Knowledge-Based Systems 37 (2013) 388–393

Contents lists available at SciVerse ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Random projection ensemble learning with multiple empirical kernels

Zhe Wang [a,*], Wenbo Jie [a], Songcan Chen [b], Daqi Gao [a]

[a] Department of Computer Science & Engineering, East China University of Science & Technology, Shanghai 200237, PR China
[b] Department of Computer Science & Engineering, Nanjing University of Aeronautics & Astronautics, Nanjing 210016, PR China

ARTICLE INFO

Article history:
Received 24 March 2012
Received in revised form 11 July 2012
Accepted 20 August 2012
Available online 12 October 2012

Keywords:
Multiple kernel learning
Empirical mapping
Random projection
Ensemble classifier
Pattern recognition

ABSTRACT

In this paper we propose an effective and efficient random projection ensemble classifier with multiple empirical kernels. For the proposed classifier, we first randomly select a subset from the whole training set and use the subset to construct multiple kernel matrices with different kernels. Then through adopting the eigendecomposition of each kernel matrix, we explicitly map each sample into a feature space and apply the transformed sample into our previous multiple kernel learning framework. Finally, we repeat the above random selection for multiple times and develop a voting ensemble classifier, which is named RPEMEKL. The contributions of the proposed RPEMEKL are: (1) efficiently reducing the computational cost for the eigendecomposition of the kernel matrix due to the smaller size of the kernel matrix; (2) effectively increasing the classification performance due to the diversity generated through different random selections of the subsets; (3) giving an alternative multiple kernel learning from the Empirical Kernel Mapping (EKM) viewpoint, which is different from the traditional Implicit Kernel Mapping (IKM) learning.

1. Introduction

Kernel-based learning is successfully applied in machine learning [10,11,14,15,19,20]. The kernel-based learning transforms the input space into a feature space and works in the feature space, where the transformation is achieved through the mapping $\Phi(x): x \to \mathcal{F}$. In the existing kernel-based learning, there are two kinds of $\Phi(x)$ including implicit and explicit representations denoted as $\Phi^i(x)$ and $\Phi^e(x)$. The implicit mapping $\Phi^i(x)$ called Implicit Kernel Mapping (IKM) [10] is achieved through introducing a kernel function $k(x_i,x_j)$ that can determine the geometrical structure of the mapped data in the feature space, where we need not obtain the form of the $\Phi^i(x)$ but just compute the $k(x_i,x_j) = \Phi^i(x_i)\cdot\Phi^i(x_j)$. In contrast, the $\Phi^e(x)$ called Empirical Kernel Mapping (EKM) [16] should explicitly give all the features of x in the mapped Φ^e-space.

On the other hand, the kernel-based learning can also be sorted into Single Kernel Learning (SKL) and Multiple Kernel Learning (MKL) according to the number of the used kernels in learning processing. SKL selects one optimized kernel from a set of candidates. MKL syncretizes and uses multiple kernels in learning processing. Therefore, in our opinion there are naturally at least four combinations: the SKL with IKM, the SKL with EKM, the MKL with IKM and the MKL with EKM also denoted as MEKL. In traditional kernel-based methods, the SKL with either IKM or EKM is widespread [10,16]. Since MKL can increase performance effectively, the MKL with IKM has recently got much attention [7,13]. In contrast, It is less attracted for the MKL with EKM except our previous work MultiK-MHKS [14].

The MultiK-MHKS explicitly maps the input samples into M feature spaces with given M different kernels. Then, it constructed a term R_{IFSL} called Inter-Function Similarity Loss and introduced R_{IFSL} into a regularization framework. In the MultiK-MHKS, we explicitly mapped each sample $x \in \mathbb{R}^d$ into $\Phi^e(x) \in \mathbb{R}^r$ through the whole training set $S = \{(x_i, \varphi_i)\}_{i=1}^N$ for M times with the given M kernels. With each kernel $k(x_i, x_j)$, we generated the kernel matrix $K \in \mathbb{R}^{N\times N}$ with the set S and carried out the eigendecomposition of the K as follows

$$K = Q_{N\times r}\Lambda_{r\times r}Q_{r\times N}^T, \tag{1}$$

where r is the rank of the K. Then through letting $R = Q\Lambda^{1/2}$, each sample x was explicitly mapped through the following form

$$\Phi^e(x)_{r\times 1} = R_{r\times N}^{-1}[k(x,x_1),\ldots,k(x,x_N)]_{N\times 1}^T. \tag{2}$$

It can be found that it would take a computational cost with $o(N^3)$ for the Eq. (1). If the N is large, it would be much larger for the computational cost of the eigendecomposition of the K.

To reduce the cost for the eigendecomposition of the K, in this paper we first adopt the strategy of the Random Projection (RP). RP aims to project the original high-dimensional data onto a lower-dimensional space with a random matrix [1–3]. From the

* Corresponding author. Tel.: +86 15000779526.
E-mail addresses: wangzhe@ecust.edu.cn (Z. Wang), pilixiaoxuanfeng@gmail.com (W. Jie), s.chen@nuaa.edu.cn (S. Chen), gaodaqi@ecust.edu.cn (D. Gao).

0950-7051/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.knosys.2012.08.017

Eq. (2), it is found that the sample x is transformed into a r-dimensional kernel space through the whole training set S with the size N. Here, based on the random characteristic of RP, we randomly select a subset $S' = \{(x_i, \varphi_i)\}_{i=1}^{p}$, $p < N$ from the whole set S and use the S' to construct M kernel matrices $K_l \in \mathbb{R}^{p\times p}$ with M different kernels. Through adopting the eigendecomposition for the K_l instead of the original K, we explicitly map each sample x into $\Phi_l^e(x), l = 1 \dots M$ based on the Eq. (2). Due to changing the transformed size for the x in the Eq. (2), we can reduce the computational cost of the matrix decomposition from $o(N^3)$ to $o(p^3)$. Subsequently, we apply the transformed sample $\Phi_l^e(x), l = 1 \dots M$ into our previous MEKL framework [14].

Secondly, in order to prevent the proposed classifier from reducing classification performance, we further give an ensemble scheme with voting. In detail, we randomly select the subset S' for multiple times and generate multiple $S'_q, q = 1 \dots H$. Subsequently, for each $S'_q, q = 1 \dots H$ we repeat the explicitly mapping (2) and the following learning applied into our previous MEKL [14]. The final ensemble is produced through a voting scheme that is applied to the classification outcomes of all the classifiers generated from $S'_q, q = 1 \dots H$. The whole procedure is named Random Projection Ensemble Learning with Multiple Empirical Kernels denoted RPEMEKL for short. The advantages of the proposed RPEMEKL are highlighted as follows: (1) reducing the computational cost for the eigendecomposition of the kernel matrix from $o(N^3)$ to $o(Hp^3)$; (2) increasing the classification performance due to the diversity generated through different random selections of the S'_qs; (3) giving an alternative multiple kernel learning from the Empirical Kernel Mapping (EKM) viewpoint, which is different from the traditional Implicit Kernel Mapping (IKM) learning.

The rest of this paper is organized as follows. Section 2 gives the architecture of the proposed RPEMEKL. Section 3 demonstrates the feasibility and effectiveness of the proposed RPEMEKL in terms of classification and computation. Finally, both conclusion and future work are given in Section 4.

2. Random Projection Ensemble Learning with Kernels (RPEMEKL)

The proposed RPEMEKL is made up of two parts. The first part is to randomly select a subset S' from the whole set S based on the random characteristic of RP and apply the transformed samples from the S' onto our previous MEKL framework [14]. The second part is to repeat the first part for multiple times based on different random selections for S' and to construct a voting ensemble with different classifiers generated from the first part.

Suppose that there is a training sample set $S = \{(x_i, \varphi_i)\}_{i=1}^{N} \subset \mathcal{X}$, where $x_i \in \mathbb{R}^d$ and its corresponding class label $\varphi_i \in \{+1, -1\}$. In the first part, we randomly select a sample subset $S' = \{(x_i, \varphi_i)\}_{i=1}^{p}$ from the whole set S. Then we adopt the subset S' to generate the kernel matrix $K_l \in \mathbb{R}^{p\times p}$ with one certain kernel $k_l(x_i, x_j)$ and carry out the eigendecomposition of each K_l as follows

$$K_l = Q_l \Lambda_l Q_l^T, \tag{3}$$

where $\Lambda_l \in \mathbb{R}^{r_l \times r_l}$ is a diagonal matrix consisting of the r_l positive eigenvalues of K_l, and $Q_l \in \mathbb{R}^{p\times r_l}$ consists of the corresponding orthonormal eigenvectors. Here we adopt the EKM for each mapping and thus each input sample x can be mapped into $\Phi_l^e(x)$ through the following equation

$$\Phi_l^e(x) = \Lambda_l^{-1/2} Q_l^T [k(x, x_1), \dots, k(x, x_p)]^T, \tag{4}$$

where $\Phi_l^e(x) \in \mathbb{R}^{r_l}$. The feasibility of doing so can be guaranteed by the characteristic of RP in kernel-based learning, which is that the learning based on the mapping (4) can lead to an approximate separability at one certain margin [3]. Since our previous work MultiK-MHKS [14] makes the p in the (4) with the size of the whole input training samples N and in this paper we adopt a smaller p, the computational cost for the eigendecomposition in the Eq. (3) can decrease from $o(N^3)$ in the MultiK-MHKS to $o(p^3)$ in the proposed method.

Further, the proposed method explicitly maps all the input samples of the set S into the transformed sample set $\{(\Phi_1^e(x_i), \dots, \Phi_l^e(x_i), \dots, \Phi_M^e(x_i))\}_{i=1}^{N}$ with the Eq. (4) and M given kernels. Then, we introduced the generated $\{(\Phi_1^e(x_i), \dots, \Phi_l^e(x_i), \dots, \Phi_M^e(x_i))\}_{i=1}^{N}$ onto our previous MEKL framework. In detail, we adopt the Modification of Ho-Kashyap algorithm with Squared approximation of the misclassification errors (MHKS) [8] as the base classifier. Therefore for the proposed RPEMEKL, we give the following optimization problem:

$$\begin{aligned} \min_{\substack{\omega_l \in \mathbb{R}^{r_l+1}, b_l \geqslant 0 \\ l = 1 \dots M}} L = & \sum_{l=1}^{M} \Big[(Y_l\omega_l - 1_{N\times 1} - b_l)^T (Y_l\omega_l - 1_{N\times 1} - b_l) \\ & + c_l \tilde{\omega}_l^T \tilde{\omega}_l \Big] + \lambda \sum_{l=1}^{M} (Y_l\omega_l \\ & - \frac{1}{M}\sum_{j=1}^{M} Y_j\omega_j)^T \left(Y_l\omega_l - \frac{1}{M}\sum_{j=1}^{M} Y_j\omega_j \right), \end{aligned} \tag{5}$$

where $\omega_l = [\tilde{\omega}_l{}^T, \omega_{0l}]^T$, $\tilde{\omega}_l \in \mathbb{R}_l^r, \omega_{0l} \in \mathbb{R}$ are the augmented weight vector, the weight vector and the bias respectively; the matrix Y_l is defined as $Y_l = \left[\varphi_1\left(\Phi_l^e(x_1)^T, 1\right); \dots; \varphi_N\left(\Phi_l^e(x_N)^T, 1\right)\right]$; $1_{N\times 1}$ represents the vector of N dimension with all entries equal to 1; $b_l \in \mathbb{R}^{N\times 1}$ represents the vector with all entries equal to nonnegative values and the regularized parameters $c_l, \lambda \geqslant 0 \in \mathbb{R}$. In the optimization problem (5), Y_l, ω_l, b_l correspond to one MHKS in one Φ_l^e-space that is determined by the corresponding $\{(\Phi_l^e(x_i), \varphi_i)\}_{i=1}^{N}$. In the right side of the Eq. (5), the first term corresponds to the principle of MHKSs in M views. The second term is to syncretize the M MHKSs, which denotes that the outputs of the samples $\{(\Phi_l^e(x_i), \varphi_i)\}_{i=1}^{N}$ in each Φ_l^e-space onto their corresponding weight vector ω_l are constrained to be maximally close to the average outputs of all the feature spaces.

In the optimizing processing for the (5), we employ a modification of the gradient descent with a heuristic update-rule for each ω_l. Through making the gradient of the L of the (5) with respect to the ω_l be zero, we can obtain

$$\begin{aligned} \frac{\partial L}{\partial \omega_l} = 0 \Longleftrightarrow \omega_l \\ = \left[\left(1 + \lambda\frac{M-1}{M}\right) Y_l^T Y_l + c_l \widetilde{I}_l \right]^{-1} Y_l^T \left(b_l + 1_{N\times 1} + \lambda \frac{1}{M} \sum_{j=1; j\neq l}^{M} Y_j\omega_j \right), \end{aligned} \tag{6}$$

where $\widetilde{I}_l$ is a diagonal matrix with full ones except the last element set to zero. In the lth Φ_l^e-space, it can be noted that ω_l depends on b_l from the Eq. (6). Then by differentiating the L in Eq. (5) with respect to b_l and setting the result equal to zero, we can get

$$\frac{\partial L}{\partial b_l} = 0 \Longleftrightarrow e_l = Y_l\omega_l - b_l - 1_{N\times 1}. \tag{7}$$

Then, we adopt the iterative algorithm for determining ω_l and b_l similarly to that in [8]. First, with b_l^k representing the vector b_l at the kth iteration, we initialize $b_l^1 \geqslant 0$, then keep $b_l^k \geqslant 0$ at each iteration, and finally obtain

$$\begin{cases} b_l^1 \geqslant 0 \\ b_l^{k+1} = b_l^k + \rho_l (e_l^k + |e_l^k|) \end{cases}, \tag{8}$$