Producing computationally efficient KPCA-based feature extraction for classification problems

Y. Xu, C. Lin and W. Zhao

An improvement to kernel principal component analysis (KPCA) to produce computationally efficient KPCA-based feature extraction is proposed. This improvement is applicable to all cases no matter whether the samples in the feature space have zero mean or not. Experiments on several benchmark datasets show that the improvement performs well in classification problems.

Introduction: Kernel principal component analysis (KPCA) has been widely used as a transform method for classification problems. When KPCA extracts features from a sample, it should calculate as many kernel functions as there are total training samples in advance [1]. As a result, the larger the size of the training sample set, the lower the computational efficiency of feature extraction. Indeed, this is a common problem of most kernel methods [2–5]. To produce computationally efficient kernel-methods-based feature extraction procedures, some improvements to kernel methods have been proposed. Among these improvements, the methods in [1, 3, 4] try their best to comply with the nature of the kernel methods when improving the naïve kernel methods. These methods obtain satisfactory improvement in computational efficiency. On the other hand, we also note that, when the method in [1] improves KPCA, it is assumed that the sample data in the feature space has zero mean, which is not so in actual situations. In this Letter, we propose an improvement to KPCA. This improvement is more suitable for real-world data. Indeed, it is applicable to all cases no matter whether the samples in the feature space have zero mean or not.

Method: KPCA is equivalent to a PCA method implemented in the feature space induced by the kernel trick [1]. We know that the eigen-equation of non-centred KPCA (referred to as naïve KPCA 1) is as follows:

$$K\alpha = \lambda\alpha \tag{1}$$

where the elements of matrix K are defined as follows: $(K)_{ij} = k(x_i, x_j)$, $i,j = 1,2,\ldots,N$, $k(x_i, x_j)$ is the kernel function of x_i and x_j. The term 'non-centred KPCA' indeed implies that (1) is obtained under the condition that the samples of the feature space have zero mean. In other words, we can derive (1) from the covariance matrix of the samples of the feature space only if the samples of the feature space have zero mean. However, it is likely that, in real-world applications, the above condition is not satisfied.

We also know that the eigen-equation of centred KPCA (referred to as naïve KPCA 2) is as follows:

$$K'\alpha = \lambda\alpha \tag{2}$$

where $K' = K - LK - KL + LKL$. L is an $N \times N$ matrix the elements of which are all $1/N$. The term 'centred KPCA' means that (2) is obtained without the zero mean assumption as shown above. In other words, K' in (2) is directly derived from the original definition of the covariance matrix of the samples of the feature space. As a result, naïve KPCA 2 can be applicable to all cases no matter whether the samples in the feature space have zero mean or not.

In this Letter, we modify naïve KPCA 2 to produce computationally efficient KPCA-based feature extraction. We refer to our improvement to KPCA proposed in this Letter as improvement to naïve KPCA 2. Compared with the improvement to KPCA described in [1], improvement to naïve KPCA 2 has wider applicability.

As shown in [1], we also assume that the eigenvector of the covariance of the samples of the feature space can be expressed by a certain linear combination of $x_1^0, x_2^0, \ldots, x_s^0$, a subset of the training sample set. $x_1^0, x_2^0, \ldots, x_s^0$ are referred to as 'nodes'. The eigen-equation of improvement to naïve KPCA 2 is as follows:

$$K^o\alpha = \lambda K_2\alpha \tag{3}$$

where $K^o = K_1K_1^T - L_1K_1K_1^T - K_1K_1^TL_1 + L_1K_1K_1^TL_1$, $(K_2)_{ij} = k(x_i^0, x_j^0)$, $i,j = 1,2,\ldots,s$. $(K_1)_{mn} = k(x_m^0, x_n)$, $m = 1,2,\ldots,s$, $j = 1,2,\ldots,n$. L is an $s \times s$ matrix the elements of which are all $1/s$. Actually, we adopt a method similar to the method of reformulating KPCA in [1] to obtain (3).

We determine $x_1^0, x_2^0, \ldots, x_s^0$ using the following procedure:

Step 1. Determine the first node

We take all the training samples as candidates to the first node. We assess all the candidates, respectively. When we assess the ith training sample x, we first calculate K_1, K_2, λ using $K_1 = \left[k(x_i, x_1) \quad k(x_i, x_2) \quad \ldots \quad k(x_i, x_N)\right]$, $K_2 = [k(x_i, x_i)]$ and $\lambda = K^o/K_2$, respectively. We take the candidate that has the maximum λ as the first node and denote it by x_1^0. Then, the matrices K_1, K_2 corresponding to x_1^0 are recorded as K_1^0, K_2^0, i.e. $K_1^0 = [k(x_1^0, x_1) \quad k(x_1^0, x_2) \quad \ldots \quad k(x_1^0, x_N)]$, $K_2^0 = [k(x_1^0, x_1^0)]$.

Step l. Determine the lth node

If $l-1$ nodes, $x_1^0, x_2^0, \ldots, x_{l-1}^0$, have been determined by the previous $l-1$ steps, we determine the lth node as follows. First, a vector k_p^1 is defined as

$$k_p^1 = [k(x_p, x_1), k(x_p, x_2), \ldots, k(x_p, x_N)] \tag{4}$$

Let K_1^0, K_2^0 respectively represent the matrices K_1, K_2 based on $x_1^0, x_2^0, \ldots, x_{l-1}^0$, i.e. $(K_1^0)_{ij} = k(x_i^0, x_j)$, $i = 1,2,\ldots,l-1$, $j = 1,2,\ldots,N$; $(K_2^0)_{ij} = k(x_i^0, x_j^0)$, $i,j,= 1,2,\ldots,l-1$. The lth node should be from the sample set $P = \{x_1, x_2, ..., x_N\} - \{x_1^0, x_2^0, \ldots, x_{l-1}^0\}$, which is a subset of the set of the total training samples. In this step, we will take each element of P as one candidate to the lth node and respectively assess them and then select the optimal candidate as the lth node. When assessing one sample (i.e. one element) x_p from P, we define K_1, K_2 as $K_1 = \begin{bmatrix} K_1^0 \\ k_p^1 \end{bmatrix}$, $K_2 = \begin{bmatrix} K_2^0 & (k_p^2)^T \\ k_p^2 & k(x_p, x_p) \end{bmatrix}$, where k_p^1 is defined as in (4), $k_p^2 = [k(x_p, x_1^0) \quad k(x_p, x_2^0) \ldots k(x_p\ x_{l-1}^0)]$. Using the K_1, K_2, we construct an eigenvalue equation in the form of (3), and then we calculate its eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_l$. Suppose that m eigenvectors of (3) are required. We introduce a variable v and define it as follows: when $l \le m$, let $v = \lambda_1 + \lambda_2 + \ldots + \lambda_l$; otherwise, let $v = \lambda_1 + \lambda_2 + \ldots + \lambda_m$. After we analyse all the elements in P using the above procedure, we denote the maximum v by v_l. Then, we select the candidate associated with v_l, as the lth node and represent it by x_l^0. We do not terminate the above procedure until s reaches a predefined value, where s is the number of the determined nodes.

Improvement to naïve KPCA 2 extracts features from sample x using the following equation: $f = [\sum_{j=1}^{s}\alpha_j^{(1)}k(x_j^0, x)/\sqrt{\lambda_1}\ \sum_{j=1}^{s}\alpha_j^{(2)}k(x_j^0, x)/\sqrt{\lambda_2}\ \ldots\ \sum_{j=1}^{s}\alpha_j^{(m)}k(x_j^0, x)/\sqrt{\lambda_m}]^T$, where $\alpha^{(i)} = [\alpha_1^{(i)} \quad \alpha_2^{(i)} \quad \ldots \quad \alpha_s^{(i)}]^T$. $\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(m)}$ are the m eigenvectors corresponding to the first m largest eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_m$ of (3), which is constructed on the basis of the determined nodes $x_1^0, x_2^0, \ldots, x_s^0$ and all the training samples $x_1, x_2, \ldots, x_N$. It is clear that the computational complexities of our method-based feature extraction and naïve-KPCA-based feature extraction are $o(ms)$ and $o(mN)$, respectively.

Results: We have conducted experiments on benchmark datasets [1] to test different KPCA methods. Each dataset consists of 100 training subsets and 100 testing subsets. We used the Gaussian kernel function $k(x, y) = \exp(-\|x - y\|^2/2\sigma^2)$. For every dataset, σ^2 was set to the square of the Frobenius norm of the covariance matrix of the samples in the first training subset. We took the first training subset as the training set, and took all the testing subsets as the testing sets. We used the nearest neighbour classifier to classify the testing sets.

Tables 1, 2, 3 and 4 show the means of the classification error rates on the datasets 'breast-cancer', 'diabetes', 'heart' and 'thyroid', respectively. From these Tables, we see that improvement to naïve KPCA 2 is able to obtain a lower classification error rate than IKPCA, i.e. the improved KPCA method in [1].

Table 1: Classification error rates of different KPCA methods on dataset 'breast-cancer'

Methods	Naïve KPCA 1	Naïve KPCA 2	IKPCA ($s = 60$)	Improvement to naïve KPCA 2 ($s = 60$)
$m = 20$	9.8182	10.169	9.8442	8.6364

Table 2: Classification error rates of different KPCA methods on dataset 'diabetes'

Methods	Naïve KPCA 1	Naïve KPCA 2	IKPCA ($s = 117$)	Improvement to naïve KPCA 2 ($s = 117$)
$m = 20$	11.707	11.873	11.85	11.633

Table 3: Classification error rates of different KPCA methods on dataset 'heart'

Methods	Naïve KPCA 1	Naïve KPCA 2	IKPCA ($s = 76$)	Improvement to naïve KPCA 2 ($s = 76$)
$m = 20$	7.35	7.35	8.08	7.77

Table 4: Classification error rates of different KPCA methods on dataset 'thyroid'

Methods	Naïve KPCA 1	Naïve KPCA 2	IKPCA ($s = 77$)	Improvement to naïve KPCA 2 ($s = 77$)
$m = 20$	0.98667	0.98667	1.8133	1.8133

Conclusions: Since naïve KPCA 2 is more applicable than naïve KPCA 1, it is probable that the improvement to naïve KPCA 2, proposed in this Letter, is theoretically also more applicable than the improved KPCA in [1]. Improvement to naïve KPCA 2 not only extracts features much more computationally efficient than naïve KPCA, but also classifies very accurately.

Acknowledgments: This Letter is partly supported by Program for New Century Excellent Talents in University (NCET-08-0156) and NSFC under grant 60602038.

7 October 2009
doi: 10.1049/el.2010.2814

Y. Xu and C. Lin (*Bio-Computing Research Center, Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen 518055, People's Republic of China*)

E-mail: laterfall2@yahoo.com.cn

W. Zhao (*Department of Mathematics, Wuhan University of Technology, Wuhan, Hubei 430070, People's Republic of China*)

References

1 Xu, Y., Zhang, D., Song, F., Yang, J.-Y., Jing, Z., and Li, M.: 'A method for speeding up feature extraction based on KPCA', *Neurocomputing*, 2007, **70**, (4–6), pp. 1056–1061
2 Yang, J., Jin, Z., Yang, J.-Y., and Zhang, D.: 'Essence of kernel Fisher discriminant: KPCA plus LDA', *Pattern Recognit.*, 2004, **37**, pp. 2097–2100
3 Xu, Y., Yang, J.-Y., Lu, J., and Yu, D.-J.: 'An efficient renovation on kernel Fisher discriminant analysis and face recognition experiments', *Pattern Recognit.*, 2004, **37**, pp. 2091–2094
4 Xu, Y., Yang, J.-Y., and Yang, J.: 'A reformative kernel Fisher discriminant analysis', *Pattern Recognit.*, 2004, **37**, pp. 1299–1302
5 Twining, C., and Taylor, C.: 'Kernel principal component analysis and the construction of non-linear active shape models'. British Machine Vision Conf., Manchester, UK, 2001, Vol. 1, pp. 23–32