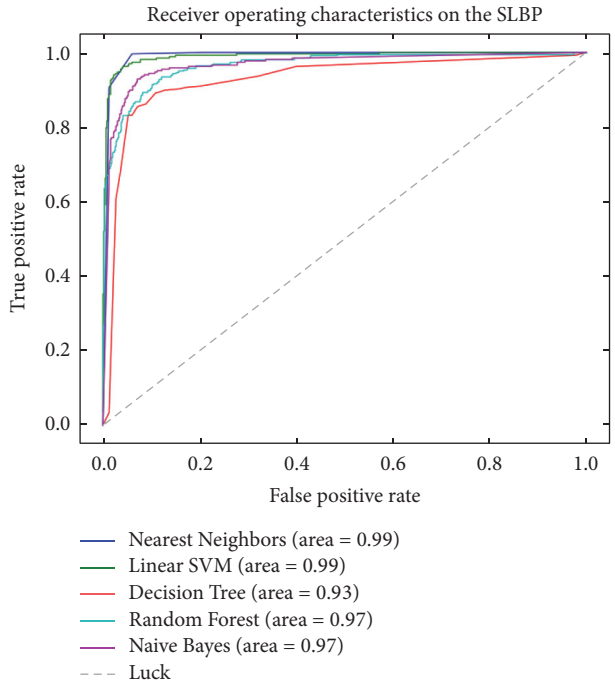


Figure 8: Comparison results of different classifiers methods on SLBP.

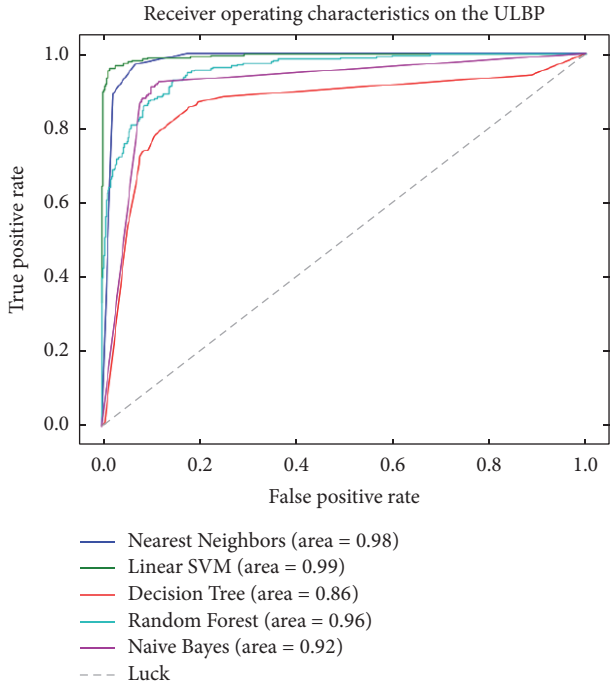


Figure 9: Comparison results of different classifiers methods on ULBP.

So, every time we use LBP-NSVC in our experiments, we must consider polynomial kernel to obtain more accurate results. The following sections show a comparison of the accuracies achieved with our experiences and other classifiers.

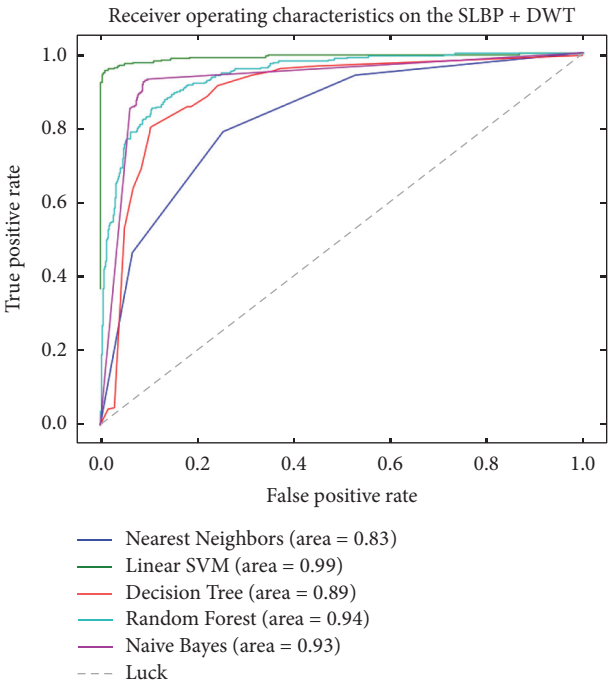


Figure 10: Comparison results of different classifiers methods on SLBP + DWT.

4.3. Results of Adaboost. After classifying our database using Adaboost, the method of boosting is particularly interesting because we can choose the number of classifiers in order to achieve the desired error rates on samples examples. Moreover, we observe that the error rate decreases exponentially with the number of used weak classifiers (Figures 6 and 7).

Figure 6 shows the classification error with respect to the number of weak classifiers for SLBP, ULBP and HOG with Adaboost, and SLBP + DWT, SLBP + HOG, and ULBP + DWT for Figure 7.

4.4. Classifiers Comparison. LBP-NSVC gave the best results for dataset. We seek to demonstrate the performance of this method in comparison with other classification methods.

Classifiers used for our comparison experiments are the following: Naive Bayes, Decision Tree, K Nearest Neighbors (KNN), linear Support Vector Machines (linear SVM), and Random Forest. We compare the classification results of these five algorithms together with our proposed NSVC on the SLBP (Figure 8), ULBP (Figure 9), SLBP + DWT (Figure 10), SLBP + HOG (Figure 11), and ULBP + DWT (Figure 12).

Figures 8–12 show the percentage of classification accuracy of different matching algorithms. It clearly shows that the classification accuracy is best for the majority of algorithms.

It is clear that the approach of the proposed LBP-NSVC produced the best or equal classification accuracy compared to other methods.

In addition to its high performance, the NSVC is a new theoretical method of classification which combines two methods of classification belonging to two different families (unsupervised method: Fuzzy C-Means and supervised method: SVM).

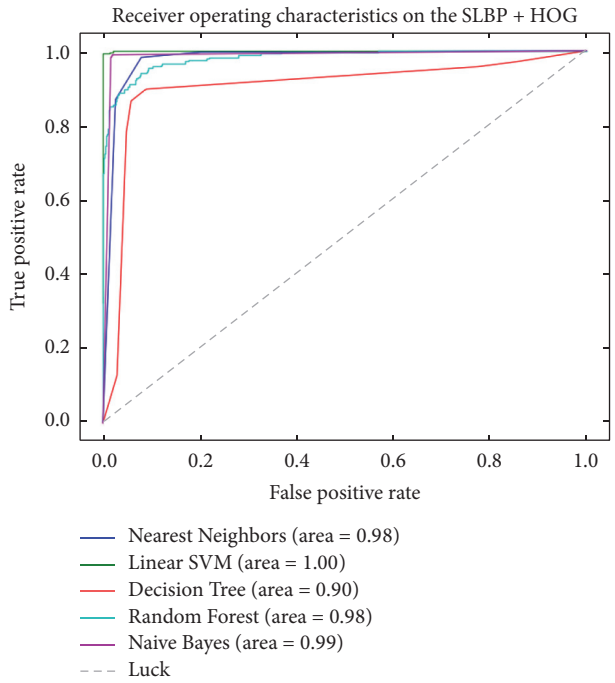


Figure 11: Comparison results of different classifiers methods on SLBP + HOG.

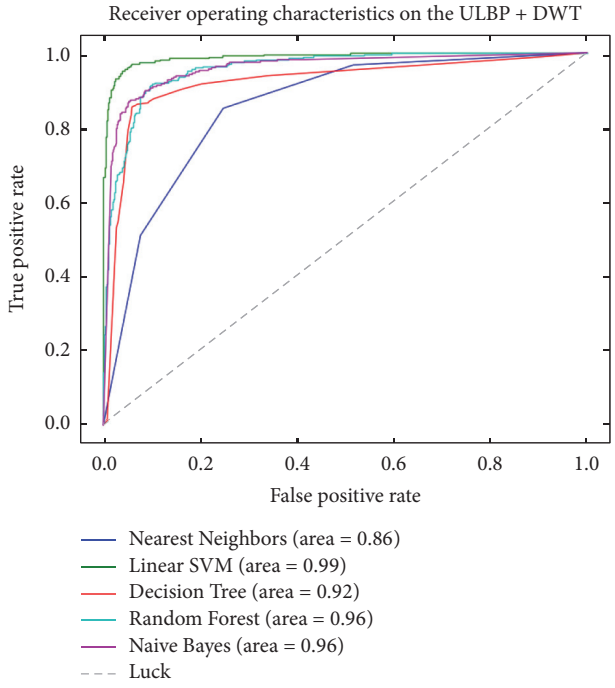


Figure 12: Comparison results of different classifiers methods on ULBP + DWT.

5. Conclusion

We have proposed an original method for face detection. Our system is based on the combination of two types of information: LBP descriptors and descriptors such as DWT and HOG. In order to manage these descriptors and combine them in an optimized way, we propose using an advanced learning system the NSVC. It allows selecting the most important information through kernel weighting depending on their relevance.

The experimental results on different real images show that the proposed method can get very good results.

Our goal in the near future is to continue the study of LBP-NSVC to test it on different datasets from other research areas and try to find the best compromise between precision and execution time.

Competing Interests

The authors declare that they have no competing interests.

References

[1] S. Zafeiriou, C. Zhang, and Z. Zhang, "A survey on face detection in the wild: past, present and future," *Computer Vision and Image Understanding*, vol. 138, pp. 1–24, 2015.

[2] A. Amine, S. Ghouzali, M. Rziza, and D. Aboutajdine, "Investigation of feature dimension reduction based DCT/SVM for face recognition," in *Proceedings of the 13th IEEE Symposium on Computers and Communications (ISCC '08)*, pp. 188–203, July 2008.

[3] A. Majid, A. Khan, and A. M. Mirza, "Gender classification using discrete cosine transformation: a comparison of different classifiers," in *Proceedings of the 7th IEEE International Multi Topic Conference (INMIC '03)*, pp. 59–64, Islamabad, Pakistan, December 2003.

[4] X. Wang, T. X. Han, and S. Yan, "An HOG-LBP human detector with partial occlusion handling," in *Proceedings of the IEEE 12th International Conference on Computer Vision (ICCV '09)*, pp. 32–39, September 2009.

[5] O. Jemai, M. Zaied, C. Ben Amar, and M. A. Alimi, "Fast learning algorithm of wavelet network based on fast wavelet transform," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 25, no. 8, pp. 1297–1319, 2011.

[6] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, pp. 886–893, San Diego, Calif, USA, June 2005.

[7] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on feature distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51–59, 1996.

[8] M. Pietikäinen, A. Hadid, G. Zhao, and T. Ahonen, *Computer Vision Using Local Binary Patterns*, vol. 40, Springer, 2011.

[9] T. Ahonen, A. Hadid, and M. Pietikaäinen, "Face recognition with local binary patterns," in *Computer Vision—ECCV 2004: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11–14, 2004. Proceedings, Part I*, vol. 3021 of *Lecture Notes in Computer Science*, pp. 469–481, Springer, Berlin, Germany, 2004.

[10] A. Gunay and V. V. Nabiyev, "Automatic age classification with LBP," in *Proceedings of the 23rd International Symposium on Computer and Information Sciences (ISCIS '08)*, pp. 1–4, Istanbul, Turkey, October 2008.

[11] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[12] T. Ojala, M. Pietikäinen, and T. Mäenpää, "A generalized local binary pattern operator for multiresolution gray scale and rotation invariant texture classification," in *Advances in Pattern Recognition—ICAPR 2001: Second International Conference Rio de Janeiro, Brazil, March 11–14, 2001 Proceedings*, vol. 2013 of *Lecture Notes in Computer Science*, pp. 399–408, Springer, Berlin, Germany, 2001.

[13] A. Hadid, M. Pietikainen, and T. Ahonen, "A discriminative feature space for detecting and recognizing faces," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '04)*, Washington, DC, USA, June 2004.

[14] K. R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, "An introduction to kernel-based learning algorithms," *IEEE Transactions on Neural Networks*, vol. 12, no. 2, pp. 181–201, 2001.

[15] X. Yang, A. Cao, Q. Song, G. Schaefer, and Y. Su, "Vicinal support vector classifier using supervised kernel-based clustering," *Artificial Intelligence in Medicine*, vol. 60, no. 3, pp. 189–196, 2014.

[16] A. Cao, Q. Song, X. Yang, S. Liu, and C. Guo, "Mammographic mass detection by vicinal support vector machine," in *Proceedings of the IEEE International Joint Conference Neural Networks*, vol. 3, pp. 1953–1958, Budapest, Hungary, July 2004.

[17] M. Ngadi, A. Amine, H. Hachimi, and A. El-Attar, "A new optimal approach for Breast Cancer Diagnosis Classification," *International Journal of Imaging and Robotics*, vol. 16, no. 4, pp. 25–36, 2016.

[18] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, NY, USA, 2nd edition, 2000.

[19] O. Chapelle, J. Weston, L. Bottou, and V. Vapnik, "Vicinal risk minimization," in *Proceedings of the 14th Annual Neural Information Processing Systems Conference (NIPS '00)*, MIT Press, December 2000.

[20] F. Camastra and A. Verri, "A novel Kernel method for clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 801–805, 2005.

[21] J. M. Leski, "Fuzzy c-varieties/elliptotypes clustering in reproducing kernel Hilbert space," *Fuzzy Sets and Systems*, vol. 141, no. 2, pp. 259–280, 2004.

[22] K. Rose, "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2210–2239, 1998.

Hindawi Publishing Corporation
Applied Computational Intelligence and Soft Computing
Volume 2016, Article ID 2783568, 9 pages
http://dx.doi.org/10.1155/2016/2783568

Research Article

Low-Rank Kernel-Based Semisupervised Discriminant Analysis

Baokai Zu,[1,2] Kewen Xia,[1,2] Shuidong Dai,[1,2] and Nelofar Aslam[1,2]

[1]*School of Electronic and Information Engineering, Hebei University of Technology, Tianjin 300401, China*
[2]*Key Lab of Big Data Computation of Hebei Province, Tianjin 300401, China*

Correspondence should be addressed to Kewen Xia; kwxia@hebut.edu.cn

Received 16 April 2016; Accepted 14 June 2016

Academic Editor: Yu Cao

Semisupervised Discriminant Analysis (SDA) aims at dimensionality reduction with both limited labeled data and copious unlabeled data, but it may fail to discover the intrinsic geometry structure when the data manifold is highly nonlinear. The kernel trick is widely used to map the original nonlinearly separable problem to an intrinsically larger dimensionality space where the classes are linearly separable. Inspired by low-rank representation (LLR), we proposed a novel kernel SDA method called low-rank kernel-based SDA (LRKSDA) algorithm where the LRR is used as the kernel representation. Since LRR can capture the global data structures and get the lowest rank representation in a parameter-free way, the low-rank kernel method is extremely effective and robust for kinds of data. Extensive experiments on public databases show that the proposed LRKSDA dimensionality reduction algorithm can achieve better performance than other related kernel SDA methods.

1. Introduction

For many real world data mining and pattern recognition applications, the labeled data are very expensive or difficult to obtain, while the unlabeled data are often copious and available. So how to use both labeled and unlabeled data to improve the performance becomes a significant problem [1, 2]. Recently, semisupervised dimensionality reduction has attracted considerable attention, which can be directly used in the whole database [3]. Illuminated by semisupervised learning (SSL), many methods have been put forward to relieve the so-called small sample size (SSS) problem of LDA [4, 5]. Semisupervised Discriminant Analysis (SDA) first is proposed by Cai et al. [2], which can easily resolve the out-of-sample problem [6] and is more suitable for the real world applications. In SDA algorithm, the labeled samples are used to maximize the different classes' separability and the unlabeled ones to estimate the data's intrinsic geometric information.

Semisupervised Discriminant Analysis may fail to discover the intrinsic geometry structure when the data manifold is highly nonlinear [2, 7]. The kernel trick [8] has been widely used to generalize linear dimensionality reduction algorithms to nonlinear ones, which maps the original nonlinearly separable problem to an intrinsically larger dimensionality space where the classes are linearly separable. So the kernel SDA (KSDA) [2, 7] can discover the underlying subspace more exactly in the feature space, which brings a better subspace for the classification task by a nonlinear learning technique. Cai et al. discussed how to perform SDA in Reproducing Kernel Hilbert Space (RKHS), which gives rise to kernel SDA [2]. You et al. have presented the derivations of a first approach to optimize the parameters of a kernel. It can map the original class distributions to a space where these are optimally (with respect to Bayes) separated with a hyperplane [7]. A new kernel-based nonlinear discriminant analysis algorithm is proposed to solve the fundamental limitations in LDA [9]. A novel KFDA kernel parameters optimization criterion is presented for maximizing the uniformity of class-pair separabilities and class separability in kernel space simultaneously [10]. To overcome the nonlinear dimensionality reduction problems and adopting multiple features restrictions of LFDA, Wang and Sun proposed a new dimensionality reduction algorithm called multiple kernel local Fisher discriminant analysis (MKLFDA) based on the multiple kernel learning [11].

The kernelization of graph embedding applies the kernel trick on the linear graph embedding algorithm to handle data with nonlinear distributions [12]. Weinberger et al. described an algorithm for nonlinear dimensionality reduction based on semidefinite programming and kernel matrix factorization which learns a kernel matrix for high dimensional data that lies on or near a low-dimensional manifold [13].

Low-rank matrix decomposition and completion are recently becoming very popular since Yang et al. and Chen et al. proved that a robust estimation of an underlying subspace which can be obtained by decomposing the observations into a low-rank matrix and a sparse error matrix [14, 15]. Recently, Liu et al. propose a low-rank representation method which is robust to noise and data corruptions due to its ability to decompose noise from the data set [14]. More recently, low-rank representation [16, 17], as a promising method to capture the underlying low-dimensional structures of data, has attracted much attention in the pattern analysis and signal processing communities. LRR method [16–18] seeks the lowest rank representation of all data jointly, such that each data point can be represented as a linear combination of some bases.

The major problem of kernel methods is to find the proper kernel parameters. But all these kernel methods usually use fixed global parameters to determinate the kernel matrix, which are very sensitive to the parameters setting. In fact, the most suitable kernel parameters may vary greatly at different random distribution of the same data. Moreover, the kernel mapping of KSDA always analyze the relationship of the data using the mode one-to-others, which emphasizes local information and lacks global constraints on their solutions. These shortcomings limit the performance and efficiency of KSDA methods. To overcome the disadvantages of the traditional kernel methods, inspired by LRR, we proposed a novel kernel-based Semisupervised Discriminant Analysis called low-rank kernel-based SDA (LRKSDA) where the low-rank representation is used as the kernel method. Compared with other kernels, the low-rank kernel jointly obtains the representation of all the samples under a global low-rank constraint [19]. Thus it is better at capturing the global data structures and very robust to different random distribution of the data set. In addition, we can get the lowest rank representation in a parameter-free way, which is very convenient and robust for kinds of data. Extensive experiments on public databases show that our proposed LRKSDA dimensionality reduction algorithm can achieve better performance than other related methods.

The rest of the paper is organized as follows. We start by a brief review on an overview of SDA in Section 2. We then introduce the low-rank kernel-based SDA framework in Section 3. Then Section 4 reports the experiment results on real world database tasks. In Section 5, we conclude the paper.

2. Overview of SDA

Given a set of samples $[\mathbf{x}_1, \ldots, \mathbf{x}_m, \mathbf{x}_{m+1}, \ldots, \mathbf{x}_{m+l}]$, where $N = m + l$, the first m samples are labeled as $[\mathbf{y}_1, \ldots, \mathbf{y}_m]$, and the remaining l are unlabeled ones. They all belong to c classes. The SDA [2] hopes to find a rejection matrix $\mathbf{a}$, which motivates us to present the prior assumption of consistency by a regularizer term. The objective function is as follows:

$$\mathbf{a} = \max_{\mathbf{a}} \frac{\mathbf{a}^T \mathbf{S}_b \mathbf{a}}{\mathbf{a}^T \mathbf{S}_t \mathbf{a} + \alpha J(\mathbf{a})}, \tag{1}$$

where $\mathbf{S}_b$ and $\mathbf{S}_t$ are the between class scatter and total class scatter matrix. And $\mathbf{S}_w$ is defined as the within class scatter matrix

$$\begin{aligned} \mathbf{S}_w &= \sum_{k=1}^{c} \left(\sum_{i=1}^{l_k} \left(\mathbf{x}_i^{(k)} - \mu^{(k)} \right) \left(\mathbf{x}_i^{(k)} - \mu^{(k)} \right)^T \right), \\ \mathbf{S}_b &= \sum_{k=1}^{c} l_k \left(\mu^{(k)} - \mu \right) \left(\mu^{(k)} - \mu \right)^T, \\ \mathbf{S}_t &= \sum_{i=1}^{l} \left(\mathbf{x}_i - \mu \right) \left(\mathbf{x}_i - \mu \right)^T, \end{aligned} \tag{2}$$

where μ is the mean vector of the total sample, l_k is the number of samples in the kth class, $\mu^{(k)}$ is the average vector of the kth class, and $\mathbf{x}_i^{(k)}$ is the ith sample in the kth class.

The parameter α in (1) balances the model complexity and the empirical loss. The regularizer term supplies us with the flexibility to incorporate the prior knowledge in the applications. We aim at constructing $J(\mathbf{a})$ graph combining the manifold structure through the available unlabeled samples [2]. The key of SSL algorithm is the prior assumption of consistency. For classification, it means that the nearby samples are likely to have same label [20]. And for dimensionality reduction, it implicates that the nearby samples have similar embeddings (low-dimensional representations).

Given a set of samples $\{\mathbf{x}_i\}_{i=1}^m$, we can construct the graph $\mathbf{G}$ to represent the relationship between nearby samples by kNN algorithm. Then put an edge between k nearest neighbors of each other. The corresponding weight matrix $\mathbf{S}$ is defined as follows:

$$\mathbf{S}_{ij} = \begin{cases} 1, & \text{if } \mathbf{x}_i \in N_k\left(\mathbf{x}_j\right) \text{ or } \mathbf{x}_j \in N_k\left(\mathbf{x}_i\right) \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

where $N_k(\mathbf{x}_i)$ denotes the set of k nearest neighbors of $\mathbf{x}_i$. Then $J(\mathbf{a})$ term can be defined as follows:

$$\begin{aligned} J(\mathbf{a}) &= \sum_{ij} \left(\mathbf{a}^T \mathbf{x}_i - \mathbf{a}^T \mathbf{x}_j \right)^2 \mathbf{S}_{ij} \\ &= 2 \sum_i \mathbf{a}^T \mathbf{x}_i \mathbf{D}_{ii} \mathbf{x}_i^T \mathbf{a} - 2 \sum_{ij} \mathbf{a}^T \mathbf{x}_i \mathbf{S}_{ij} \mathbf{x}_j^T \mathbf{a} \\ &= 2\mathbf{a}^T \mathbf{X} (\mathbf{D} - \mathbf{S}) \mathbf{X}^T \mathbf{a} = 2\mathbf{a}^T \mathbf{X} \mathbf{L} \mathbf{X}^T \mathbf{a}, \end{aligned} \tag{4}$$

where $\mathbf{D}$ is a diagonal matrix whose entries are column (or row since $\mathbf{S}$ is symmetric) sum of $\mathbf{S}$; that is, $\mathbf{D}_{ii} = \sum_j \mathbf{S}_{ij}$. The Laplacian matrix [21] is $\mathbf{L} = \mathbf{D} - \mathbf{S}$.

We can get the objective function of the SDA with regularizer term $J(\mathbf{a})$ [2]:

$$\mathbf{a} = \max_{\mathbf{a}} \frac{\mathbf{a}^T \mathbf{S}_b \mathbf{a}}{\mathbf{a}^T \left(\mathbf{S}_t + \alpha \mathbf{X} \mathbf{L} \mathbf{X}^T \right) \mathbf{a}}. \tag{5}$$

By maximizing the generalized eigenvalue problem, we can obtain the projective vector $\mathbf{a}$:

$$\mathbf{S}_b\mathbf{a} = \lambda\left(\mathbf{S}_t + \alpha\mathbf{X}\mathbf{L}\mathbf{X}^T\right)\mathbf{a}. \tag{6}$$

3. Low-Rank Kernel-Based SDA Framework

3.1. Low-Rank Representation. Yan and Wang [22] proposed sparse representation (SR) to construct l_1-graph [23] by solving l_1 optimization problem. However, l_1-graph lacks global constraints, which greatly reduce the performance when the data is grossly corrupted. To solve this drawback, Liu et al. proposed the low-rank representation and used it to construct the affinities of an undirected graph (here called LR-graph) [19]. It jointly obtains the representation of all the samples under a global low-rank constraint, and thus it is better at capturing the global data structures [24].

Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$ be a set of samples; each column is a sample which can be represented by a linear combination of the dictionary $\mathbf{A}$ [19]. Here, we select the samples themselves $\mathbf{X}$ as the dictionary $\mathbf{A}$:

$$\mathbf{X} = \mathbf{A}\mathbf{Z}, \tag{7}$$

where $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n]$ is the coefficient matrix with each $\mathbf{z}_i$ being the representation coefficient of $\mathbf{x}_i$. Different from the SR which may not capture the global structure of the data, LRR seeks the lowest rank solution by solving the following optimization problem [19]:

$$\begin{aligned} \min_{\mathbf{Z}} \quad & \operatorname{rank}(\mathbf{Z}) \\ \text{s.t.} \quad & \mathbf{X} = \mathbf{A}\mathbf{Z}. \end{aligned} \tag{8}$$

The above optimization problem can be relaxed to the following convex optimization [25]:

$$\begin{aligned} \min_{Z} \quad & \|\mathbf{Z}\|_* \\ \text{s.t.} \quad & \mathbf{X} = \mathbf{A}\mathbf{Z}. \end{aligned} \tag{9}$$

Here $\|\cdot\|_*$ denotes the nuclear norm (or trace norm) [26] of a matrix, that is, the sum of the matrix's singular values. By considering the noise or corruption in our real world applications, a more reasonable objective function is

$$\begin{aligned} \min_{\mathbf{Z},\mathbf{E}} \quad & \|\mathbf{Z}\|_* + \lambda\|\mathbf{E}\|_l \\ \text{s.t.} \quad & \mathbf{X} = \mathbf{A}\mathbf{Z} + \mathbf{E}, \end{aligned} \tag{10}$$

where $\|\cdot\|_l$ can be $l_{2,1}$-norm or l_1-norm. In this paper we choose $l_{2,1}$-norm as the error term which is defined as $\|\mathbf{E}\|_{2,1} = \sum_{j=1}^{n}\sqrt{\sum_{i=1}^{n}([\mathbf{E}]_{ij})^2}$. The parameter λ is used to balance the effect of low rank and the error term. The optimal solution $\mathbf{Z}^*$ can be obtained via the inexact augmented Lagrange multipliers method [27, 28].

3.2. Kernel SDA. Semisupervised Discriminant Analysis may fail to discover the intrinsic geometry structure when the data manifold is highly nonlinear. The kernel trick is a popular technique in machine learning which uses a kernel function to map samples to a high dimensional space [8, 29, 30]. By using the kernel trick, we can nonlinearly map the original data to the kernel feature space.

Let $Z, Z : R^m \to F$ be a nonlinear mapping from R^m into F feature space. For any two points $\mathbf{x}_i$ and $\mathbf{x}_j$, we use a kernel function $K(\mathbf{x}_i, \mathbf{x}_j) = \langle\mathbf{\Phi}(\mathbf{x}_i), \mathbf{\Phi}(\mathbf{x}_j)\rangle$ to map the data into a kernel feature space. Some commonly used kernels are including the Gaussian radial basis function (RBF) kernel $K(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|^2/\sigma^2)$, polynomial kernel $K(\mathbf{x}, \mathbf{y}) = (c + \langle\mathbf{x}, \mathbf{y}\rangle)^d$, and sigmoid kernel $K(\mathbf{x}, \mathbf{y}) = \tanh(\langle\mathbf{x}, \mathbf{y}\rangle + \alpha)$ [2, 31].

Let ϕ denote the data matrix in the kernel space: $\mathbf{\Phi} = [\phi, (\mathbf{x}_1)\phi(\mathbf{x}_2), \ldots, \phi(\mathbf{x}_m)]$. The projective vectors $\alpha_1, \alpha_2, \ldots, \alpha_c$ are the eigenvector problem in (6) and then we get $m \times c$ transformation matrix $\mathbf{\Theta} = [\alpha_1, \alpha_2, \ldots, \alpha_c]$. The number of the feature dimensions c can be decided by us. Then a data point can be embedded into c dimensional feature space by

$$\mathbf{x} \longrightarrow \mathbf{y} = \mathbf{\Theta}^T K(:, \mathbf{x}), \tag{11}$$

where $K(:, \mathbf{x}) = [K(\mathbf{x}_1, \mathbf{x}), \ldots, K(\mathbf{x}_m, \mathbf{x})]^T$.

Kernel SDA (KSDA) [2, 7] can discover the underlying subspace more exactly in the feature space. It results in a better subspace for the classification task by a nonlinear learning technique.

3.3. Low-Rank Kernel-Based SDA. The major problem of all these kernel methods is to find the proper kernel parameters. And they usually use fixed global parameters to determinate the kernel matrix, which is very sensitive to the parameters setting. In fact, the most proper kernel parameters may vary greatly at different random distribution even if they are for the same data. Moreover, the traditional kernel mapping always analyzes the relationship of the data using the mode one-to-others, which emphasizes local information and lacks global constraints on their solutions. These shortcomings limit the performance and efficiency of KSDA methods. To overcome these shortcomings mentioned above, inspired by low-rank representation, we propose a novel kernel-based Semisupervised Discriminant Analysis (LRKSDA) where LRR is used as the kernel representation.

Let $Z, Z : R^m \to F$ be a low-rank mapping from R^m into a low-rank kernel feature space F. For the database $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$, a reasonable objective function is as follows:

$$\begin{aligned} \min_{\mathbf{Z},\mathbf{E}} \quad & \|\mathbf{Z}\|_* + \lambda\|\mathbf{E}\|_{2,1} \\ \text{s.t.} \quad & \mathbf{X} = \mathbf{A}\mathbf{Z} + \mathbf{E}. \end{aligned} \tag{12}$$

The optimal solution $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n]$ is the coefficient matrix with each $\mathbf{z}_i$ being the low-rank representation coefficient of $\mathbf{x}_i$.

Let $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n]$ denote the data matrix in the kernel space. The projective vectors $\alpha_1, \alpha_2, \ldots, \alpha_c$ are the eigenvector problem in (6) and $m \times c$ transformation matrix is $\mathbf{\Theta} = [\alpha_1, \alpha_2, \ldots, \alpha_c]$. The number of the feature dimensions

c can be decided by us. Then a data point can be embedded into c dimensional feature space by

$$\mathbf{x} \longrightarrow \mathbf{y} = \mathbf{\Theta}^T \mathbf{z}, \tag{13}$$

where $\mathbf{z}$ is the low-rank representation of $\mathbf{x}$.

Since the low-rank representation jointly obtains the representation of all the samples under a global low-rank constraint to capture the global data structures, we can get the lowest rank representation in a parameter-free way, which is very convenient and robust for kinds of data. So low-rank kernel-based SDA algorithm can improve the performance to a very large extent. The step of the LRKSDA is as follows.

Firstly, map the labeled and unlabeled data to the LR-graph kernel space. Secondly, execute the SDA algorithm for dimensionality reduction. Finally execute the nearest neighbor method for the final classification in the derived low-dimensional feature subspace. The procedure of low-rank kernel-based SDA is described as follows.

Algorithm 1 (low-rank kernel-based SDA algorithm). *Input*. The whole data set $[\mathbf{x}_1, \ldots, \mathbf{x}_m, \mathbf{x}_{m+1}, \ldots, \mathbf{x}_{m+l}]$, where l samples are labeled and m are unlabeled ones.

Output. The classification results.

Step 1. Map the labeled and unlabeled data $\mathbf{X}$ to feature space by the LRR algorithm:

$$\begin{aligned} \min_{\mathbf{Z},\mathbf{E}} \quad & \|\mathbf{Z}\|_* + \lambda \|\mathbf{E}\|_{2,1} \\ \text{s.t.} \quad & \mathbf{X} = \mathbf{AZ} + \mathbf{E}. \end{aligned} \tag{14}$$

Step 2. Implement the SDA algorithm for dimensionality reduction.

Step 3. Execute the nearest neighbor method for final classification.

4. Experiments and Analysis

In this section, we conduct extensive experiments to examine the efficiency of low-rank kernel-based SDA algorithm. The simulation experiment is conducted in MATLAB7.11.0 (R2010b) environment on a computer with AMD Phenom(tn)II P960 1.79 GHz CPU and 2 GB RAM.

4.1. Experiment Overview

4.1.1. Databases. The proposed LRKSDA is tested on six real world databases, including three face databases and three University of California Irvine (UCI) databases. In these experiments, we normalize the sample to a unit norm.

(1) Extended Yale Face Database B [2]. This database has 38 individuals and around 64 near frontal images under different illuminations per individual. Each face image is resized to 32 × 32 pixels. And we select the first 20 persons and choose 20 samples of each subject.

(2) ORL Database [22]. The ORL database contains 10 different images of each for 40 distinct subjects. The images are taken at different times, varying the lighting, facial expressions, and facial details. Each face image is manually cropped and resized to 32 × 32 pixels, with 256 grey levels per pixel.

(3) CMU PIE Face Database [2]. It contains 68 subjects with 41,368 face images. The face images were captured under varying poses, illuminations, and expressions. The size of each image is resized to 32 × 32 pixels. We select the first 20 persons and choose 20 samples for per subject.

(4) Musk (Version 2) Data Set 2. This database contains 2 classes and 6598 instances with 166 features. Here, we randomly select 300 examples for the experiments.

(5) Seeds Data Set. It contains 210 instances for three different wheat varieties. A soft X-ray technique and GRAINS package were used to construct all seven, real-valued attributes.

(6) SPECT Heart Data Set. The database describes diagnosing of cardiac Single Proton Emission Computed Tomography (SPECT) images. Each of the patients is classified into two categories: normal and abnormal. The database of 267 SPECT image sets was processed to extract features that summarize the original SPECT images. The pattern was further processed to obtain 22 binary feature patterns.

4.1.2. Compared Algorithms. In order to demonstrate how the semisupervised dimensionality reduction performance can be improved by low-rank kernel-based SDA, we list out SDA, KSDA1, and KSDA2 algorithm for comparison. In all experiments, the number of the nearest neighbors in the kNN regularizer graph is set to 4.

(1) KSDA1 Algorithm. KSDA1 algorithm is the KSDA with Gaussian radial basis function (RBF) kernel $K(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|^2/\sigma^2)$.

(2) KSDA2 Algorithm. KSDA2 algorithm is the KSDA which uses polynomial kernel $K(\mathbf{x}, \mathbf{y}) = (c + \langle \mathbf{x}, \mathbf{y} \rangle)^d$. Here, $c = 1$.

The classification accuracy is influenced by the kernel parameters. So after comparing, we choose a proper kernel parameters σ and c for the KSDA1 and KSDA2 algorithm in each database in the following pairs, respectively, where $(0.9, 0.9)$ is for Extended Yale Face Database B, $(0.55, 1.5)$ is for ORL database, $(0.9, 0.9)$ is for CMU PIE database, $(0.65, 0.2)$ is for Musk database, $(0.05, 0.6)$ is for Seeds Data Set, and $(0.8, 0.3)$ is for SPECT Heart Data Set, respectively. Since the most suitable kernel parameters vary greatly at different random distribution even if they are for the same data, these kernel parameters are relatively suitable after comparing by many times' runs.

4.2. Experiment 1: Different Algorithms Performances. To examine the effectiveness of the proposed LRKSDA algorithm, we conduct experiments on the six public databases. In our experiments, we randomly select 30% samples from

Table 1: Classification accuracy of different SDA algorithms on six databases.

	Yale B	ORL	PIE	Musk	Seeds	SPECT Heart
LRKSDA	0.825769	0.815	0.578243	0.836667	0.90625	0.778378
KSDA1	0.691392	0.693025	0.541478	0.756849	0.825	0.683333
KSDA2	0.723549	0.681576	0.534542	0.755128	0.709814	0.694154
SDA	0.668397	0.687692	0.527715	0.757407	0.819122	0.69857

Table 2: Classification accuracy of different graphs on ORL, Yale, and USPS databases.

Database	Algorithm	The percentage of labeled samples				
		10%	20%	30%	40%	50%
Yale B	LRKSDA	0.711471	0.792667	0.825769	0.848636	0.877778
Yale B	KSDA1	0.316113	0.536868	0.691392	0.807183	0.856101
Yale B	KSDA2	0.33994	0.569484	0.723549	0.819317	0.877637
Yale B	SDA	0.325919	0.560259	0.668397	0.815348	0.855167
ORL	LRKSDA	0.615556	0.728125	0.815	0.873333	0.9
ORL	KSDA1	0.172412	0.448653	0.693167	0.868578	0.937414
ORL	KSDA2	0.172454	0.454899	0.681576	0.851755	0.930487
ORL	SDA	0.173478	0.442731	0.687692	0.877294	0.948035
PIE	LRKSDA	0.29	0.46	0.578243	0.701044	0.82734
PIE	KSDA1	0.195252	0.371027	0.541478	0.711166	0.827522
PIE	KSDA2	0.195345	0.37646	0.543015	0.712033	0.825519
PIE	SDA	0.195707	0.387806	0.527715	0.725264	0.82658

each class as the labeled samples to evaluate the performance with different numbers of selected features. The evaluations are conducted with 20 independent runs for each algorithm. We average them as the final results. First we utilize different kernel methods to get the kernel mapping, and then we implement the SDA algorithm for dimensionality reduction. Finally, the nearest neighbor approach is employed for the final classification in the derived low-dimensional feature subspace. For each database, the classification accuracy for different algorithms is shown in Figure 1. Table 1 shows the performance comparison of different algorithms. Note that the results are the best results of all these different selected features mentioned above. From these results, we can observe the following.

In most cases, our proposed low-rank kernel-based SDA algorithm consistently achieves the highest classification accuracy compared to the other algorithms. LRKSDA achieves the best performance when the dimensionality is larger than a certain low dimension. And the classification accuracy is much higher than the other kernel SDA algorithms. So it improves the classification performance to a large extent, which suggests that low-rank kernel is more informative and suitable for SDA algorithm.

Since the proper kernel parameters are the most important thing of these traditional algorithms and since the kernel parameters of KSDA1 and KSDA2 algorithm are fixed global parameters, the two algorithms are very sensitive to different data or different random distribution of the same data. The performance improvement of these KSDA methods is not obvious. More seriously, as a result of randomly select labeled samples, the random distribution in each run may not adapt the so-called proper kernel parameters of KSDA1 and KSDA2 algorithm. Moreover, the traditional kernel mapping always analyzes the relationship of the data using the mode one-to-others, which emphasizes local information and lacks global constraints on their solutions. This situation may result in not good performance in some case, while the low-rank representation is better at capturing the global data structures. And we can get the lowest rank representation in a parameter-free way, which is very convenient and robust for kinds of data. So low-rank kernel-based SDA separates the different classes very well compared to other kernel SDA. And it can improve the performance to a very large extent, which means that our proposed low-rank kernel method is extremely effective.

4.3. Experiment 2: Influence of the Label Number. We evaluate the influence of the label number in this part. The experiments are conducted with 20 independent runs for each algorithm. We average them as the final results. The procedure is the same with experiment 1. For each database, we vary the percentage of labeled samples from 10% to 50% and the recognition accuracy is shown in Tables 2 and 3, from which we observe the following.

In most cases, our proposed low-rank kernel-based SDA algorithm consistently achieves the best results, which is robust to the label percentage variations. While some other compared algorithms are not as robust as our LRKSDA algorithm, we can see that the classification accuracy is very awful when the label rate is low. Thus, our proposed method has much superiority than the traditional KSDA and SDA algorithms. Sometimes these traditional methods

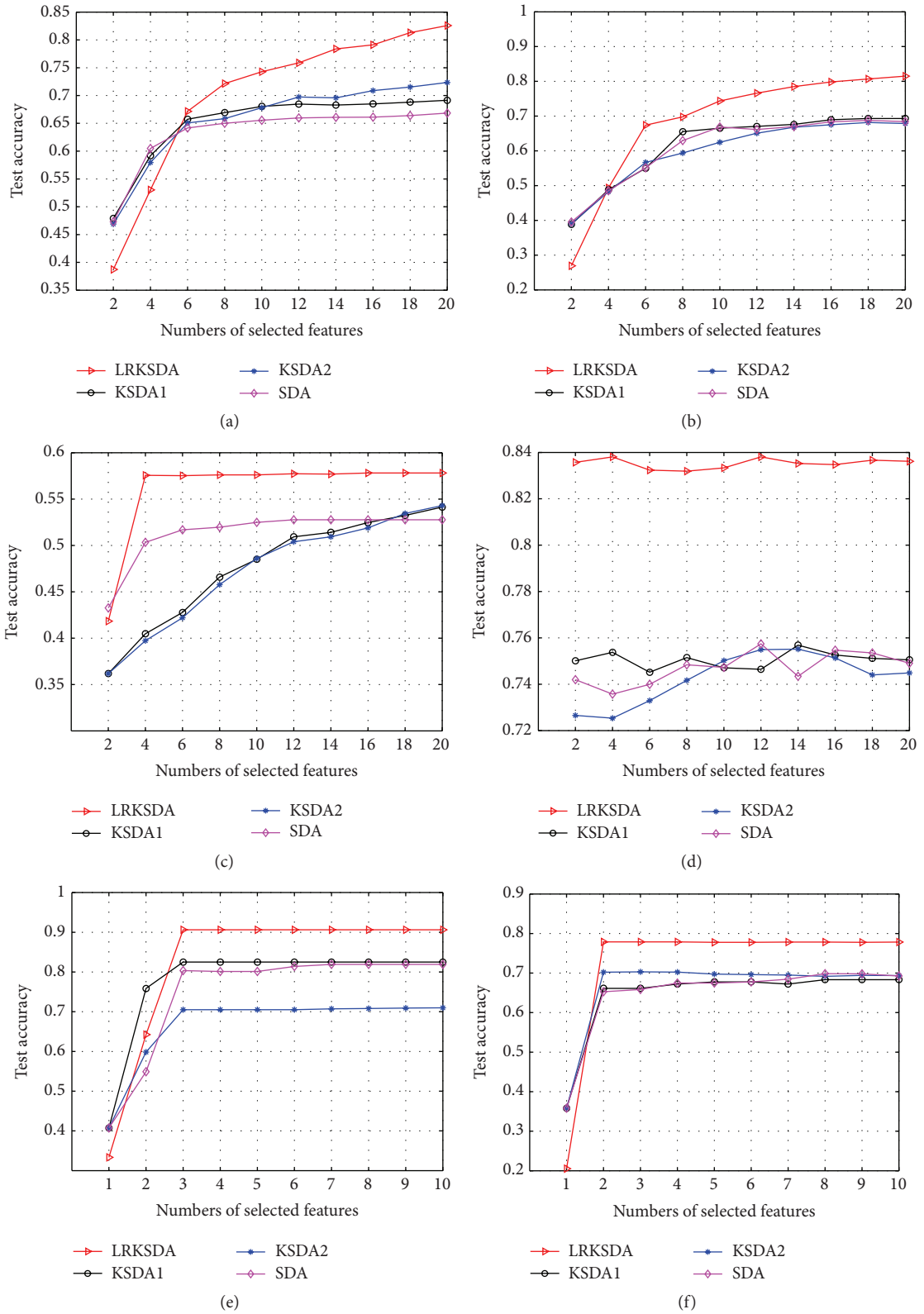


Figure 1: Classification accuracy of different SDA algorithms on the six databases of (a) Extended Yale Face Database B, (b) ORL database, (c) CMU PIE face database, (d) Musk (Version 2) Data Set 2, (e) Seeds Data Set, and (f) SPECT Heart Data Set.

Table 3: Classification accuracy of different graphs on Musk, Seeds, and SPECT Heart databases.

Database	Algorithm	The percentage of labeled samples				
		10%	20%	30%	40%	50%
Musk	LRKSDA	0.767778	0.827083	0.838095	0.838889	0.895125
Musk	KSDA1	0.356299	0.592578	0.756849	0.83253	0.883174
Musk	KSDA2	0.418741	0.607271	0.755128	0.817146	0.888439
Musk	SDA	0.352444	0.611676	0.757407	0.840006	0.894315
Seeds	LRKSDA	0.890323	0.893333	0.90625	0.90813	0.929608
Seeds	KSDA1	0.46676	0.654862	0.825	0.874946	0.914757
Seeds	KSDA2	0.410025	0.609322	0.709814	0.845034	0.890559
Seeds	SDA	0.503879	0.725435	0.819122	0.872932	0.929595
SPECT Heart	LRKSDA	0.778992	0.778378	0.776216	0.78038	0.826076
SPECT Heart	KSDA1	0.404513	0.622916	0.683333	0.786381	0.85786
SPECT Heart	KSDA2	0.398401	0.608538	0.702989	0.77983	0.869786
SPECT Heart	SDA	0.364647	0.556669	0.69857	0.759696	0.818995

Table 4: Classification accuracy of different graphs with varying noise on Yale B database.

Noise types	Algorithm	Variance or density of the three noises					
		0	0.02	0.04	0.06	0.08	0.1
Gaussian	LRKSDA	0.825769	0.816429	0.814286	0.807857	0.808214	0.807143
Gaussian	KSDA1	0.691392	0.555408	0.565422	0.562556	0.574249	0.579816
Gaussian	KSDA2	0.723549	0.585366	0.597015	0.602456	0.590576	0.59866
Gaussian	SDA	0.668397	0.543879	0.540266	0.542199	0.541947	0.543264
"Salt and pepper"	LRKSDA	0.825769	0.794643	0.7675	0.711786	0.643929	0.599286
"Salt and pepper"	KSDA1	0.691392	0.56888	0.509246	0.474557	0.450803	0.436003
"Salt and pepper"	KSDA2	0.723549	0.59498	0.522505	0.478096	0.446308	0.43581
"Salt and pepper"	SDA	0.668397	0.553305	0.498777	0.468533	0.452681	0.429647
Multiplicative	LRKSDA	0.825769	0.825357	0.821429	0.82	0.814286	0.793929
Multiplicative	KSDA1	0.691392	0.631297	0.619849	0.594597	0.584588	0.576168
Multiplicative	KSDA2	0.723549	0.641188	0.622062	0.616446	0.594529	0.594516
Multiplicative	SDA	0.668397	0.594035	0.588897	0.58513	0.582225	0.556328

may achieve good performances in some databases with high enough label rate. But they are not as stable as our proposed algorithm. Since the labeled data is very expensive and difficult, our proposed algorithm is much robust and suitable to the real word data.

As we mentioned in the previous part, since the low-rank kernel method gets the kernel matrix in a parameter-free way, it is robust for different kinds of data, while for the traditional kernel like Gaussian radial basis function kernel and polynomial kernel, if the data's structure does not fit the stable kernel parameters they used, they cannot obtain the good representation of the original data set. Therefore, the low-rank kernel method is much more stable for all the data sets we use. And the low-rank representation jointly obtains the representation of all the samples under a global low-rank constraint, which can capture the global data structures. So it is robust to the label percentage variations even though the label rate is low.

4.4. Experiment 3: Robustness to Different Types Noises. In this test we compare the performance of different algorithms in the noisy environment. Extended Yale Face Database B and Musk database are randomly selected in this experiment. The Gaussian white noise, "salt and pepper" noise, and multiplicative noise are added to the data, respectively. The Gaussian white noise is with mean 0 and different variances from 0 to 0.1. The "salt and pepper" noise is added to the image with different noise densities from 0 to 0.1. And multiplicative noise is added to the data I, using the equation $J = I + n * I$, where I and J are the original and noised data and n is uniformly distributed random noise with mean 0 and varying variance from 0 to 0.1. The number of labeled samples in each class is 30%. The experiments are conducted with 20 runs for each algorithm. We average them as the final results. The procedure is the same with experiment 1. For each graph, we vary the parameter of different noise. The results are shown in Tables 4 and 5.

As we can see, our proposed low-rank kernel-based SDA algorithm always achieves the best results, which means that our method is stable for Gaussian noise, "salt and pepper" noise, and multiplicative noise. And because of the robustness of the low-rank representation to noise, our method LRKSDA is much more robust than other algorithms. With the different kinds of gradually increasing noise, the

Table 5: Classification accuracy of different graphs with varying noise on Musk database.

Noise types	Algorithm	Variance or density of the three noises					
		0	0.02	0.04	0.06	0.08	0.1
Gaussian	LRKSDA	0.838095	0.783333	0.810476	0.795238	0.789524	0.777619
Gaussian	KSDA1	0.756849	0.689112	0.705138	0.702206	0.699312	0.710083
Gaussian	KSDA2	0.755128	0.705054	0.695936	0.697523	0.695125	0.70306
Gaussian	SDA	0.757407	0.713289	0.699202	0.714286	0.676558	0.681785
"Salt and pepper"	LRKSDA	0.838095	0.785238	0.771429	0.772143	0.766429	0.761905
"Salt and pepper"	KSDA1	0.756849	0.683009	0.667079	0.656237	0.66388	0.653854
"Salt and pepper"	KSDA2	0.755128	0.705003	0.664427	0.658723	0.656934	0.652174
"Salt and pepper"	SDA	0.757407	0.70503	0.697131	0.681818	0.678207	0.666734
Multiplicative	LRKSDA	0.838095	0.832381	0.827143	0.809524	0.793333	0.784286
Multiplicative	KSDA1	0.756849	0.733777	0.723228	0.71144	0.716774	0.71115
Multiplicative	KSDA2	0.755128	0.737889	0.716812	0.710216	0.701506	0.68323
Multiplicative	SDA	0.757407	0.749432	0.738486	0.726044	0.703764	0.68799

traditional KSDA and SDA algorithms' performance falls a lot, while our method's performance is robust to these three noises and drops a few.

Notice that the noise is from a different model other than the original data's subspaces. LRR can well solve the low-rank representation problem. When the data corrupted by arbitrary errors, LRR can also approximately recover the original data with theoretical guarantees. In other words, LRR is robust in an efficient way. Therefore, our method is much more robust than other algorithms with the three noises mentioned above.

5. Conclusions

In this paper, we propose a novel low-rank kernel-based SDA (LRKSDA) algorithm, which largely improves the performance of KSDA and SDA. Since low-rank representation is better at capturing the global data structures, LRKSDA algorithm separates the different classes very well compared to other kernel SDA. Therefore, our proposed low-rank kernel method is extremely effective. Empirical studies on six real world databases show that our proposed low-rank kernel-based SDA is much robust and suitable to the real word applications.

Disclosure

Current affiliation for Baokai Zu is Computer Science Department, Worcester Polytechnic Institute, Worcester, MA 01609, USA.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. 51208168), Tianjin Natural Science Foundation (no. 13JCYBJC37700), Hebei Province Natural Science Foundation (no. E2016202341), Hebei Province Natural Science Foundation (no. F2013202254 and no. F2013202102), and Hebei Province Foundation for Returned Scholars (no. C2012003038).

References

[1] D. Zhang, Z. H. Zhou, and S. Chen, *Semi-Supervised Dimensionality Reduction*, SDM, Minneapolis, Minn, USA, 2007.

[2] D. Cai, X. He, and J. Han, "Semi-supervised discriminant analysis," in *Proceedings of the IEEE 11th International Conference on Computer Vision (ICCV '07)*, pp. 1–7, Rio de Janeiro, Brazil, October 2007.

[3] Y. Zhang and D.-Y. Yeung, "Semi-supervised discriminant analysis using robust path-based similarity," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '08)*, pp. 1–8, Anchorage, Alaska, USA, June 2008.

[4] M. Sugiyama, T. Idé, S. Nakajima, and J. Sese, "Semi-supervised local Fisher discriminant analysis for dimensionality reduction," *Machine Learning*, vol. 78, no. 1-2, pp. 35–61, 2010.

[5] Y. Song, F. Nie, C. Zhang, and S. Xiang, "A unified framework for semi-supervised dimensionality reduction," *Pattern Recognition*, vol. 41, no. 9, pp. 2789–2799, 2008.

[6] Y. Bengio, J. F. Paiement, P. Vincent et al., "Out-of-sample extensions for LLE, isomap, MDS, eigenmaps, and spectral clustering," in *Advances in Neural Information Processing Systems 16*, pp. 177–184, MIT Press, 2004.

[7] D. You, O. C. Hamsici, and A. M. Martinez, "Kernel optimization in discriminant analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 631–638, 2011.

[8] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, "An introduction to kernel-based learning algorithms," *IEEE Transactions on Neural Networks*, vol. 12, no. 2, pp. 181–201, 2001.

[9] W.-J. Zeng, X.-L. Li, X.-D. Zhang, and E. Cheng, "Kernel-based nonlinear discriminant analysis using minimum squared errors criterion for multiclass and undersampled problems," *Signal Processing*, vol. 90, no. 8, pp. 2333–2343, 2010.

[10] J. Liu, F. Zhao, and Y. Liu, "Learning kernel parameters for kernel Fisher discriminant analysis," *Pattern Recognition Letters*, vol. 34, no. 9, pp. 1026–1031, 2013.

[11] Z. Wang and X. Sun, "Multiple kernel local Fisher discriminant analysis for face recognition," *Signal Processing*, vol. 93, no. 6, pp. 1496–1509, 2013.

[12] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin, "Graph embedding and extensions: a general framework for dimensionality reduction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 40–51, 2007.

[13] K. Q. Weinberger, B. D. Packer, and L. K. Saul, "Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization," in *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics (AISTATS '05)*, pp. 381–388, January 2005.

[14] S. Yang, X. Wang, M. Wang, Y. Han, and L. Jiao, "Semi-supervised low-rank representation graph for pattern recognition," *IET Image Processing*, vol. 7, no. 2, pp. 131–136, 2013.

[15] C.-F. Chen, C.-P. Wei, and Y.-C. F. Wang, "Low-rank matrix recovery with structural incoherence for robust face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 2618–2625, IEEE, Providence, RI, USA, June 2012.

[16] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 171–184, 2013.

[17] G. Liu and S. Yan, "Latent low-rank representation for subspace segmentation and feature extraction," in *Proceedings of the IEEE International Conference on Computer Vision (IC-CV '11)*, pp. 1615–1622, IEEE, Barcelona, Spain, November 2011.

[18] X. Lu, Y. Wang, and Y. Yuan, "Graph-regularized low-rank representation for destriping of hyperspectral images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 51, no. 7, pp. 4009–4018, 2013.

[19] G. Liu, Z. Lin, and Y. Yu, "Robust subspace segmentation by low-rank representation," in *Proceedings of the 27th International Conference on Machine Learning (ICML '10)*, pp. 663–670, Haifa, Israel, June 2010.

[20] D. Zhou, O. Bousquet, N. T. La et al., "Learning with local and global consistency," *Advances in Neural Information Processing Systems*, vol. 16, no. 16, pp. 321–328, 2004.

[21] D. M. Cvetkovic and P. Rowlinson, "Spectral graph theory," in *Topics in Algebraic Graph Theory*, pp. 88–112, Cambridge University Press, 2004.

[22] S. Yan and H. Wang, "Semi-supervised learning by sparse representation," in *Proceedings of the 2009 SIAM International Conference on Data Mining*, pp. 792–801, SDM, 2009.

[23] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.

[24] C. Cortes and M. Mohri, "On transductive regression," in *Advances in Neural Information Processing Systems 19*, pp. 305–312, 2007.

[25] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *Journal of the ACM*, vol. 58, no. 3, article 11, 2011.

[26] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[27] Z. Lin, M. Chen, and Y. Ma, "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices," https://arxiv.org/abs/1009.5055.

[28] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 171–184, 2013.

[29] S. Yang, Z. Feng, Y. Ren, H. Liu, and L. Jiao, "Semi-supervised classification via kernel low-rank representation graph," *Knowledge-Based Systems*, vol. 69, no. 1, pp. 150–158, 2014.

[30] V. N. Vapnik and V. Vapnik, *Statistical Learning Theory*, John Wiley & Sons, New York, NY, USA, 1998.

[31] H. Nguyen, W. Yang, F. Shen, and C. Sun, "Kernel Low-Rank Representation for face recognition," *Neurocomputing*, vol. 155, pp. 32–42, 2015.