International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol.7, No.4 (2014), pp.303-316
http://dx.doi.org/10.14257/ijsip.2014.7.4.29

Image Classification Based on KPCA and SVM with Randomized Hyper-parameter Optimization

Lin Li[1,2], Jin Lian[2], Yue Wu[1] and Mao Ye[1]

[1]*University of Electronic Science and Technology of China, Chengdu 611731, China*
[2]*Sichuan TOP IT Vocational Institute*
lilin200909@gmail.com

Abstract

Image classification is one of the most fundamental and useful activities in computer vision domain. For better accuracy and executing efficiency under the circumstance of high dimensional feature descriptors in image classification, we proposes a novel framework for multi-class image classification based on kernel principal component analysis(KPCA) for feature descriptors post-processing and support vector machine (SVM) with randomized hyper-parameter optimization for classification. We produce the image feature representation by extracting pyramid histogram of visual word (PHOW) descriptors of image, then map the descriptors though additive kernels. At the third step we use KPCA for feature dimensionality reduction. Finally we classify image by SVM with randomized hyper-parameter optimization. Extensive experiments are tested on two data sets: Msrcv2, 15-Scenes. These experiments justify that (1) feature descriptors with KPCA is superior to that with PCA for dimensionality reduction;(2)SVM with randomized hyper-parameter optimization greatly saves time while keeping high accuracy.

Keywords: *Image Classification, Support Vector Machine, Kernel Principal Component, Hyper-parameter Optimization, Grid search, Randomized Grid Search*

1. Introduction

Nowadays, image classification is the task of classifying image into different classes according to their visual characteristics. It has become one of the fundamental basis for all other image processing activities. It involves two factors: one is feature representation and coding, the other is classifier selection and optimization.

With regard to image representation, Haralick, *et al.,* [1] first propose texture based on gray tone spatial dependencies as image feature representation. This feature is based on statistics which summarize the relative frequency distribution (which describes how often one gray tone will appear in a specified spatial relationship to another gray tone on the image). Afterwards rotation, affine, scale etc. factors being considered, people propose LBP [2], SIFT [3], HOG [4] *etc.*, for feature extraction. These feature extracting algorithms have wide applications in computer vision.

Feature coding is the next step for enhancing feature representation. The typical coding schemes include histogram of texture [5], Bag of Words (BOW) or Bag of Features (BOF) [6]. BOW has been widely used and shows high performance. It has three parts: (1) region selection and representation; (2) cookbook generation and feature quantization; (3)frequency histogram based image representation. In recent years, coding schemes based on pyramid are proposed such as Pyramid Histogram Of visual Word (PHOW) [7], which is based on spatial pyramid matching [8]. The pyramid match computation is linear in the number of features,

ISSN: 2005-4254 IJSIP

and it implicitly finds correspondences based on the finest resolution histogram cell where a matched pair first appears.

Finally, for further incrementing the discriminating capability, Vedaldi, *et al.*, [9] propose additive kernel transformation of feature coding.

However the transformation will produce high dimensional descriptors of image representation. It is necessary to do dimensionality reduction. As one of the classical methods of dimensionality reduction, Principal Component Analysis (PCA) [10] represents the input data as vectors and uses Euclidean distance as the basic metric for nearest neighbor classification. This method seeks the global projection via an optimization formulation assuming that the Gaussian distribution can describe the original data space. However, for the image classification task of kernel transformed descriptors, PCA is not very effective for extracting the discriminant features. Kernel principal component analysis (KPCA) [11] is devoted to the case for nonlinear principal component analysis. KPCA has the advantage that no nonlinear optimization is involved that we only need to solve an eigenvalue problem as in the case of standard PCA. Therefore, we are not in danger of getting trapped in local minima during training.

Nowadays many classifiers have been successfully applied in image classification such as random forests, support vector machine *etc.*, [12]. Haralick, *et al.*, [1] propose piecewise linear discriminant function method and min-max decision rule algorithms with application to photo micrographs, earth resource technology satellite multi-spacial imagery, obtaining the scores of 82% and 83% overall accuracies respectively.

In recent years, image classification by learning algorithms catches a lot of interest in computer vision. Bosch, *et al.*, [7] propose random forests and ferns for image classification. Performance with 30 training images scores 80% on caltech 101 data and 44% on caltech 256. Chapelle, *et al.*, [13] use support vector machine for image classification on data set Corel14, obtaining score of 89% accuracy. Foody, *et al.*, [14] also use SVM for classification on remote sensing imagery data set, they achieve a score of 93.5% accuracy which is superior to the scores of 90.3% for decision tree and 90% for discriminating analysis.

In this paper, in order to improve running performance and the accuracy of image classification, We propose PHOW feature with KPCA dimensionality reduction for feature descriptor and SVM classifier with randomized hyper-parameter optimization. We achieve performances of above 71% accuracy on 15-scenes data set and 49% on Msrcv2 data set with only 10% training samples.

The paper is organized as follows: Section 2 briefly describes the framework proposed for high dimensional image classification with kernel principal component analysis (KPCA) and randomized hyper-parameter optimization. In Section 3 we detail feature extraction with pyramid histogram of visual word (PHOW). In Section 4 we introduce descriptors transformation with additive kernel. Descriptors dimensionality reduce by KPCA are introduced in Section 5. In section 6 we explain how to train and classify by support vector machine (SVM) with randomized hyper-parameter optimization. The experiment results are show in section 7. Finally conclusion and remarks are presented in Section 8.

2. Framework of High Dimensional Image Classification with Kernel PCA and SVM with Hyper-parameter Optimization

We discuss the overall procedure of image classification. We propose a new framework of image classification with high dimensional feature descriptors based on KPCA for feature descriptors dimensional reduction and SVM with hyper-parameter optimization for classification (see Figure 1).

The proposed method is composed of four stages as follows: feature extraction with PHOW, descriptors transformation via additive kernel, descriptors dimensionality reduction by KPCA, model training and classifying by SVM with randomized hyper-parameter optimization.

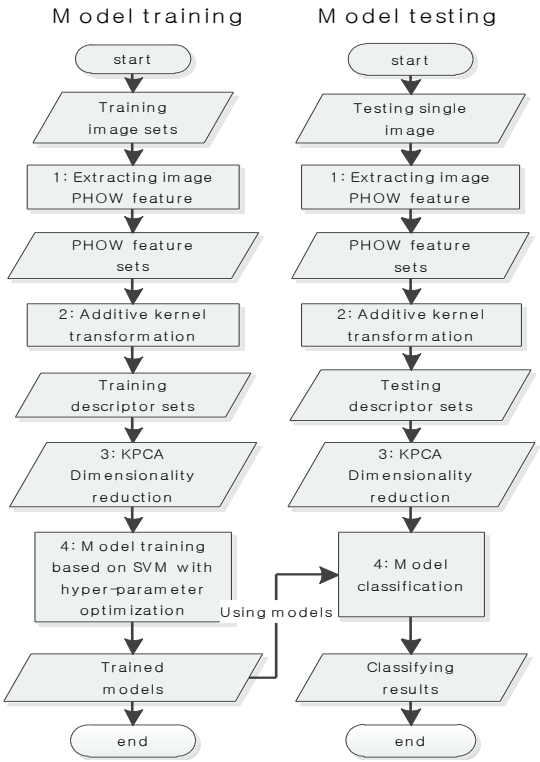

Figure 1. The Outline of our Solution. The Main Framework Consists of Four Processing Steps: Extracting Image PHOW Feature, Additive Kernel Transformation, KPCA Dimensionality Reduction, Model Training based on SVM with Hyper-Parameter Optimization

3. Feature Extraction with PHOW

First of all, we extract image PHOW feature, which is an improvement of dense SIFT descriptors. The basic steps for feature extraction include: (1) We follow the approach of Bosch, *et al.,* [15]. SIFT descriptors [3] are computed at points on a regular grid with spacing M pixels. At each grid point the descriptors are computed over four circular support patches with different radii, consequently each point is represented by four SIFT descriptors. Multiple descriptors are computed to allow for scale variation between images. (2)The dense features are vector quantized into *V visual words* [16] using K-means clustering. (3)Multiple *V visual words* are summed to obtain the total descriptors.

4. Descriptor Transformation with Additive Kernel

At second stage, the PHOW feature descriptor is transformed through additive kernel for good representation. The additive kernel is defined as:

$$K(x, y) = \sum_{b=1}^{B} k(x_b, y_b) \tag{1}$$

where b is number for each bin of histogram, B is count of bins, x_b, y_b is the distribution of each bin, $k : R_0^+ \times R_0^+ \to R_0^+$ is positive definite kernel. We use χ^2 kernel proposed by Vedaldi[9] for the transformation. In our experiment, this will produce 36000 dimensionality feature descriptors at each image.

5. Descriptors Dimensionality Reduction by KPCA

In the third stage of the proposed framework, the dimensionality of feature vectors is reduced with the KPCA algorithm. KPCA is used to generate the unique features set and minimize the computational complexity of our framework. For good understanding of dimensionality reduction, we briefly recall the linear PCA.

5.1. Linear PCA

The main goal of PCA is to reduce an initial large number of variables to a smaller set of uncorrelated ones, called Principal Components (PCs) [17]. The set of PCs is capable of reproducing as much variance from the original variables as possible. PCA solutions with more than one PC are referred to as multi-dimensional solutions. In such cases, the PCs are ordered according to their eigenvalues. The first component is associated with the largest eigenvalue, and accounts for most of the variance, the second accounts for as much as possible of the remaining variance, and so on.

Given a set of ℓ vectors $x_k, k = 1, 2, 3, \ldots \ell, x_k \in R^N$, $\sum_{k=1}^{\ell} x_k = 0$, the covariance matrix defined as

$$C = \frac{1}{\ell} \sum_{k=1}^{\ell} x_k x_k^{\cdot} \tag{2}$$

This can be accomplished by solving the eigenvalue equation

$$\lambda_i v_i = C v_i \tag{3}$$

where $\lambda_i > 0$ are the eigenvalues and $v_i \in R^N$ the eigenvectors of the covariance matrix. The new coordinates in the eigenvector basis, *i.e.*, the orthogonal projections onto the eigenvectors are called principal components.

5.2. Kernel Principal Component Analysis

Kernel Principal Component Analysis (KPCA) generalizes linear transformation to a nonlinear one of the following kinds[11]. Suppose we first map the data nonlinearly into a feature space F by

$$\Phi : R^N \to \mathbb{F}, x \to \phi(x). \tag{4}$$

We can perform PCA in F on arbitrarily large dimensionality by the use of kernel functions known from support vector machines.

The covariance matrix, $C_F \in \mathbb{F}$, will be defined similarly as

$$C_F = \frac{1}{N} \sum_{i=1}^{N} \phi(x_i) \phi(x_i)^T \tag{5}$$

We assume that $\phi(x_i)$ are centered in feature space. We shall come back to this point later on. We define a $N \times N$ kernel matrix

$$K_F(x_i, xj) = (\phi(x_i), \phi(x_j)) \tag{6}$$

which allows us to compute the value of dot product in F without having to carry out the map ϕ. The kernel function has to satisfy the Mercer's theorem to ensure that it is possible to construct a mapping into a space where K_F acts as a dot product. The projection of a new test point, n is given by

$$(v_\phi^l . \phi(n)) = \sum_{i=1}^{N} \alpha_{\phi_i}^l K_F(x_i, n) \tag{7}$$

where α_ϕ^l is defined by the solutions to the eigen value equation $N\lambda_\phi \alpha_\phi = K_F \alpha\phi$.

Finally, it is important to stress that all the arguments shown in this sub-section rely on the assumption that the data are centered in feature space. This is not a direct consequence of using X instead of G.

At this point, we have the tools necessary to compute the centered kernel matrix based on dot products in input space. However, we still need to choose a form for the kernel function $k(x_i, x_j) := K_{F_{ij}}$.

In our work, we choose five kinds of kernels[18].

(1)linear kernel:

$$k(x_i, x_j) = x_i \cdot x_j^{\bullet} \tag{8}$$

where x_i and x_j are vectors in the input space. $\bullet$ is the transpose of matrix x_j.

(2)polynomial kernel:

$$K(x_i, x_j) = (\gamma(x_i \cdot x_j^{\bullet}) + c)^d \tag{9}$$

where X and Y are vectors in the input space. γ is parameter of polynomial kernel, $c \geq 0$ is a constant trading off the influence of higher-order versus lower-order terms in the polynomial. d is the kernel degree. $\bullet$ is the transpose of matrix x_j.

(3)sigmoid kernel:

$$K(x_i, x_j) = \tanh(\gamma(x_i \cdot x_j^{\bullet}) + c) \tag{10}$$

where x_i and x_j are vectors in the input space. c is a constant trading off parameter for sigmoid kernel. $\bullet$ is the transpose of matrix x_j.

(4)radial basis function(rbf) kernel:

$$K(x_i, x_j) = exp(\frac{-\gamma}{\| x_i - x_j \|^2}) \tag{11}$$

where x_i and x_j are vectors in the input space. γ is parameter of rbf kernel. $\|\downarrow\|$ is the dot product.

(5)cosine kernel:

$$K(x_i, x_j) = \frac{x_i \cdot x_j^{\bullet}}{\left\| x_i * x_j \right\|} \tag{12}$$

where x_i and x_j are vectors in the input space. $\|\cdot\|$ is the dot product. $\bullet$ is the transpose of matrix x_j.

6. Model Training and Classifying by SVM with Randomized Hyper-Parameter Optimization

In this section we propose support vector machine combining with randomize hyper-parameter optimization. First of all, we briefly introduction multi-class support vector machine.

6.1. Support Vector Machine

A support vector machine constructs a hyper-plane or set of hyper-planes in a high or infinite dimensional space, which can be used for classification, regression or other tasks. Intuitively, a good separation is achieved by the hyper-plane that has the largest distance to the nearest training data points of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier. Given training vectors $x_i \in R^n, i = 1,...,l$, in k classes, and a vector $y \in R^n$ such that $y_i \in 1,2,...,k$, Support Vector Classifier(SVC)[19] solves the following primal problem:

$$\phi(w,\xi) = \frac{1}{2}\sum_{m=1}^{k} w_m^T w_m + C\sum_{i=1}^{\ell}\sum_{m \neq y_i} \xi_i^m \tag{13}$$

Where w is unit vector: $\|w\| = 1$, C is trade off coefficient, ξ is slack variable. Equation 13 is subject to

$$\begin{aligned} &(w_{y_i} \cdot x_i) + b_{y_i} \geq (w_m^T \cdot x_i) + b_m + 2 - \xi_i^m \\ &\xi_i^m \geq 0, i = 1,...,\ell. m \in 1,...,k\ , \quad y_i. \end{aligned} \tag{14}$$

The decision function is

$$f(x) = \arg\max[w_n \cdot x] + b_n, n = 1,..,k \tag{15}$$

We can find the solution to this optimization problem in dual variables through the Lagrangian function

$$\begin{aligned} L(w,b,\xi,\alpha,\beta) &= \frac{1}{2}\sum_{m=1}^{k}(w_m^T \cdot w_m) + C\sum_{i=1}^{\ell}\sum_{m=1}^{k}\xi_i^m \\ &-\sum_{i=1}^{\ell}\sum_{m=1}^{k}\alpha_i^m[((w_{y_i} - w_m)\cdot x_i) + b_{y_i} - b_m - 2 + \xi_i^m] - \sum_{i=1}^{\ell}\sum_{m=1}^{k}\beta_i^m\xi_i^m \end{aligned} \tag{16}$$

Here the dummy variables are subject to

$$\alpha_i^{y_i} = 0, \xi_i^{y_i} = 2, \beta_i^{y_i} = 0, i = 1,...,\ell \tag{17}$$

and constraints are

$$\alpha_i^m \geq 0, \beta_i^m \geq 0, \xi_i^m \geq 0, i = 1,...,\ell\ m \in 1,...,k\ , \quad y_i \tag{18}$$

In our experiment, SVM models derived from libsvm[20]. The hyper-parameter include: C as regularization parameter for the error term, d as degree of the polynomial kernel function(ploy), α, γ as kernel coefficients for 'rbf', 'poly' and 'sigm', *tol* as tolerance for stopping criterion and *kernels* are linear, rbf, poly, sigmoid.

6.2. Hyper-parameter optimization

In machine learning areas, hyper-parameter optimization is the problem of choosing a set of hyper-parameter for a learning algorithm, usually with the goal of obtaining good generalization[21]. For training samples (x, y), hyper-parameter optimization problem($\lambda^{(*)}$) can express in terms of a hyper-parameter response function Ψ.

$$\begin{aligned}\lambda^{(*)} &= \arg\min_{\lambda\in\Lambda} \mathrm{E}_{y\sim g_x}[L(y;\mathrm{A}_\lambda(x^{train}))] \\ &\approx \arg\min_{\lambda\in\Lambda} \underset{x\in x^\lambda}{mean}\ \mathrm{L}(y;\mathrm{A}_\lambda(x^{train})) \equiv \arg\min_{\lambda\in\Lambda}\Psi(\lambda) \\ &\approx \arg\min_{\lambda\in\lambda^1,\ldots,\lambda^S}\Psi(\lambda) \equiv \hat{\lambda}\end{aligned} \tag{19}$$

Where λ is the hyper-parameter, L is loss function, A_λ is the learning algorithm with hyper-parameter λ, $\mathrm{E}_{x\sim g_x}$ is generalization error, S is the amount of λ. Λ is the indexed by various configuration variables.

6.2.1. Grid search optimization

The most widely used strategies for hyper-parameter optimization is grid search, which is simply an exhaustive searching through a manually specified subset of the hyper-parameter space of a learning algorithm. Grid search experiments are common in the literature of empirical machine learning, where they are used to optimize the hyper-parameter of learning algorithms. It is also common to perform multistage, multi-resolution grid experiments that are more or less automated.

In our experiment, the hyper-parameter space includes five parameters: C is set at 1e3, 5e3, 1e4, 5e4, 1e5 respectively. d is set at 1, 2, 3, 4, 5 respectively. γ is kernel coefficient for 'rbf', 'poly' and 'sigm'. We set it at 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.1 respectively. *tol* is set at 1e-2, 1e-3, 1e-4, 1e-5, 1e-6 respectively. kernel is set to linear, rbf, poly, sigmoid respectively.

6.2.2. Randomized Search Optimization

A major drawback of manual search is the difficulty in reproducing results. This is important both for the progress of scientific research in machine learning as well as for ease of application of learning algorithms by non-expert users. On the other hand, grid search alone does very poorly in practice. This paper shows empirically and theoretically that randomly chosen trials are more efficient for hyper-parameter optimization than trials on a grid in image classification.

In our experiments, randomized search optimization is done by a randomized search over parameters, where each setting is sampled from a distribution over possible parameter values [22]. This has two main advantages over a grid search optimization:

(1)A budget can be chosen independent of the number of parameters and possible values.

(2)Adding parameters that do not influence the performance does not decrease efficiency.

Specifying how parameters should be sampled is done using a dictionary, very similar to specifying parameters for grid search optimization. And furthermore, a computation cost, being the number of sampled candidates or sampling iterations, is specified using the loop parameter. For each parameter, either a distribution over

possible values or a list of discrete choices (which will be sampled uniformly) can be specified.

7. Experiments

In our experiment, we use two data sets. The Microsoft Research Cambridge Object Recognition Image Database version 2 (Msrcv2) data set has 591 images, 23 object classes. The original 15-scenes [23] data set consists of 15 classes and totally 4485 images.

For the two data sets, we randomly selected 10%, 20%, 30%, 40%, 50% samples as training samples, the rest as test ones. Following the procedure of Figure 1, we extracted the image feature descriptors and did KPCA transformation with 100 principal components. Finally we got $N \times 100$ for training features. Here N is the counts of training samples. The test samples were also done by the same procedure.

In the experiments, the feature extraction and transformation is done in matlab environment with VLF eat [9]. The classification tasks are carried out in python environment with scikit-learn[22], numpy, scipy and etc. The testing environment is on a desktop with CPU Intel Core2 2.53 GHZ and 2 GB RAM.

7.1. Experiments and Discussions

7.1.1. Feature Transformation

In this experiment, we compare the training time of PCA and KPCA on data sets: Msrcv2 and 15-Scenes. The kernel type of KPCA is respectively linear, poly, rbf, sigmoid and cosine. The training sampling percentages are set at 10%, 20%, 30%, 40% and 50% respectively. Then the experimental results are illustrated in Figure 2 and Figure 3. Where y-axis denotes the training time (sec), x-axis denotes the training sampling percentages(%).

As observed from Figure 2 and Figure 3, we find the following properties:

(1)The executing performance of KPCA is greatly superior to that of PCA in image feature descriptor dimensionality reduction.

(2)The training time increases with increase of training sampling percentage.

(3)KPCA with different kernels almost have same training runtime performances.

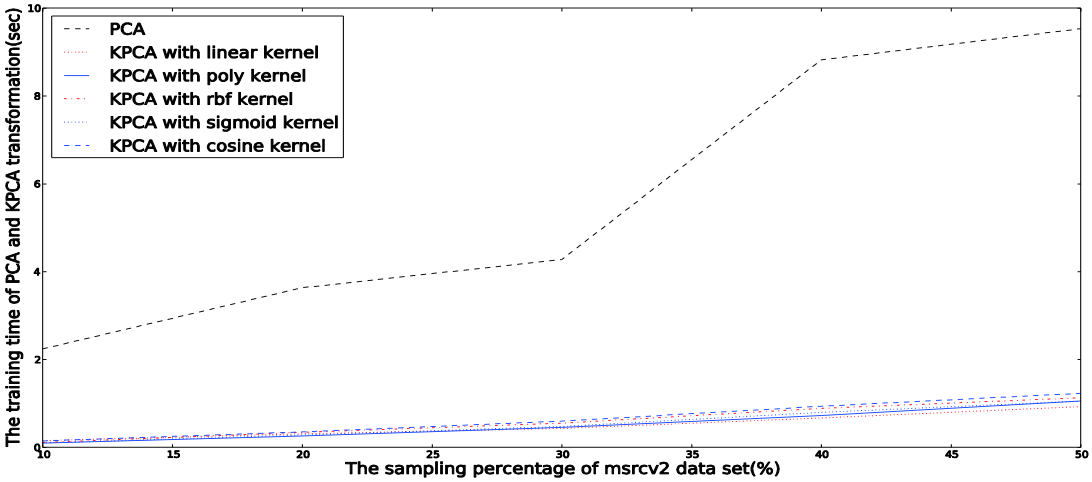

Figure 2. The PCA and KPCA Transformation Training Time on Msrcv2

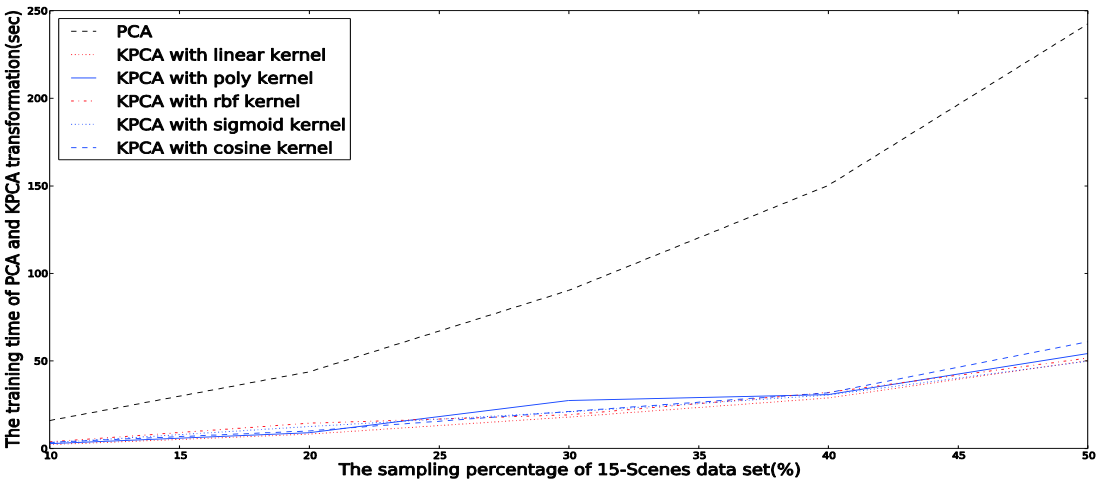

Figure 3. The PCA and KPCA Transformation Training Time on 15-Scenes

7.1.2. The PCA and KPCA Transformation Training Time on 15-Scenes

In this experiment, we compare the accuracy of PCA and KPCA by various principal components on data sets: Msrcv2 and 15-Scenes with 10% sampling for training. The different principal components are set at 10, 20, ..., 200 with interval of 10. Here we choose SVM with randomize hyper-parameter optimization. The experiment results show in Figure 4 and Figure 5 (y-axis denotes the total accuracy and x-axis denotes the principal components extracted from 10 to 200 with interval of 10).

From Figure 4 and Figure 5, we obtain the following results:

(1)On data set 15-Scences PCA obtains the lowest accuracy at the point with 50 principal components. KPCA with sigmoid kernel oscillates up and down at some PCs on both data sets.

(2)On data set Msrcv2, PCA and KPCA almost have same accuracy.

(3)The majority of classifier performances reach stable accuracy at point with 100 PCs.

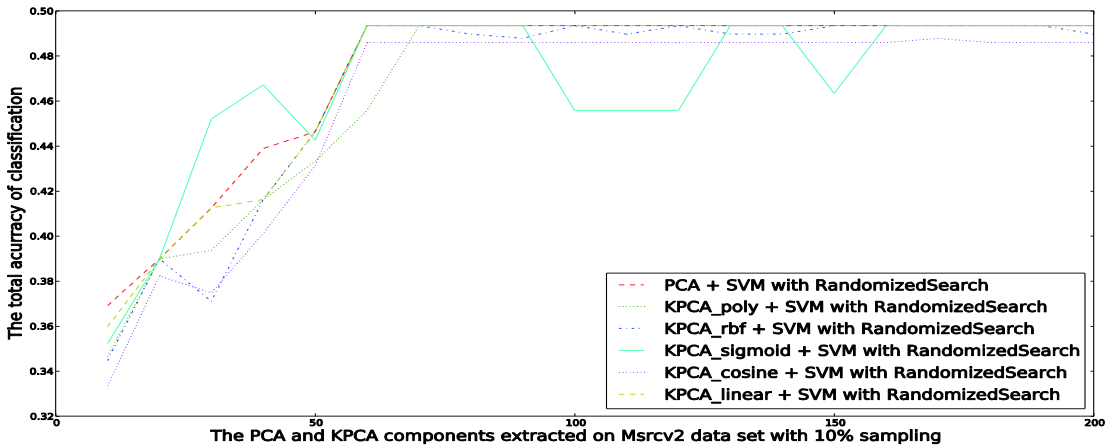

Figure 4. The Results by Various Principal Components on Msrcv2 with 10% Sampling

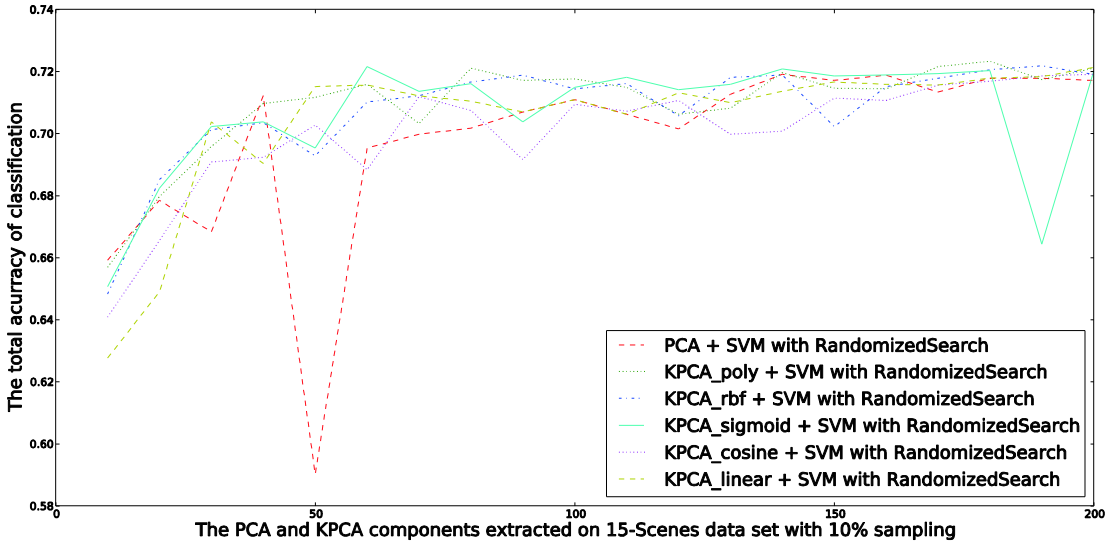

Figure 5. The Results by Various Principal Components on 15-Scenes with 10% Sampling

7.1.2. Image Classification with Hyper-Parameter Optimization by Various Sampling Percentages

In this experiment, we compare the accuracy and executing performance of PCA, KPCA by various kernel on data sets: Msrcv2 and 15-Scenes with different sampling for training. The kernel type of KPCA is set at linear, poly, rbf, sigmoid and cosine respectively. The SVM hyper-parameter optimization is grid search and randomized search. The experimental results are illustrated in table 1, table 2. The first column in tables denotes the various methods used for image classification. The method of PCA + GS stands for feature descriptors dimensionality reduction by PCA and SVM classification by hyper-parameters optimization with grid search. Likewise, PCA + RS stands for feature descriptors dimensionality reduction by PCA and SVM classifier by hyper-parameters optimization with randomized search. KPCA + poly + GS is feature descriptors dimensionality reduction by KPCA with poly kernel and SVM classifier by hyper-parameters optimization with grid search. KPCA + poly + RS is feature descriptors dimensionality reduction by KPCA with poly kernel and SVM classifier by hyper-parameters optimization with randomized search. The rest methods have similar meaning. Column 2 to column 6 are the sampling percentages for training.

Table 1. The Comparison of Accuracy and Running Time of Two Algorithms on Msrcv2 with Various Sampling Percentages

accuracy(%) /time(sec) Method	10%	20%	30%	40%	50%
PCA + GS	49.34/1.7	51.69/3.5	57.14/5.7	59.32/8.0	62.71/11.1
PCA + RS	49.34/0.2	53.81/0.2	57.14/0.4	59.32/0.7	62.37/0.9

KPCA + poly + GS	45.57/1.4	53.60/2.8	57.14/4.6	60.73/7.0	62.03/9.9
KPCA + poly + RS	49.34/0.1	53.60/0.2	57.14/0.4	60.73/0.6	62.03/0.9
KPCA + rbf + GS	45.57/1.4	53.81/2.8	57.14/4.7	60.45/6.9	63.72/9.8
KPCA + rbf + RS	49.34/0.1	53.81/0.2	57.14/0.4	60.45/0.6	63.72/0.8
KPCA + sigmoid + GS	45.57/1.4	53.38/2.8	57.14/4.6	62.99/6.8	63.05/9.7
KPCA + sigmoid + RS	45.57/0.1	53.38/0.2	57.86/0.4	62.99/0.6	63.05/0.8
KPCA + cosine + GS	48.58/1.5	53.38/3.0	57.62/4.9	59.32/7.5	61.69/10.3
KPCA + cosine + RS	48.58/0.1	53.38/0.2	57.62/0.4	58.47/0.6	62.03/0.9
KPCA + linear + GS	49.34/1.5	51.69/3.1	57.14/5.1	59.32/7.2	62.71/10.2
KPCA + linear + RS	49.34/0.1	53.81/0.2	57.14/0.4	60.16/0.6	62.03/0.9

Table 2. The Comparison of Accuracy and Running Time of Two Algorithms on 15-Scenes with Various Sampling Percentages

accuracy(%) /time(sec) Method	10%	20%	30%	40%	50%
PCA + GS	71.53/17.0	76.47/49.1	78.52/100.9	80.25/162.4	80.53/228.4
PCA + RS	71.08/1.5	71.08/1.4	76.08/3.9	78.52/9.0	80.25/17.4
KPCA + poly + GS	71.75/15.7	75.96/50.0	77.15/104.3	77.98/176.1	79.28/257.5
KPCA + poly + RS	71.75/1.3	74.90/4.1	77.21/9.3	77.98/14.4	78.88/24.6
KPCA + rbf + GS	71.43/15.6	75.18/50.7	77.08/104.8	77.98/177.7	78.61/266.2
KPCA + rbf + RS	71.55/1.4	73.96/4.7	77.34/7.5	77.98/14.3	78.61/22.3
KPCA + sigmoid + GS	71.48/15.9	75.18/52.9	77.02/112.1	77.98/193.7	78.57/286.0
KPCA + sigmoid + RS	71.48/1.4	75.18/4.7	75.68/10.4	78.05/15.9	78.57/23.2
KPCA + cosine + GS	70.81/15.4	74.74/43.5	77.34/84.9	77.87/138.2	79.55/201.2
KPCA + cosine + RS	69.67/1.3	75.49/3.5	76.86/7.6	77.87/15.5	77.36/16.6
KPCA + linear + GS	71.53/15.3	76.47/43.8	78.52/83.6	80.25/138.0	80.53/197.3
KPCA + linear + RS	70.78/1.3	76.08/3.7	78.52/6.7	80.25/13.1	80.53/17.9

From Table 1 and Figure 4, we obtain the following results on data set msrcv2:

(1)The training time for SVM with hyper-parameter by randomized search is great superior to that for SVM with hyper-parameter by grid search.

(2)SVM classifiers with grid search have same running performance for training at same sampling percentage.

(3)SVM classifiers with randomized search have same running performance for training at each sampling percentage.

(4)The training time increased with the increase of sampling percentage.

(5)The total accuracies of various methods are almost the same at each sampling percentage.

8. Conclusions

The problems of high dimensional feature descriptors dimensionality reduction and with hyper-parameter optimization for classifier are vital important in image classification. The two factors directly affect the accuracy and running efficiency in real application.

In this paper, we propose an effective image classification based on feature descriptors dimensionality reduction by KPCA and SVM classifier with hyper-parameter optimization by randomized search. Experimental results illustrate that the proposed solution achieves better performance than feature descriptors dimensionality reduction by PCA and SVM classifier with hyper-parameter optimization by grid search.

At feature dimensionality reduction stage, KPCA apparently has advantages over PCA for feature descriptors dimensionality reduction. The total accuracies reach stable status at 100 principal components extracted by PCA and KPCA.

At the classification step, the running time of hyper-parameter with randomized search is about ten times as fast as that of hyper-parameter with grid search while keeping the total accuracy.

Acknowledgements

This work was supported in part by 973 National Basic Research Program of China (2010CB732501) and Fundamental Research Funds for the Central University.

References

[1] R. M. Haralick, K. Shanmugam and I. H. Dinstein, "Textural features for image classification", IEEE Transactions on Systems, Man and Cybernetics. SMC, vol. 3, no. 6, **(1973).**

[2] T. Ojala, M. Pietikinen and D. Harwood, "A comparative study of texture measures with classification based on featured distributions", Pattern Recognition, vol. 29, no. 1, **(1996).**

[3] D. G. Lowe, "Distinctive image features from scale-invariant keypoints", International journal of computer vision, vol. 60, no. 2, **(2004).**

[4] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection", Proceedings of IEEE 12th International Conference on Computer Vision, **(2005)** June 20-26, San Diego, USA.

[5] T. Leung and J. Malik, "Representing and recognizing the visual appearance of materials using three-dimensional textons", International journal of computer vision, vol. 43, no. 1, **(2001).**

[6] L. Nanni and A. Lumini, "Heterogeneous bag of features for object/scene recognition", Applied Soft Computing, vol. 13, no. 4, **(2013).**

[7] A. Bosch, A. Zisserman and X. Muoz, "Image classification using random forests and ferns", Proceedings of IEEE 11th International Conference on Computer Vision, **(2007)** October 14-21, Rio de Janeiro, Brazil.

[8] K. Grauman and T. Darrell, "The pyramid match kernel: Discriminative classification with sets of image features", Proceedings of IEEE 12th International Conference on Computer Vision, **(2005)** June 20-26, San Diego, USA.

[9] A. Vedaldi and A. Zisserman, "Auto collage, Efficient additive kernels via explicit feature maps. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, no. 3, **(2012)**.
[10] M. A. Turk and A. P. Pentland, "Face recognition using eigen faces", Proceedings of 1991 IEEE Conference on Computer Vision and Pattern Recognition, **(1991)** June 3-6, Maui, HI, USA.
[11] B. Scholkopf, A. Smola and K.-R. Mller, "Advances in kernel methods-support vector learning, MIT Press, Cambridge, Massachusetts, **(1999).**
[12] P. Kamavisdar, S. Saluja and S. Agrawal, "A survey on image classification approaches and techniques", International Journal of Advanced Research in Computer and Communication Engineering, vol. 28, no. 5 **(2007).**
[13] O. Chapelle, P. Haffner and V. N. Vapnik, "Support vector machines for histogram based image classification", IEEE Transactions on Neural Networks, vol. 10, no. 5, **(1999).**
[14] G. M. Foody and A. Mathur, "A relative evaluation of multiclass image classification by support vector machines", IEEE Transactions on Geoscience and Remote Sensing, vol. 42, no. 6, **(2004).**
[15] A. Bosch, A. Zisserman and X. Munoz, "Scene classification via PLSA", Proceedings of 9th European Conference on Computer Vision, **(2006)** May 7-13, Graz, Austria.
[16] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos", Proceedings of IEEE 10th International Conference on Computer Vision, **(2003)** October 14-17, Nice, France.
[17] L. I. Smith, "A tutorial on principal components analysis", 2011 IEEE Conference on Computer Vision and Pattern Recognition, **(2011)** June 20-25, Colorado Springs, CO, USA.
[18] J. C. Prncipe, W. Liu and S. Haykin, "Kernel Adaptive Filtering: A Comprehensive Introduction", John Wiley and Sons, River Street, **(2011).**
[19] J. Weston and C. Watkins, "Multi-class support vector machines", Tech. Rep., **(1998)**.
[20] C. Chang and C. Lin, "Libsvm: a library for support vector machines", ACM Transaction on Intelligent Systems and Technology, vol. 2, no. 3, **(2011).**
[21] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization", The Journal of Machine Learning Research, vol. 13, **(2012)**.
[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss and V. Dubourg, "Scikit-learn: Machine learning in python", The Journal of Machine Learning Research, vol. 12, **(2011)**.
[23] S. Lazebnik, C. Schmid and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories", Proceedings of 2006 IEEE Conference on Computer Vision and Pattern Recognition, **(2006)** June 17-22, Minnesota, USA.

Authors

Lin Li, Ph. D. candidate at the School of Computer Science and Engineering, University of Electronic Science and Technology of China. He is an associate professor, system analyzer of computer and software technology of P.R. China and student member of China Computer Federation. His research interest covers machine learning and its application in computer image processing and vision. Corresponding author of this paper.)

Jin Lian, got his master degree from the Southwest Jiantong University, a lecture at Sichuan TOP IT Vocational Institute. His interest covers image processing, machine learning, and their computer applications .

Yue Wu, Professor and doctor supervisor at the School of Computer Science and Engineering, University of Electronic Science and Technology of China. His interest covers computer network and data mining.

Mao Ye, is professor and doctor supervisor at the School of Computer Science and Engineering. His interest covers data mining, computer vision, intelligence information processing, *etc.*