

# Spectral Kernels for Classification

Wenyuan Li<sup>1</sup>, Kok-Leong Ong<sup>2</sup>, Wee-Keong Ng<sup>1</sup>, and Aixin Sun<sup>3</sup>

<sup>1</sup> Nanyang Technological University, Centre for Advanced Information Systems  
Nanyang Avenue, N4-B3C-14, Singapore 639798  
liwy@pmail.ntu.edu.sg, awkng@ntu.edu.sg

<sup>2</sup> School of Information Technology, Deakin University  
Waurm Ponds, VIC 3217, Australia  
leong@deakin.edu.au

<sup>3</sup> School of Computer Science and Engineering, University of New South Wales  
Sydney, NSW 2052, Australia  
aixinsun@cse.unsw.edu.au

**Abstract.** Spectral methods, as an unsupervised technique, have been used with success in data mining such as LSI in information retrieval, HITS and PageRank in Web search engines, and spectral clustering in machine learning. The essence of success in these applications is the spectral information that captures the semantics inherent in the large amount of data required during unsupervised learning. In this paper, we ask if spectral methods can also be used in supervised learning, e.g., classification. In an attempt to answer this question, our research reveals a novel kernel in which spectral clustering information can be easily exploited and extended to new incoming data during classification tasks. From our experimental results, the proposed Spectral Kernel has proved to speedup classification tasks without compromising accuracy.

## 1 Introduction

Kernel-based learning first appear in the form of Support Vector Machines (SVM), and readily became the state-of-the-art for learning algorithms. The framework of kernel-based learning methods (KM) is also known as kernel-based analysis of data in both supervised and unsupervised learning [1–3]. Within this framework, kernels encode all the information required by the learning machinery, and acts as the interface between the data and the learning modules [4]. Hence, they are implicitly high-dimensional spaces that contain more information than the original explicit feature space. The advantage of this is that once obtained, kernel algorithms can perform analysis without further information from the original data set.

There have been many success stories [5–8] with kernels. In text categorization, the kernel was used to capture a semantic network of terms to better compute the similarities between documents [9]. In natural language learning, subparse trees are taken into consideration in the semantic kernel to improve the accuracy of classifying predicative arguments [10]. And in image retrieval, the knowledge about users' queries are encoded in the kernel to improve query accuracy [11]. While domain knowledge is usually encoded in the kernel by the expert user, they can also be obtained from automated

discovery algorithms. The pioneering attempt to integrate unsupervised discovery, in the form of kernels, for supervised learning is Latent Semantic Indexing (LSI). It has been shown [12] that the Latent Semantic Kernel (LSK) benefits from the automated discovery of latent semantics that aid the task of classification. In fact, the semantics uncovered in LSI is simply the ‘tip of the iceberg’ of spectral graph analysis on kernel matrices. Under spectral graph theory, there have been active research on the use of latent semantics for clustering. This research, known as *spectral clustering*, is a method that uses spectral information to assist clustering algorithms.

Obtaining the spectral information of a data set is a three step process: (i) compute the similarity matrix  $\mathbf{S}$  from the data; (ii) transform  $\mathbf{S}$  to another matrix  $\Gamma(\mathbf{S})$ ; and finally (iii) perform an eigen-decomposition on  $\Gamma(\mathbf{S})$ . Our analysis of this process led to an important discovery — if certain matrix transformation (e.g., normalized Laplacian) is performed, we can observe some interesting latent clustering semantics in the eigenvalues and eigenvectors of  $\Gamma(\mathbf{S})$  [5, 13–17] that can be used in the kernel for the task of classification. Our observation, and hence the main contribution of this paper, led to our proposal of the *Spectral Kernel*. The spectral kernel combines two state-of-the-art learning algorithms: kernel-based learning and spectral clustering; and introduces a mechanism that supports the spectral embedding of new input data into the kernel to improve classification precision.

We present our proposal as follows. The next section provides the background about kernels and spectral clustering. In doing so, we provide the theoretical analysis and examples to demonstrate the steps to compute the spectral embedding space. We then present in Section 3, the steps to update the kernel values as new input arrives — a differentiation of our approach from other spectral learning methods. We then provide empirical results in Section 4 to support the feasibility of our proposal. Finally, we conclude with related and future work in Section 5.

## 2 Spectral Graph Analysis of Kernel Matrices

To facilitate understanding of our proposal, as well as the analysis and proofs presented in the later sections of this paper, we first introduce some basic facts of kernel matrices and its mathematical foundation [13].

Given a set of data points  $D = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$  and a kernel function  $\kappa(\cdot, \cdot)$ , the kernel matrix  $\mathbf{K} = (\mathbf{K}_{ij})_{i,j=1}^n$  is defined as  $\mathbf{K}_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$ , where  $\mathbf{K}$  is symmetric and usually positive semi-definite. By operating on  $\mathbf{K}$ , we can easily recode the data in a manner suitable for the learning module. A simple and widely used  $\kappa$  is the inner product  $\kappa_I(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$ . And if we have  $\kappa_1(\mathbf{x}, \mathbf{z})$  as a kernel, we can construct new kernels using other kernel functions, e.g., exponential kernel  $\kappa_E(\mathbf{x}, \mathbf{z}) = \exp(\kappa_1(\mathbf{x}, \mathbf{z}))$ ; polynomial kernel  $\kappa_P(\mathbf{x}, \mathbf{z}) = (\kappa_1(\mathbf{x}, \mathbf{z}) + d)^p$  with positive coefficients; and Gaussian kernel  $\kappa_G(\mathbf{x}, \mathbf{z}) = \exp((\kappa_1(\mathbf{x}, \mathbf{x}) + \kappa_1(\mathbf{z}, \mathbf{z}) - 2\kappa_1(\mathbf{x}, \mathbf{z})) / (2\sigma^2))$ .

In many cases,  $\kappa$  need not be an explicit function if the kernel matrix can be given directly. Examples of that include the Latent Semantic Kernel and our proposed Spectral Kernel. The idea is that while some kernels can be represented using explicit functions, many are implicitly represented without one. Regardless of whether  $\kappa$  is an explicit function, the matrix serves as the underlying representation of a kernel capturing all the

**Table 1.** Solution to two graph cut criteria used in spectral clustering: (i) the corresponding transformation by eigen-decomposition of the matrix  $\Gamma(\mathbf{S})$ ; (ii) on an incoming data  $\mathbf{x}$ , the similarity with the training examples is computed as  $\mathbf{S}_x$  and its corresponding transformation is  $\tau(\mathbf{S}_x)$ . *Note:* the original version of the normalized Laplacian matrix should be  $\Gamma_N(\mathbf{S}) = \mathbf{D}^{-1/2}(\mathbf{D} - \mathbf{S})\mathbf{D}^{-1/2}$ ; see Section 2.2 and Lemma 1.

		Average Volume	Normalized Cut
	Criterion	$\max \frac{\text{vol}(A)}{ A } + \frac{\text{vol}(B)}{ B }$	$\min \frac{\text{cut}(A,B)}{\text{vol}(A)} + \frac{\text{cut}(A,B)}{\text{vol}(B)}$
	Solution to criterion	$\mathbf{S}\mathbf{x} = \lambda\mathbf{x}$	$(\mathbf{D} - \mathbf{S})\mathbf{x} = \lambda\mathbf{D}\mathbf{x}$
(i)	Transformed $\mathbf{S}$	$\Gamma_I(\mathbf{S}) = \mathbf{S}$	$\Gamma_N(\mathbf{S}) = \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{-1/2}$
(ii)	Transformed $\mathbf{S}_x$	$\tau_I(\mathbf{S}_x) = \mathbf{S}_x$	$\tau_N(\mathbf{S}_x) = \mathbf{D}^{-1/2}\mathbf{S}_x\mathbf{d}_x^{-1/2}$

information required for supervised or unsupervised learning. More interesting perhaps, is that the underlying idea found in spectral embedding and clustering methods (proposed in recent years) coincides with that of kernel-based methods, i.e., a symmetric matrix is used in the analysis. As a result, there are some interesting properties that we can learn about kernels through spectral properties.

A symmetric matrix  $\mathbf{S} = (\mathbf{S}_{ij})_{n \times n}$  (where  $\mathbf{S}_{ij} = \mathbf{S}_{ji}$ ) is naturally mapped to an undirected graph  $\mathcal{G}(\mathbf{S})$ , where its adjacency matrix is  $\mathbf{S}$ . In spectral graph theory, the spectral component of the transformed  $\mathbf{S}$  has a natural relationship with the structure and properties of the graph  $\mathcal{G}(\mathbf{S})$  [13]. Further let  $\mathcal{G}(\mathbf{S}) = \langle V, E, \mathbf{S} \rangle$  be the graph of  $\mathbf{S}$ , where  $V$  is the set of  $n$  vertices and  $E$  is the set of weighted edges. Each vertex  $i$  of  $\mathcal{G}(\mathbf{S})$  corresponds to the  $i$ -th column (or row) of  $\mathbf{S}$ , and the weight of each edge  $\widehat{ij}$  corresponds to the non-diagonal entry  $\mathbf{S}_{ij}$ . For any two vertices  $(i, j)$ , a larger value of  $\mathbf{S}_{ij}$  indicates a higher connectivity, and vice versa. From the above, we have the following interesting spectral properties:

**Eigenvalues** The spectrum of the Normalized Laplacian transformation of  $\mathbf{S}$  reveals the embedding clustering structure of  $\mathcal{G}(\mathbf{S})$  with different global bisection (or cut) criteria [5, 13].

**Eigenvectors** Correspondingly, the  $i$ -th eigenvector naturally explains the meaning of the  $i$ -th eigenvalue. This led to the development of spectral clustering [15–17].

## 2.1 Graph Cut Criteria of Kernel Matrices

In spectral clustering, since  $\mathbf{S}$  is actually an adjacency matrix of the weighted graph  $\mathcal{G}(\mathbf{S})$ , finding the clustering structure of  $\mathbf{S}$  can be transformed into the problem of finding an optimum graph cut in  $\mathcal{G}(\mathbf{S})$ . Notably, a different graph cut criterion leads to a different solution of  $\mathcal{G}(\mathbf{S})$ .

In the case of Table 1, the criterion is to find an optimal cut of  $\mathcal{G}(\mathbf{S})$  such that we have two non-overlapping subsets  $A, B \subseteq V$  satisfying the conditions  $A \cap B = \emptyset$  and  $A \cup B = V$ , where  $|A|$  is the number of vertices or data points;  $\text{vol}(A) = \sum_{i \in A} d_i$  is the volume with  $d_i = \sum_{j \in V} \mathbf{S}_{ij}$  being the degree of the vertex  $i$ ;  $\text{cut}(A, B) = \sum_{i \in A, j \in B} \mathbf{S}_{ij}$  is the cut between  $A$  and  $B$ ; and  $\mathbf{D}$  is the diagonal matrix formed from the degrees of the vertices. In both solutions, the second largest eigenvalues (a.k.a.

interested eigenvalues) and the corresponding eigenvectors relating to the equations in Table 1 provide the global optimum.

Clearly, the idea of bisecting the kernel matrix can be extended to the first  $k$  largest eigenvalues or eigenvectors. And this observation leads to the construction of multi-way spectral clustering. In the criterion of *average volume*, if we consider the inner product matrix of the term-document matrix (without normalization) as  $\mathbf{S}$ , then its first  $k$  largest eigenvectors is used in LSI for information retrieval. In the criterion of *normalized cut*, the second largest eigenvector of  $\Gamma_N(\mathbf{S})$  is also the clustering information used in the normalized cut image segmentation algorithm [15]. Finally, we have the NJW clustering algorithm [17] when we consider the first  $k$  largest eigenvectors of  $\Gamma_N(\mathbf{S})$ .

We can thus conclude the following. First, the criterion determines the solution and transformation of the kernel matrix that in turn, affects the behavior in the learning module. Second, the type of application to be delivered by the kernel is determined by how the eigenvalues and/or eigenvectors are used. The spectral kernel, presented next, is the result of exploiting these observations.

## 2.2 Computing the Spectral Embedding Space

Among the different spectral clustering techniques proposed in the literature, e.g., [15–17], an analysis of the underlying mathematical representation suggests that with appropriate transformations, they essentially reduce to a common representation. This observation motivates the first contribution of the spectral kernel – a unifying framework which by means of different parameters, creates different kernel instances that exhibit different behaviors. We will first prove the existence of this framework, and then show how the spectral embedding is computed.

From Table 1, we see that it is easy to mathematically transform the solutions into a standard eigendecomposition problem of symmetric matrices, i.e.,  $\Gamma(\mathbf{S})\mathbf{x} = \lambda\mathbf{x}$ . To do so however, requires the transformation of  $\mathbf{S}$ , and this is dependent on the solution to the cut criteria. In Table 1,  $\Gamma_I(\mathbf{S})$  represents the original matrix while  $\Gamma_N(\mathbf{S})$  is the normalized Laplacian matrix. From spectral graph theory [13],  $\mathbf{K}_1 = \mathbf{D}^{-1/2}(\mathbf{D} - \mathbf{S})\mathbf{D}^{-1/2}$  is actually the normalized Laplacian matrix. Notably,  $\mathbf{K}_1$  has the same eigenvectors as  $\mathbf{K}_2 = \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{-1/2}$  and the eigenvalues is related by  $\text{eig}(\mathbf{K}_1) = \{1 - \lambda | \lambda \in \text{eig}(\mathbf{K}_2)\}$ , where  $\text{eig}(\cdot)$  is the set of eigenvalues of a symmetric matrix. Furthermore, the interested eigenvalues change from the smallest in  $\mathbf{K}_1$  to the largest in  $\mathbf{K}_2$ . Therefore, it is actually possible to compute the normalized Laplacian matrix using  $\mathbf{K}_2$  giving us  $\Gamma_N(\mathbf{S}) = \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{-1/2}$ . In fact, the equivalence relationship between  $\mathbf{K}_1$ ,  $\mathbf{K}_2$ , and the stochastic matrix  $\mathbf{P} = \mathbf{D}^{-1}\mathbf{S}$  can be proven, and therefore in the remaining part of this paper, we will use  $\mathbf{K}_2$  in place of  $\mathbf{K}_1$  and  $\mathbf{P}$  if any.

**Lemma 1 (Equivalence of  $\mathbf{K}_1$ ,  $\mathbf{K}_2$  and  $\mathbf{P}$ ).** *If  $\lambda$  and  $\mathbf{x}$  are correspondingly the eigenvalue and eigenvector of matrix  $\mathbf{K}_1$ , then  $(1 - \lambda)$  are the eigenvalues of the matrices  $\mathbf{K}_2$  and  $\mathbf{P}$ ; and the eigenvectors of  $\mathbf{K}_2$  and  $\mathbf{P}$  are  $\mathbf{x}$  and  $\mathbf{D}^{-1/2}\mathbf{x}$  respectively.*

*Proof.* By definition of  $\mathbf{K}_1$ ,  $\mathbf{K}_2$  and  $\mathbf{P}$ , we have:

$$\mathbf{K}_1 = \mathbf{I} - \mathbf{K}_2 \tag{1}$$

$$\mathbf{K}_2 = \mathbf{D}^{1/2}\mathbf{P}\mathbf{D}^{-1/2} \tag{2}$$

Suppose  $\lambda$  and  $\mathbf{x}$  are eigenvalue and eigenvector of  $\mathbf{K}_1$ , i.e.,  $\mathbf{K}_1\mathbf{x} = \lambda\mathbf{x}$ . By Equation (1), substituting  $\mathbf{K}_1$  with  $\mathbf{K}_2$  gives us  $(\mathbf{I} - \mathbf{K}_2)\mathbf{x} = \lambda\mathbf{x}$ . After transformation, we have  $\mathbf{K}_2\mathbf{x} = \mathbf{I}\mathbf{x} - \lambda\mathbf{x} = (1 - \lambda)\mathbf{x}$ . This proves the relationship between  $\mathbf{K}_1$  and  $\mathbf{K}_2$ . We next complete the proof by showing the equivalence of  $\mathbf{K}_2$  and  $\mathbf{P}$ . Suppose now  $\lambda$  and  $\mathbf{x}$  are eigenvalue and eigenvector of  $\mathbf{K}_2$ , i.e.,  $\mathbf{K}_2\mathbf{x} = \lambda\mathbf{x}$ . By Equation (2), substituting  $\mathbf{K}_2$  with  $\mathbf{P}$  gives us  $\mathbf{D}^{1/2}\mathbf{P}\mathbf{D}^{-1/2}\mathbf{x} = \lambda\mathbf{x}$ . By left-multiplication of the matrix  $\mathbf{D}^{-1/2}$  on this equation, we get  $\mathbf{P}\mathbf{D}^{-1/2}\mathbf{x} = \lambda\mathbf{D}^{-1/2}\mathbf{x}$ . Here,  $\lambda$  is the eigenvalue of  $\mathbf{P}$ , and  $\mathbf{D}^{-1/2}\mathbf{x}$  is the eigenvector of  $\mathbf{P}$ .

Within this framework, we can compute the spectral embedding for any specific instance of the spectral kernel. The steps to do so are given in Figure 1. After the spectral components of  $\Gamma(\mathbf{S})$  is computed, the  $k$  interested extreme eigenvalues and eigenvectors are selected to construct the reduced data space. Let the first  $k$  interested eigenvalues of  $\Gamma(\mathbf{S})$  be  $\lambda_1 \triangle \lambda_2 \dots \triangle \lambda_k$ , where “ $\triangle$ ” is “ $\leq$ ” or “ $\geq$ ” according to the different matrix transformation  $\Gamma$ , and  $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k$  as their corresponding eigenvectors each of dimension  $n$ . The  $k$  dimensional data space is constructed by the two steps shown in Figure 1. The first step has two implementations that can be selected based on the desired application behavior.

Step 1(a) has been proved to be effective and useful on  $\Gamma_N(\mathbf{S})$  in revealing the clustering structure of  $\mathbf{S}$  in [16]. When considered with Step 2, its effectiveness was proven, both theoretically and empirically, in [17]. When Step 1(a) is used with  $\Gamma_I(\mathbf{S})$ , it has proven applications in latent semantic analysis and indexing [6, 18]. Step 1(b) on the other hand is well-suited in the context of  $k$ -rank approximation when used with  $\Gamma(\mathbf{S})$  as supported by Lemma 2 below. Furthermore, latent semantic analysis has shown that the  $k$ -rank approximation of a similarity matrix (that is also  $\Gamma_I(\mathbf{S})$ ) incorporates semantic information in measure of similarity between two data points (the same conclusion was also given in latent semantic kernels). We will elaborate this point in Section 3.3.

**Lemma 2 (Approximation of  $\Gamma(\mathbf{S})$  by embedding of Step 1(b)).** *The matrix  $\mathbf{S}'$ , computed by the embedding of Step 1(b) using inner product (i.e.,  $\mathbf{S}'_{ij} = \mathbf{y}_i^T \mathbf{y}_j$ ), is the best  $k$ -rank matrix approximation of  $\Gamma(\mathbf{S})$ .*

*Proof.* Lemma 2 is a variant of the Eckart-Young theorem [19]. Given  $\mathbf{S}_{n \times n} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}$  (singular value decomposition),  $\mathbf{A} = \mathbf{S}_k = \mathbf{U}_k\mathbf{\Lambda}_k\mathbf{V}_k$  is the best rank- $k$  approximation to  $\mathbf{S}$  that minimizes  $\|\mathbf{A} - \mathbf{S}\|_F^2$  among all matrices  $\mathbf{A}$  with rank  $k$  ( $F$  denotes Frobenius norm of a matrix). And because  $\mathbf{S}$  is symmetric, singular values and vectors of  $\mathbf{S}$  are the same as eigenvalues and eigenvectors of  $\mathbf{S}$ .

### 3 Spectral Kernels

In the previous section, the spectral graph analysis of the kernel matrix shows that the spectral embedding (obtained by either Step 1(a) or 1(b)) reveals more latent semantics than the original kernel matrix. By projecting the original feature vectors onto the spectral embedding subspace, we can define a kernel, originating from this subspace, through a particular choice of similarity measure. This effectively registers the clustering information inherent in the subspace into the spectral kernel (SK).

**Step 1(a).**

Directly get  $\mathbf{y}_i = (\mathbf{v}_1^i, \mathbf{v}_2^i, \dots, \mathbf{v}_k^i)^T$ , where  $i=1, 2, \dots, n$ .

**Step 1(b).**

Compute  $\mathbf{y}_i = (\sqrt{\lambda_1} \mathbf{v}_1^i, \sqrt{\lambda_2} \mathbf{v}_2^i, \dots, \sqrt{\lambda_k} \mathbf{v}_k^i)^T$ , where  $i=1, 2, \dots, n$  or  $\mathbf{\Lambda}_k^{1/2}(\mathbf{v}_1^i, \mathbf{v}_2^i, \dots, \mathbf{v}_n^i)^T$ .

**Step 2 (Optional).**

Renormalize each  $\mathbf{y}_i$  to have the unit length (i.e.  $\mathbf{y}_i = \frac{1}{\|\mathbf{y}_i\|} \mathbf{y}_i$ ).

**Fig. 1.** Spectral embedding in the spectral kernel. Let the projected  $k$ -dimensional data space be  $\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n \in \mathbb{R}^{k \times 1}$ , and the first  $k$  interested eigenvalues  $\lambda_1, \lambda_2, \dots, \lambda_k$  are positive. *Note:*  $\mathbf{v}_j^i$  denotes the  $i$ -th coordinate of the eigenvector  $\mathbf{v}_j$ .  $\mathbf{\Lambda}_k$  is the truncated diagonal of  $\mathbf{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$ , its last  $(n - k)$  diagonal entries are set 0.

Computing the spectral kernel for classification can be done in three phases: (i) transformation, (ii) spectral embedding, and (iii) kernel computation. Thus, we define the spectral kernel with three components, i.e.,  $\text{SK}(\mathbb{T}, \mathbb{E}, \mathbb{S})$ , where  $\mathbb{T}$  is the transformation in Table 1;  $\mathbb{E}$  is the embedding step in Figure 1; and  $\mathbb{S}$  is one of the similarity measure in Table 2 selected to compute the final spectral kernel value in the spectral embedding subspace.

In classification, the kernel can contain values from either the training set, or from the training set and its new input (from the testing set). Since during transformation and spectral embedding, the input kernel matrix  $\mathbf{S}$  only holds kernel values from the training set, the spectral embedding can be computed by following the steps given in Section 2.2. In the case where we need to compute the kernel values from both the training and testing set, a different way to compute the spectral embedding of the new input within the same subspace of the training set is needed.

### 3.1 Transforming and Spectral Embedding of New Input

When a new input arrives, its spectral embedding is computed in a similar fashion as described in Section 2.2. The difference is that the transformation and computation of spectral embedding is applied to the vector  $\mathbf{S}_x$  rather than the symmetric matrix. This gives rise to a different transformation and computation of the spectral embedding. After getting the spectral embedding of the new input, the kernel values can be updated with the same similarity measure used during training.

The new input can be given in the form of a vector containing kernel values, i.e.,  $\mathbf{S}_x = (\mathbf{S}_{1x}, \mathbf{S}_{2x}, \dots, \mathbf{S}_{nx})^T$ , where  $\mathbf{S}_{ix}$  represents the kernel value between the  $i$ -th training example and itself. The rationale to why we used  $\mathbf{S}_x$  instead of the vector  $\mathbf{x}$  in the original space is that spectral kernels are based on the other input kernels. In order to compute the spectral embedding of the new input, there is a need to recompute the transformation and the embedding space. Therefore, the vector transformation corresponding to its matrix transformation  $\Gamma(\mathbf{S})$  is defined as  $\tau(\mathbf{S}_x)$  and is given in Table 1 for different  $\Gamma$ . After obtaining  $\tau(\mathbf{S}_x)$ , the following lemma defines how the spectral embedding of the new input is computed.

**Lemma 3 (Spectral embedding of new input).** *Given a kernel matrix  $\mathbf{S}$  for training data, its transformation  $\Gamma(\mathbf{S})$ , the  $k$  interested eigenvalues/vectors of  $\Gamma(\mathbf{S})$  ( $\lambda_i \geq$*

**Table 2.** The similarity measures  $s(\mathbf{x}, \mathbf{y})$  used in the computation of spectral kernels.

	Inner product	Extension of Euclidean distance	Pearson correlation coefficient
Symbol	$S_I$	$S_E$	$S_P$
Formula	$\mathbf{x}^T \mathbf{y}$	$\exp(-\frac{\ \mathbf{x}-\mathbf{y}\ ^2}{\sigma})$	$\frac{(\mathbf{x}-\bar{\mathbf{x}})^T(\mathbf{y}-\bar{\mathbf{y}})}{\ \mathbf{x}-\bar{\mathbf{x}}\ \ \mathbf{y}-\bar{\mathbf{y}}\ }$

0 and  $\mathbf{v}_i, i = 1, 2, \dots, k$ , and a new input  $\mathbf{S}_x$  in form of kernel values, the spectral embedding of  $\mathbf{S}_x$  for Step 1(a) is  $\mathbf{y} = \Lambda_k^{-1} \mathbf{V}^T \tau(\mathbf{S}_x)$ , and Step 1(b) is  $\mathbf{y} = \Lambda_k^{-1/2} \mathbf{V}^T \tau(\mathbf{S}_x)$ ; where the diagonal matrix  $\Lambda_k = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_k, 0, \dots, 0)$ , and  $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k, \mathbf{v}_{k+1}, \dots, \mathbf{v}_n)^T$ .

*Proof.* By eigendecomposition,  $\Gamma(\mathbf{S}) = \mathbf{V} \Lambda \mathbf{V}^T$ , where  $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$ . Therefore,  $\Gamma(\mathbf{S})$  can be approximated by  $\Gamma(\mathbf{S}) \approx \mathbf{V} \Lambda_k \mathbf{V}^T = (\Lambda_k^{1/2} \mathbf{V}^T)^T (\Lambda_k^{1/2} \mathbf{V}^T)$ , since the interested  $k$  eigenvalues are the largest positive eigenvalues of  $\Gamma(\mathbf{S})$ . Therefore, each training example  $i$  can be represented by the spectral embedding  $\mathbf{y}_i = \Lambda_k^{1/2} (\mathbf{v}_1^i, \mathbf{v}_2^i, \dots, \mathbf{v}_n^i)^T$  in terms of matrix approximation, where  $\mathbf{y}_i$  is the spectral embedding of the  $i$ -th training example by Step 1(b) from Table 1. If we assume that  $\mathbf{y}$  is also the spectral embedding of the new input in terms of matrix approximation, then  $\mathbf{y}$  should be the spectral embedding obtained by Step 1(b). By matrix approximation, we can therefore approximate the transformed kernel vector  $\tau(\mathbf{S}_x)$  of the new input  $\mathbf{S}_x$  in the same way. Thus, we have  $\tau(\mathbf{S}_x) = (\Lambda_k^{1/2} \mathbf{V}^T)^T \mathbf{y}$ . Since  $\Lambda_k^{1/2} \mathbf{V}^T$  is an orthogonal matrix, it can be solved by taking the matrix inverse to obtain  $\mathbf{y} = \Lambda_k^{-1/2} \mathbf{V}^T \tau(\mathbf{S}_x)$ . This gives the result in Step 1(b). Further, the spectral embedding by Step 1(a) can be computed by multiplying the diagonal matrix  $\Lambda_k^{-1/2}$ , which gives the spectral embedding of the new input  $\mathbf{y} = \Lambda_k^{-1/2} (\Lambda_k^{1/2} \mathbf{V}^T \tau(\mathbf{S}_x)) = \Lambda_k^{-1} \mathbf{V}^T \tau(\mathbf{S}_x)$  by Step 1(a).

Essentially, Lemma 3 specifies how to project the new input onto the spectral embedding space given by the training examples. And Step 2 of Figure 1 served as the optional step that can be applied to the spectral embedding of the new input (computed either by Step 1(a) or 1(b)), and its used is dependent on whether Step 2 was used during training so that the new input can be compared with the training examples within the same embedding space.

### 3.2 Computing the Spectral Kernel

When the spectral embedding of the training and testing set is ready, the final step is to compute the spectral kernel values from the spectral embedding using a selected similarity measure. This step is flexible and many typical similarity measures can be used. Table 2 lists some possible options for the spectral kernel. Notice that the magnitude of the similarity measure need not be constrained within 0 and 1 since there is no strict requirement on the range of possible kernel values in classification.

### 3.3 Relationship to Latent Semantic Kernel

We conclude this section with the proof that the latent semantic kernel is only a specific instance of the spectral kernel. The objective is two-fold: (i) we want to clarify the difference between spectral kernels and latent semantic kernels as they appear similar at first glance; and (ii) by this proof, we seek to establish spectral kernels as a framework under which spectral clustering information can be further researched to improve the task of classification.

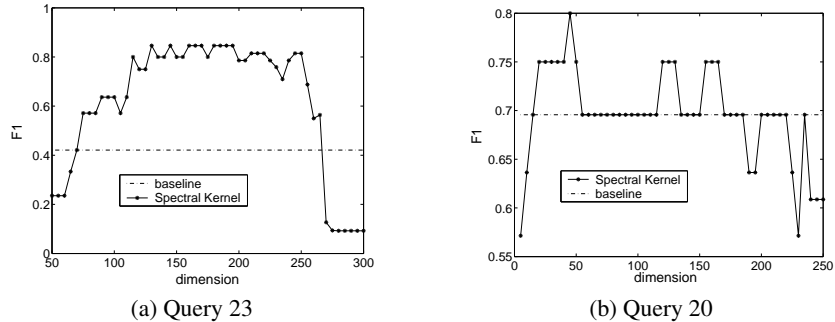
**Theorem 1.** *Given the term-document matrix  $\mathbf{D}$ , the latent semantic kernel is a specific instance of the spectral kernel, i.e.,  $SK\langle\Gamma_I(\mathbf{D}^T\mathbf{D}), 1(b), S_I\rangle$  reduces to the latent semantic kernel.*

*Proof.* From [12], we have the following facts: latent semantic kernels are computed from the term-document matrix  $\mathbf{D}$  or  $\mathbf{S} = \mathbf{D}^T\mathbf{D}$ ; the LSK matrix of training set is  $\mathbf{K} = \mathbf{V}\mathbf{\Lambda}_k\mathbf{V}^T$ ; the LSK values between the training set  $\mathbf{d}_i$  and the new input  $\mathbf{d}$  is  $\kappa(\mathbf{d}_i, \mathbf{d}) = (\mathbf{V}\mathbf{\Lambda}_k\mathbf{V}^T\mathbf{t})_i$ , where  $\mathbf{t} = \mathbf{D}^T\mathbf{d}$  and the matrix  $\mathbf{V}$  is obtained from eigen decomposition  $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$ . Using the above, we prove that spectral kernels with a configuration of  $SK\langle\Gamma_I(\mathbf{S}), 1(b), S_I\rangle$  gives the same  $\mathbf{K}$  and  $\kappa(\mathbf{d}_i, \mathbf{d})$  as LSK. We denote  $\hat{\mathbf{K}}$  as the SK matrix of the training set and  $\hat{\kappa}(\mathbf{d}_i, \mathbf{d})$  as the SK values between the training set  $\mathbf{d}_i$  and the new input  $\mathbf{d}$ . Since the input kernel matrix of SK is  $\mathbf{S} = \mathbf{D}^T\mathbf{D}$  and is computed by the inner product of each document  $\mathbf{d}_i$ , we can easily get the input kernel values of the new input  $\mathbf{d}$  as  $\mathbf{S}_x = \mathbf{D}^T\mathbf{d}$ . As the transformation is  $\Gamma_I$ , we have  $\Gamma_I(\mathbf{S}) = \mathbf{S}$  and  $\tau_I(\mathbf{S}_x) = \mathbf{S}_x = \mathbf{D}^T\mathbf{d} = \mathbf{t}$ . Then, we compute the spectral embedding (using Step 1(b)) of  $\mathbf{d}_i$  as  $\mathbf{y}_i = \mathbf{\Lambda}_k^{1/2}(\mathbf{v}_1^i, \mathbf{v}_2^i, \dots, \mathbf{v}_n^i)^T$ , where  $\mathbf{v}_i$  is the  $i$ -th eigenvector of  $\mathbf{S}$  or the  $i$ -th column of  $\mathbf{V}$ . Further, the spectral embedding of the new input  $\mathbf{d}$  (by Step 1(b)) is  $\mathbf{y} = \mathbf{\Lambda}_k^{-1/2}\mathbf{V}^T\tau(\mathbf{S}_x) = \mathbf{\Lambda}_k^{-1/2}\mathbf{V}^T\mathbf{t}$ . Since the final component is the inner product  $S_I$ , we immediately get  $\hat{\mathbf{K}} = (\mathbf{y}_i^T\mathbf{y}_j)_{n\times n} = (\mathbf{\Lambda}_k^{1/2}\mathbf{V}^T)^T(\mathbf{\Lambda}_k^{1/2}\mathbf{V}^T) = \mathbf{V}\mathbf{\Lambda}_k\mathbf{V}^T = \mathbf{K}$  and  $\hat{\kappa}(\mathbf{y}_i, \mathbf{y}) = \mathbf{y}_i^T\mathbf{y} = \left((\mathbf{\Lambda}_k^{1/2}\mathbf{V}^T)^T(\mathbf{\Lambda}_k^{-1/2}\mathbf{V}^T\mathbf{t})\right)_i = (\mathbf{V}\mathbf{\Lambda}_k\mathbf{V}^T\mathbf{t})_i = \kappa(\mathbf{y}_i, \mathbf{y})$ .

## 4 Experimental Results

We evaluated our spectral kernels on two text data sets, namely Medline1033 and Reuters-21578, to demonstrate its applicability and effectiveness. We chose these two datasets for easy comparison with the experiments reported in [12]. Compared with the baseline method, i.e., SVM classifier with linear kernel without feature selection, our results were either much better or positively on-par, than that of baseline method according to  $F_1$  measure. Further details can be obtained from our technical report [20].

**Medline1033** This data set contains 1,033 documents and 30 queries obtained from the National Library of Medicine. Following the experimental setting reported in [12], we focused on `query23` and `query20`. Each query contains 39 relevant documents from which we selected 90% of them (i.e., 24 documents) as the training set and the remaining 10% (15 documents) as the testing set. We performed 50 random splits of this data set in our experiments and reported the average performance.



**Fig. 2.** Generalization performance of the SVM classifier with the proposed spectral kernel  $\text{SK}\langle\Gamma_N, 1(a), S_I\rangle$  and the linear kernel for the Medline1033 data set.

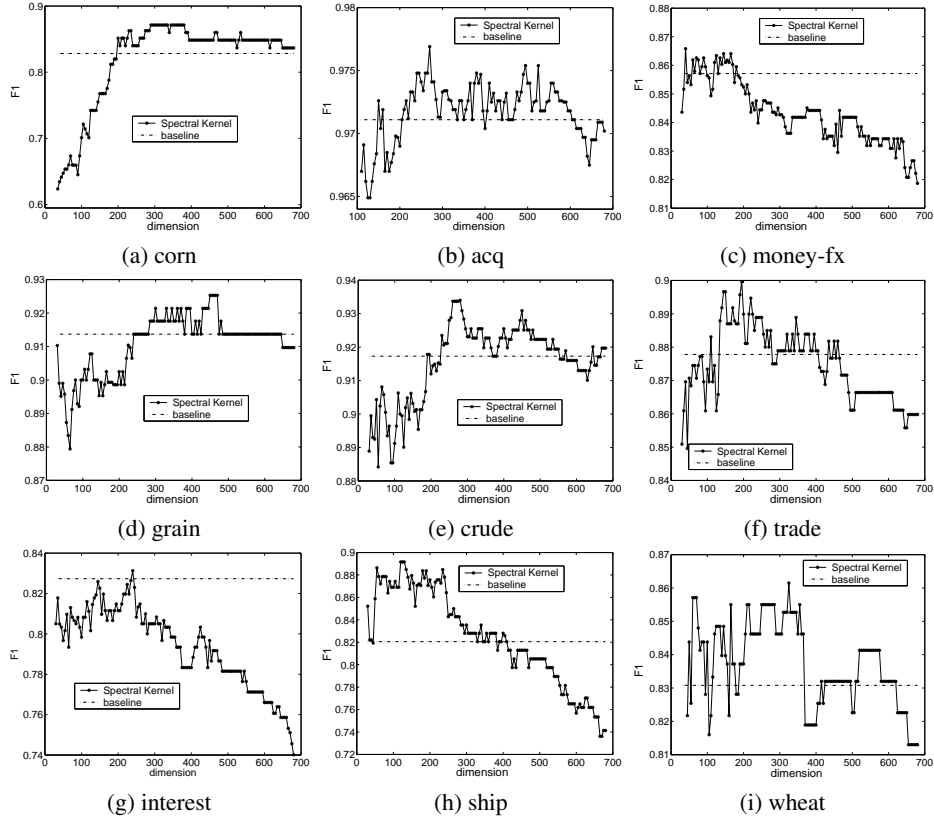
**Reuters21578** This data set contains 21,578 news articles organized in 135 categories and has been widely used in text classification [21]. Each document is labelled with zero, one, or more categories. We used the `ModApte` split to obtain the training and testing set, and conducted our experiments on the top ten largest categories. They are `earn`, `acq`, `money-fx`, `grain`, `crude`, `trade`, `interest`, `ship`, `wheat`, and `corn`. The number of training documents and testing documents are 6,494 and 2,548 respectively.

In our experiments, each document was represented as a feature vector with term frequency (TF) weighting and all document vectors were normalized to unit length. Terms were stemmed after stopword removal. We used *SVM<sup>light</sup>* with default parameter setting for spectral embedding and baseline in all experiments. As a binary classifier, one SVM classifier was trained for each category. The SVM classifier with linear kernel was used as the baseline method where no feature selection was performed. The classification performance was evaluated with  $F_1$  for each category and *Micro*- and *Macro*-averaged  $F_1$  on all the categories [22].

#### 4.1 Results on Medline1033

We configured a spectral kernel of the form  $\text{SK}\langle\Gamma_N, 1(a), S_I\rangle$  and compared its performance against the baseline method. The results are reported in Figure 2. We started with a small  $k$  feature space for the classifier with spectral kernel and increased the dimensionality until the classification performance deteriorated, i.e., when  $k > 250$ . The results of both `query23` and `query20` proved to be very encouraging.

With a small  $k$  (less than 200), the spectral kernel  $\text{SK}\langle\Gamma_N, 1(a), S_I\rangle$  increased quickly to a result that was much better than baseline method according to  $F_1$  measure. In particular, for `query23`, the best performance delivered by the spectral kernel was 84.62%, almost twice that of the baseline method which was 42.11%.



**Fig. 3.**  $F_1$  values of SVM classifier with a spectral kernel configured as  $SK(\Gamma_N, 1(b)2, S_I)$  versus the SVM classifier with a linear kernel on Reuters21578 dataset. *Note:* due to space constraints, we only showed nine of the ten categories. The complete set of graphs can be obtained from our technical report [20].

## 4.2 Results on Reuters21578

On the Reuters data set, we configured the spectral kernel to  $SK(\Gamma_N, 1(b)2, S_I)$  giving the results shown in Figure 3. While performing comparably on categories *earn* and *interest*, the spectral kernel outperformed the linear kernel (i.e., baseline method) on the remaining eight categories with small  $k$  values (generally less than 300). In particular, on the *ship* category, the best  $F_1$  achieved by spectral kernel was 89.16% which is much higher than 82.05% delivered by linear kernel. More importantly, the  $F_1$  performance under most values of  $k$  were much higher than the baseline as shown in Figure 3(h). This is an encouraging result showing the effectiveness of spectral kernels in text classification tasks.

Furthermore, eight of the performance plots on Reuters data set, and two of the performance plots on Medline data set showed that a small value of  $k$  (usually  $100 \leq k \leq 300$ ) is often sufficient to achieve good  $F_1$  performance. This observation is different

**Table 3.**  $F_1$  using spectral kernel  $\text{SK}\langle\Gamma_N, 1(b)2, S_I\rangle$  and linear kernel on the ten Reuters categories with the best F1 results in bold.

category	k			baseline
	120	270	420	
earn	0.987	0.986	0.988	<b>0.990</b>
acq	0.966	<b>0.977</b>	0.973	0.971
money-fx	<b>0.861</b>	0.847	0.834	0.857
grain	0.908	0.914	<b>0.918</b>	0.914
crude	0.900	<b>0.934</b>	0.922	0.917
trade	0.875	<b>0.880</b>	0.873	0.878
interest	0.811	0.805	0.798	<b>0.827</b>
ship	<b>0.892</b>	0.843	0.813	0.821
wheat	0.846	<b>0.855</b>	0.825	0.831
corn	0.701	<b>0.852</b>	0.849	0.828
micro-avg	0.939	<b>0.945</b>	0.942	0.944
macro-avg	0.875	<b>0.889</b>	0.879	0.883

from the selection of  $k$  in latent semantic kernels, where a larger  $k$  implied better performance (i.e., in the range of 500 to 1000 as observed in [12]). The small values of  $k$  is encouraging because they lead to shorter runtime to achieve the same classification accuracy as LSK.

Nevertheless, the experimental results on Reuters data set did reveal a shortcoming of the spectral kernel: there is not a fix value of  $k$  that ensures consistent performance for different categories. This will be a practical limit that requires automated mechanisms to determine  $k$ . While we are working on this as part of our future work, using a single value of  $k$  enables us to compare the spectral kernel against the linear kernel objectively. As Table 3 shows, we achieved better performance than the baseline method when  $k = 270$ , and comparable performance when  $k = 120$ .

### 4.3 Computational Complexity

We end this section by discussing the computational complexity of the spectral kernel. In text collections, the data is often sparse, i.e., the number of non-zero entries  $h$  in  $\mathbf{S}$  is less than the number of zero entries. If the symmetric matrix  $\mathbf{S}$  has  $n$  rows and columns, then the complexity of transformation from  $\mathbf{S}$  to  $\Gamma(\mathbf{S})$  would be  $O(h+n)$ . The study of how to effectively compute the largest eigenvalues has been well-developed within the domain of symmetric generalized eigen problems that arised from structural analysis in physics and computational chemistry. Thus, a series of mature mathematical tools are available for our purpose.

In the case where the symmetric matrix is sparse, i.e.,  $h$  and  $n^2$  do not have the same magnitude, we can use the Lanczos method to compute the eigenvalues in  $k \ll n$  iterations allowing the process to converge quickly [23]. Since the complexity of each iteration is  $O(h+n)$ , the final complexity of this method is therefore  $O(k(h+n))$ . If we ignore the very small  $k$  in real computations, the complexity of our method becomes  $O(h+n)$ . In our experiments, the ARPACK package is used. It is a collection of

Fortran77 subroutines that effectively computes the  $k$  eigenvalues and eigenvectors using only  $2nk + O(k^2)$  storage space (with no auxiliary storage [24] required). Since our matrices are symmetric, the method used in ARPACK actually reduces to a variant of the Lanczos process known as Implicitly Restarted Lanczos Method (IRLM).

## 5 Conclusion and Future Work

In this paper, we proposed a new kernel that uses the semantics extracted from spectral clustering. Unlike the latent semantic kernels, spectral kernels are unique not only by the virtue of using spectral clustering information, but also its ability to support incremental updates to the kernel matrix that keeps the cost of training to a minimal. Furthermore, we have shown that we can obtain the spectral embedding of both training and testing sets by matrix approximation. Hence, it is possible for spectral kernels to handle feature space of any dimensionality.

Our experiments on text data proves the feasibility of spectral kernels in terms of accuracy and efficiency. The results from our experiments are promising. In most cases, the spectral kernel achieved substantial improvement in performance without any lost of accuracy. We have additional results from experiments of other data sets but for space reasons, we refer the reader to our technical report [20] instead.

The closest piece of related work, to our knowledge, is the classification of projected  $k$ -dimensional space obtained by spectral clustering algorithms [25]. However, the proposal by Kamvar *et al.* suffers from three drawbacks. First, if a new input arrives, there is a need to eigendecompose the new  $\mathbf{S}$  (which includes the new input) to obtain the new spectral space. This is time-consuming for classification during operation where large number of data may arrive. Second, since the spectral space is dependent on the testing set rather than the training set, the spectral space is unstable if the testing set is highly random. Third, the similarity relationship between training data is not fully exploited to significantly improve the accuracy of classification. Our proposal overcomes these drawbacks and provided a kernel framework of applying spectral clustering to classification.

As an attempt to develop a novel kernel for classification, we foresee much future work for research. In particular, our immediate interest is to be able to find a suitable value of  $k$  for each category in classification by means of automated mechanisms. This is important for practical reasons as maintaining the right value of  $k$  over the lifetime of classification can significantly improve classification accuracy.

## References

1. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines. Cambridge University Press (2000)
2. Scholkopf, B., Smola, A., Muller, K.R.: Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation* **10** (1998) 1299–1319
3. Bach, F.R., Jordan, M.I.: Kernel independent component analysis. *The Journal of Machine Learning Research* **3** (2003) 1–48
4. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press (2004)

5. Li, W., Ng, W.K., Ong, K.L., Lim, E.P.: A spectroscopy of texts for effective clustering. In: Proc. 8th PKDD, Pisa, Italy (2004)
6. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. *J. American Society of Information Science* **41** (1990) 391–407
7. Kleinberg, J.M.: Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM* **46** (1999) 604–632
8. Lawrence, P., Sergey, B., Rajeev, M., Terry, W.: The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project (1999)
9. Siolas, G., d’Alché Buc, F.: Support vector machines based on a semantic kernel for text categorization. In: Proc. Int. Joint Conf. on Neural Networks (IJCNN). (2000) 5205–5210
10. Moschitti, A., Bejan, C.A.: A semantic kernel for predicate argument classification. In: Proc. Natural Language Learning (CoNLL), Boston, MA, USA (2004) 17–24
11. Gosselin, P.H., Cord, M.: Semantic kernel updating for content-based image retrieval. In: Proc. of IEEE 6th International Symposium on Multimedia Software Engineering (ISMSE), Miami, Florida (2004) 537–544
12. Cristianini, N., Shawe-Taylor, J., Lodhi, H.: Latent semantic kernels. In: In Proc. of 18th International Conference on Machine Learning (ICML), Williams College, US (2001) 66–73
13. Chung, F.R.K.: Spectral Graph Theory. Number 92 in CBMS Regional Conference Series in Mathematics. American Mathematical Society (1997)
14. Ding, C.: Tutorial: Spectral clustering. In: Proc. of 21st International Conference on Machine Learning (ICML), Alberta, Canada (2004)
15. Shi, J., Malik, J.: Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22** (2000) 888–905
16. Maila, M., Shi, J.: A random walks view of spectral segmentation. In: Proc. of the 8th International Workshop on Artificial Intelligence and Statistics, Florida (2001)
17. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: Proc. of Advances in Neural Information Processing Systems 14. (2001)
18. Papadimitriou, C.H., Tamaki, H., Raghavan, P., Vempala, S.: Latent semantic indexing: A probabilistic analysis. In: Proc. ACM PODS, Seattle, Washington, USA (1998) 159–168
19. Golub, G., Reinsch, C.: Handbook for Matrix Computation II, Linear Algebra. Springer-Verlag, New York (1971)
20. Li, W., Ong, K.L., Sun, A., Ng, W.K.: Spectral kernels for classification. Technical Report TRC-5/05 (<http://www.deakin.edu.au/~leong/tr>), Deakin University (2005)
21. Lewis, D.D.: Reuters-21578 text categorization test collection. (<http://www.daviddlewis.com/resources/testcollections/reuters21578/>)
22. Yang, Y.: An evaluation of statistical approaches to text categorization. Technical Report CMU-CS-97-127, Carnegie Mellon University (1997)
23. Golub, G., Loan, C.V.: Matrix Computations (Johns Hopkins Series in the Mathematical Sciences). 3rd edn. The Johns Hopkins University Press (1996)
24. Lehoucq, R.B., Sorensen, D.C., Yang, C.: ARPACK User’s Guide: Solution of Large-Scale Eigenvalue Problems by Implicitly Restarted Arnoldi Methods. SIAM (1998)
25. Kamvar, S.D., Klein, D., Manning, C.D.: Spectral learning. In: Proc. of the 18th International Joint Conference on Artificial Intelligence (IJCAI). (2003)