Graph Classification Based on Optimizing Graph Spectra

Nguyen Duy Vinh[1], Akihiro Inokuchi[1,2], and Takashi Washio[1]

[1] The Institute of Scientific and Industrial Research, Osaka University
[2] PRESTO, Japan Science and Technology Agency
{inokuchi,washio}@ar.sanken.osaka-u.ac.jp

Abstract. Kernel methods such as the SVM are becoming increasingly popular due to their high performance in graph classification. In this paper, we propose a novel graph kernel, called SPEC, based on graph spectra and the Interlace Theorem, as well as an algorithm, called OPT-SPEC, to optimize the SPEC kernel used in an SVM for graph classification. The fundamental performance of the method is evaluated using artificial datasets, and its practicality confirmed through experiments using a real-world dataset.

Keywords: Graph Kernel, Interlace Theorem, Graph Spectra.

1 Introduction

A natural way of representing structured data is to use graphs. As an example, the structural formula of a chemical compound is a graph where each vertex corresponds to an atom in the compound, and each edge corresponds to a bond between two atoms therein. By using such graph representations, a new research field has emerged from data mining, namely graph mining, with the objective of mining information from a database consisting of graphs. With the potential to find meaningful information, graph mining has raised great interest, and research in the field has increased rapidly in recent years. Furthermore, since the need for classifying graphs has increased in many real-world applications, *e.g.,* analysis of proteins in bioinformatics and chemical compounds in cheminformatics [11], graph classification has also been widely researched worldwide. The main objective of graph classification is to classify graphs of similar structures into the same classes. This originates from the fact that instances represented by graphs usually have similar properties if their graph representations have high structural similarity.

Kernel methods such as the SVM are becoming increasingly popular due to their high performance in graph classification [10]. Most graph kernels are based on the idea of an object decomposed into substructures and a feature vector

B. Pfahringer, G. Holmes, and A. Hoffmann (Eds.): DS 2010, LNAI 6332, pp. 205–220, 2010.

containing counts of the substructures. Since the dimensionality of feature vectors is typically very high and includes the subgraph isomorphism matching problem that is known to be NP-complete [4], kernels deliberately avoid explicit computations of feature values and employ efficient procedures.

One of the representative graph kernels is the Random Walk Kernel [12,10], which computes $k(g_i, g_j)$ in $O(|g|^3)$ for graphs g_i and g_j, where $|g|$ is the number of vertices in g_i and g_j. The kernel returns a high value if the random walk on the graph generates many sequences with the same labels for vertices and edges, *i.e.,* the graphs are similar to each other. The Neighborhood Hash Kernel (NHK) [6] is another recently proposed kernel that computes $k(g_i, g_j)$ in $O(|g|d)$ for g_i and g_j, where d is the average degree of the vertices. The NHK uses logical operations such as the exclusive-OR on the label set of the connected vertices. The updated labels given by repeating the hash, propagate the label information over the graph and uniquely represent the high order structures around the vertices beyond the vertex or edge level. An SVM with two graph kernels works very well with benchmark data consisting of graphs with common small subgraphs (consisting of 1-6 vertices).

In many real-world applications using graph structured data, large subgraphs have a greater significance than small ones, because they contain more useful structural information. However, existing algorithms employing graph kernels, including the Random Walk Kernel and Neighborhood Hash Kernel, do not achieve good performance when classifying graphs whose classes depend on whether the graphs contain some large common subgraphs. The main reason for this is that the core principle of kernel algorithms is the use of a very small number of neighbor vertices when characterizing each vertex. Our experiments show that these two graph kernels do not work well with this kind of graph, and thus the application thereof is limited.

Based on the background, in this paper we aim to solve a classification problem of graphs where the binary class of each graph is defined by whether it contains some graphs in a large graph set as induced subgraphs. For this purpose, we propose a new graph kernel, called SPEC, and an algorithm, referred to as OPTSPEC (*OPT*imizing graph *SPEC*tra for graph classification) which optimizes the classification performance of SPEC when included in an SVM. The key feature of our algorithm is the use of graph spectra and the Interlace Theorem [7,8] for constructing the SPEC graph kernel. The graph spectrum of a graph is a vector consisting of eigenvalues of a matrix representing the graph, while the Interlace Theorem gives conditions of the largest common subgraph of two graphs by comparing their graph spectra. This theorem provides a very sensitive measure to identify graphs with large common subgraphs. Thus, an SVM using the SPEC kernel, which takes full advantage of this theorem, can efficiently classify graphs based on whether or not they contain some graphs in a large graph set as induced subgraphs. The cost of calculating graph spectra for a graph composed of $|g|$ vertices is $O(|g|^3)$, and therefore, our kernel requires only $O(|g|^3)$ computation time to compute $k(g_i, g_j)$ for graphs g_i and g_j.

2 Problem Definition

A labeled graph g is represented as $g = (V, E, L, l)$, where $V = \{v_1, \cdots, v_z\}$ is a set of vertices, $E = \{(v, v') \mid (v, v') \in V \times V\}$ is a set of edges[1], and L is a set of labels such that $l : V \cup E \rightarrow L$. If an edge exists between two vertices, the vertices are said to be adjacent. The number of vertices in a graph g is referred to as the size of the graph, and is denoted as $|g|$. Given two graphs $g = (V, E, L, l)$ and $g' = (V', E', L', l')$, g' is called a subgraph of g, if there exists an injective function $\phi : V' \rightarrow V$ that satisfies the following three conditions for $\forall v, v_1, v_2 \in V'$.

1. $(\phi(v_1), \phi(v_2)) \in E$, if $(v_1, v_2) \in E'$,
2. $l'(v) = l(\phi(v))$,
3. $l'((v_1, v_2)) = l((\phi(v_1), \phi(v_2)))$.

In addition, a subgraph g' of g is an "induced subgraph", where $\phi(v_1)$ and $\phi(v_2)$ are adjacent in g, if and only if v_1 and v_2 in $V(g')$ are adjacent in g'.

Classification problem of graphs is defined as follows. Given training examples $\{(g_i, y_i)\}$ $(i = 1, \cdots, n)$, where each example is a pair of a labeled graph g_i and class $y_i \in \{+1, -1\}$ that the graph belongs to, the objective of the learning machine is to learn a function f to predict the classes of test examples correctly. As mentioned in Section 1, in this paper, we focus on the classification of graphs whose class labels are determined by whether or not they contain some large induced subgraphs belonging to a given graph set S. In concrete terms, the class label 1 is assigned to graphs containing some graph in S as an induced subgraph, while the class label -1 is assigned to the other graphs[2].

3 Graph Kernel for Large Graph Classification Problems

In this section, we present our method for constructing the SPEC graph kernel to measure similarity between two graphs efficiently. Based on graph spectra and the Interlace Theorem, the SPEC kernel is very sensitive when working with graphs containing large common subgraphs. As a result of this sensitivity, an SVM employing the SPEC kernel is expected to have high accuracy in the classification of graphs.

3.1 Matrix Representation of Graphs

Matrices are a very useful representation of graphs, because we can obtain useful information about the topological structure of a graph from its matrix representation. In fact, there are various kinds of matrices that can be used to represent

[1] Although this paper focuses on undirected graphs only, we discuss in Section 6 that the proposed method is also applicable to directed graphs without loss of generality.

[2] In Section 6, we also discuss the classification of graphs whose class labels are determined by whether or not they contain some large common subgraphs as "general" subgraphs, but not "induced" subgraphs.

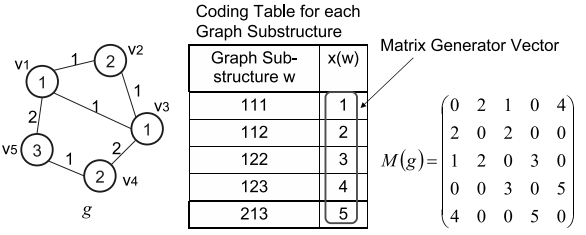


Fig. 1. Coding table and adjacency matrix

a graph, each of which captures different features of the graph. The most basic matrix representation of a graph is the *adjacency matrix*. In this paper, for a graph $g = (V, E, L, l)$ containing $|g|$ vertices, its adjacency matrix representation $M(g)$ is a $|g| \times |g|$ matrix whose elements are given by

$$M(g)_{i,j} = \begin{cases} c(w(i,j)) & \texttt{if } \{v_i, v_j\} \in E, 1 \leq i, j \leq |g|, \\ 0 & \texttt{otherwise,} \end{cases} \quad (1)$$

where $M(g)_{i,j}$ is the (i,j)-th element of $M(g)$ and $w(i,j)$, called the *graph substructure*, is a combination of labels $[l(v_i)\ l(\{v_i, v_j\})\ l(v_j)]$ expressing the structure between two vertices v_i and v_j, which corresponds to an element the adjacency matrix. Moreover, $c(w(i,j))$ is a code characterizing the graph substructure $w(i,j)$ and is represented by a real number. All graph substructures and their corresponding codes are collated into a table called the *Coding table*. Furthermore, we call the vector consisting of all the codes the *matrix generator vector* $\boldsymbol{x}$. Because we assume that the graph g is an undirected graph, the adjacency matrix of g is symmetric. Moreover, the eigenvalues of $M(g)$ are real.

For example, the adjacency matrix $M(g)$ of the graph g in Fig. 1 is created based on the Coding table and the matrix generator vector $\boldsymbol{x} = (1, 2, 3, 4, 5)^T$. Let us consider the two vertices v_3 with label 1 and v_4 with label 2 in g in our discussion of the construction of $M(g)$. Because there is an edge with label 2 connecting these two vertices, the graph substructure corresponding to these two vertices is $w(3,4) = [1\ 2\ 2]$. Therefore, since the Coding table assigns the value 3 to the graph substructure $[1\ 2\ 2]$, the two elements $M(g)_{3,4}$ and $M(g)_{4,3}$ of $M(g)$ have the value 3. On the other hand, since there is no edge connecting the two vertices v_1 and v_4 in g, the value 0 is assigned to $M(g)_{1,4}$ and $M(g)_{4,1}$.

3.2 Graph Spectrum and Interlace Theorem

We can calculate the *graph spectrum* of a matrix representing a graph, such as an adjacency matrix. The graph spectrum of a graph is a vector consisting of ordered eigenvalues of the matrix representing the graph. Due to the nature of matrices, a graph spectrum is known to be one of the graph invariants. This arises from the fact that the eigenvalues of a matrix remain constant when its rows and columns of the matrix are exchanged. Using graph spectra, we can compute ranges for the eigenvalues of matrices of common induced subgraphs that may exist in two arbitrary graphs by the following theorem.

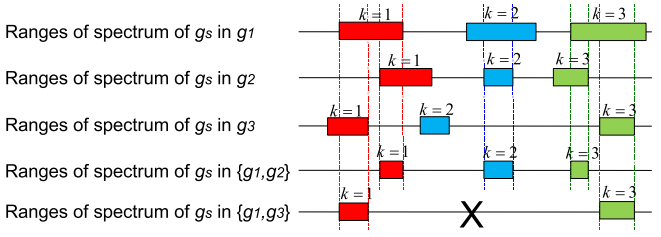


Fig. 2. Example of calculating the ranges of a common subgraph spectrum

Theorem 1 (Cauchy's Interlace Theorem [7,8]). *Let* $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_p$ *be eigenvalues of a symmetric matrix* A *of size* $p \times p$*, and let* $\mu_1 \leq \mu_2 \leq ... \leq \mu_m$ *be eigenvalues of a symmetric matrix* B *of size* $m \times m$ $(m < p)$*. If* B *is a principal submatrix of* A*, then*

$$\lambda_k \leq \mu_k \leq \lambda_{k+p-m}, \qquad k = 1, ..., m. \tag{2}$$

■

If g_s of size m is an induced subgraph of g of size p, $M(g_s)$ is a principal submatrix of $M(g)$. Therefore, if g_s is an induced subgraph of g, Eq. (2) holds for eigenvalues of the matrices $M(g)$ and $M(g_s)$. In the remainder of this paper, we simply denote the i-th eigenvalue of the graph spectrum of a graph g as $\gamma_i(g)$ $(1 \leq i \leq |g|)$, and the graph spectrum of g as $\gamma(g) = \{\gamma_1(g), \cdots, \gamma_{|g|}(g)\}$.

By utilizing the Interlace Theorem, we can either compute the range of the k-th eigenvalue of the spectrum of common induced subgraphs contained in two given graphs or decide that the graphs do not contain a common induced subgraph of size m. This is illustrated in Fig. 2. Each hatched rectangle in the first, second, and third parts of Fig. 2 represents the range that eigenvalues of a spectrum of an induced subgraph with 3 vertices in g_1, g_2, and g_3, respectively, can take. The ranges of eigenvalues which the graph spectrum of a common induced subgraph g_s of g_1 and g_2 can take are limited to the intersections of the corresponding ranges given by g_1 and g_2, as shown by rectangles in the fourth part of Fig. 2. On the other hand, g_1 and g_3 cannot contain g_s with 3 vertices as a common induced subgraph, because the ranges given by g_1 and g_3 do not have any intersection for the second eigenvalue of the graph spectrum of g_s.

Interlace Theorem is very sensitive to large induced subgraphs of a graph. Let the graph spectrum of a graph g be $\gamma(g) = \{\gamma_1(g), \cdots, \gamma_{|g|}(g)\}$, and an induced subgraph of size m in g be g_s^m. A range that the k-th eigenvalue of the graph spectrum of g_s^m can take is $[\gamma_k(g), \gamma_{k+|g|-m}(g)]$. Thus, width of a range that the k-th eigenvalue of the graph spectrum of $g_s^{m'}$ $(m < m')$ can take is no more than that of g_s^m, because $k + |g| - m' < k + |g| - m$ and $\gamma_{k+|g|-m'}(g) \leq \gamma_{k+|g|-m}(g)$. Therefore, the theorem is very sensitive to large induced subgraphs of a graph.

Given two arbitrary graphs g_i and g_j, one of the problems we intend to solve in this paper is the construction of a graph kernel $k(g_i, g_j)$ that can measure the similarity between the two graphs efficiently, especially when they contain large common induced subgraphs. For this purpose, we employ graph spectra and the

Interlace Theorem described above, since a good kernel is the key to the success of an SVM in the classification of graphs.

3.3 Graph Kernel for Large Graph Classification

First, we describe the kernel function, the kernel matrix characterized by a kernel, and their requirements. Given two graphs g_i and g_j, let $k(g_i, g_j)$ denote a kernel function between graphs g_i and g_j. If g_i and g_j have high similarity, $k(g_i, g_j)$ should be large. Given a graph dataset $\{(g_i, y_i)\}$ $(1 \leq i \leq n)$ and a kernel function $k(\cdot, \cdot)$, we compute each element of a kernel matrix K as

$$K_{i,j} = K_{j,i} = k(g_i, g_j) \quad (1 \leq i, j \leq n).$$

To be applicable to an SVM, the graph kernel must be a PSD (positive semi-definite) kernel [13]. Let k_1 and k_2 be arbitrary PSD kernel functions, $\boldsymbol{x}$ and $\boldsymbol{y}$ be arbitrary vectors, and A be a $|\boldsymbol{x}| \times |\boldsymbol{y}|$ matrix where $|\boldsymbol{x}|$ is the dimensionality of the vector $\boldsymbol{x}$. Moreover, let $\boldsymbol{x} = (\boldsymbol{x}_a^T, \boldsymbol{x}_b^T)^T$, where $\boldsymbol{x}$ is a concatenation of $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$. $k(\boldsymbol{x}, \boldsymbol{y})$ is another PSD kernel if one of the following holds:

$$k(\boldsymbol{x}, \boldsymbol{y}) = \exp(k_1(\boldsymbol{x}, \boldsymbol{y})), \tag{3}$$

$$k(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^T A \boldsymbol{y}, \tag{4}$$

$$k(\boldsymbol{x}, \boldsymbol{y}) = k_1(\boldsymbol{x}_a, \boldsymbol{y}_a) + k_2(\boldsymbol{x}_b, \boldsymbol{y}_b), \quad \texttt{or} \tag{5}$$

$$k(\boldsymbol{x}, \boldsymbol{y}) = k_1(\boldsymbol{x}_a, \boldsymbol{y}_a) \times k_2(\boldsymbol{x}_b, \boldsymbol{y}_b). \tag{6}$$

To construct our graph kernel, we consider two graphs g_i and g_j whose graph spectra are given by

$$\gamma(g_i) = \{\gamma_1(g_i), \cdots, \gamma_{|g_i|}(g_i)\} \ \texttt{and} \ \gamma(g_j) = \{\gamma_1(g_j), \cdots, \gamma_{|g_j|}(g_j)\},$$

respectively. By the Interlace Theorem, if a common induced subgraph g_s of size m is contained in the graphs g_i and g_j, the range that $\gamma_k(g_s)$ $(1 \leq k \leq m \leq min(|g_i|, |g_j|))$ can take is the intersection of

$$[\gamma_k(g_i), \gamma_{k+|g_i|-m}(g_i)] \ \texttt{and} \ [\gamma_k(g_j), \gamma_{k+|g_j|-m}(g_j)]. \tag{7}$$

Consider a matrix $\frac{1}{c_{max}} M(g_s)$ of g_s where c_{max} is the maximum absolute value among elements in $M(g_s)$. Since it is a random symmetric matrix with elements of absolute value at most 1, the probability that the k-th eigenvalue of $\frac{1}{c_{max}} M(g_s)$ deviates from its median by more than t is at most $4e^{-t^2/32k^2}$, where $1 \leq k \leq m$ [1]. On the other hand, if the eigenvalues are uniformly spaced, the average interval $\overline{d}$ between two of the eigenvalues is at most $\overline{d} = \frac{2(m-1)}{m-1} = 2$, since m eigenvalues of $\frac{1}{c_{max}} M(g_s)$ whose diagonal elements are 0 as defined by Eq. (1) must exist in $[-(m-1), (m-1)]$. Comparing the interval $\overline{d}$ with the standard deviation $4k$ of the distribution of the eigenvalues [1], the standard deviation is large enough for all k. Since the eigenvalues do not extremely concentrate around the median and are widely distributed, there is a strong possibility that $\gamma_k(g_s)$

exists in the intersection of the rages (7) when the width of the intersection is large. Furthermore, if the possibility that $\gamma_k(g_s)$ exists in the intersection increases for all k, the possibility that g_s is included as an induced subgraph in g_i and g_j also increases according to the Interlace Theorem. Therefore, we define the *SPEC* (graph SPECtra) kernel based on the widths of the intersections of the ranges (7) to measure the similarity between two graphs.

If g_i and g_j contain common induced subgraphs of size m, the two ranges (7) for every k must intersect each other. This can only be satisfied when

$$\gamma_k(g_i) \leq \gamma_{k+|g_j|-m}(g_j) \ \texttt{and} \ \gamma_k(g_j) \leq \gamma_{k+|g_i|-m}(g_i), \tag{8}$$

because $\gamma_k(g_i) > \gamma_{k+|g_j|-m}(g_j)$ and $\gamma_k(g_j) > \gamma_{k+|g_i|-m}(g_i)$ cannot be satisfied simultaneously. Eq. (8) is equivalent to

$$(\gamma_{k+|g_j|-m}(g_j) - \gamma_k(g_i))(\gamma_{k+|g_i|-m}(g_i) - \gamma_k(g_j)) \geq 0, \tag{9}$$

and by taking the exponential of Eq. (9), we obtain the following inequality.

$$\begin{aligned} h(\cdot) = &\exp(\gamma_{k+|g_j|-m}(g_j)\gamma_{k+|g_i|-m}(g_i) + \gamma_k(g_i)\gamma_k(g_j)) \\ &\times \exp(-\gamma_{k+|g_j|-m}(g_j)\gamma_k(g_j)) \exp(-\gamma_{k+|g_i|-m}(g_i)\gamma_k(g_i)) \geq 1. \end{aligned}$$

$h(\cdot)$ can be further rewritten as $h(\cdot) = exp(\boldsymbol{\lambda}'^T_{mk}\boldsymbol{\theta}'_{mk}) \times (\phi_{\lambda_{mk}}\phi_{\theta_{mk}})$, where

$$\begin{aligned} \boldsymbol{\lambda}'_{mk} &= \begin{bmatrix} \gamma_k(g_i) \\ \gamma_{k+|g_i|-m}(g_i) \end{bmatrix}, \ \boldsymbol{\theta}'_{mk} = \begin{bmatrix} \gamma_k(g_j) \\ \gamma_{k+|g_j|-m}(g_j) \end{bmatrix}, \\ \phi_{\lambda_{mk}} &= \exp(-\gamma_{k+|g_i|-m}(g_i)\gamma_k(g_i)), \ \texttt{and} \\ \phi_{\theta_{mk}} &= \exp(-\gamma_{k+|g_j|-m}(g_j)\gamma_k(g_j)). \end{aligned}$$

We can easily see that the former exponential term in the rhs includes the inner product of $\boldsymbol{\lambda}'_{mk}$ and $\boldsymbol{\theta}'_{mk}$ in its exponent. Since the inner product is a PSD kernel function in the event that A is an identity matrix in Eq. (4), this term is a PSD kernel function based on Eq. (3). Besides, when $|\boldsymbol{x}| = |\boldsymbol{y}| = 1$ and $A = 1$, Eq. (4) mentions that the product of two independent scalars is a PSD kernel. Therefore, the latter product term of two ϕs is also a PSD kernel function based on Eq. (4). These observation shows that $h(\cdot)$ which is a product of the former and the latter terms is a PSD kernel based on Eq. (6). We restate $h(\cdot)$ as

$$k'_{mk}(\boldsymbol{\lambda}_{mk}, \boldsymbol{\theta}_{mk}) = \exp(\boldsymbol{\lambda}'^T_{mk}\boldsymbol{\theta}'_{mk}) \times (\phi_{\lambda_{mk}}\phi_{\theta_{mk}}),$$

$$\texttt{where} \ \boldsymbol{\lambda}_{mk} = \begin{bmatrix} \gamma_k(g_i) \\ \gamma_{k+|g_i|-m}(g_i) \\ \gamma_k(g_i)\gamma_{k+|g_i|-m}(g_i) \end{bmatrix} \ \texttt{and} \ \boldsymbol{\theta}_{mk} = \begin{bmatrix} \gamma_k(g_j) \\ \gamma_{k+|g_j|-m}(g_j) \\ \gamma_k(g_j)\gamma_{k+|g_j|-m}(g_j) \end{bmatrix}.$$

Since the ranges (7) must intersect each other for "all k" ($k = 1, \cdots, m$) when g_i and g_j contain a common subgraph g_s of size m, we take the product of k'_{mk} over all k, and have a new PSD kernel function:

$$k_m(\boldsymbol{\lambda}_m, \boldsymbol{\theta}_m) = \prod_{k=1}^{m} k'_{mk}(\boldsymbol{\lambda}_{mk}, \boldsymbol{\theta}_{mk}),$$

$$\texttt{where } \boldsymbol{\lambda}_m = [\boldsymbol{\lambda}_{m1}^T, \boldsymbol{\lambda}_{m2}^T, \cdots, \boldsymbol{\lambda}_{mm}^T]^T \texttt{ and } \boldsymbol{\theta}_m = [\boldsymbol{\theta}_{m1}^T, \boldsymbol{\theta}_{m2}^T, \cdots, \boldsymbol{\theta}_{mm}^T]^T,$$

based on Eq. (6). Furthermore, since the ranges (7) must intersect each other for "at least one of m" ($m = 1, \cdots, min(|g_i|, |g_j|)$) when g_i and g_j contain an arbitrary common subgraph g_s, we take a summation of k_m over all m which tends to be large when some k_m is large, and provide a new PSD kernel function:

$$k(g_i, g_j) = k(\boldsymbol{\lambda}, \boldsymbol{\theta}) = \sum_{m=1}^{min(|g_i|,|g_j|)} k_m(\boldsymbol{\lambda}_m, \boldsymbol{\theta}_m), \tag{10}$$

$$\texttt{where } \boldsymbol{\lambda} = [\boldsymbol{\lambda}_1^T, \boldsymbol{\lambda}_2^T, \cdots, \boldsymbol{\lambda}_m^T]^T \quad \texttt{and} \quad \boldsymbol{\theta} = [\boldsymbol{\theta}_1^T, \boldsymbol{\theta}_2^T, \cdots, \boldsymbol{\theta}_m^T]^T,$$

based on Eq. (5). $k(\boldsymbol{\lambda}, \boldsymbol{\theta})$ is expected to have a high score when measuring the similarity between the graphs, especially where g_i and g_j have large common induced subgraphs. We call this kernel function the *SPEC* kernel, and use it in an SVM to classify graphs.

By summarizing the above discussion, the following lemmas are derived.

Lemma 1. *The SPEC kernel is a PSD kernel.* ■

Lemma 2. *The SPEC kernel $k(g_i, g_j)$ is computed in $O(|g|^3)$ where $|g|$ is the maximum number of vertices in g_i and g_j.* ■

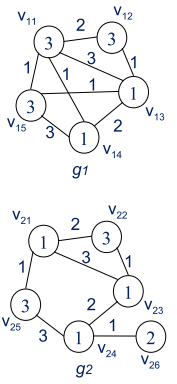
Proof. To obtain the eigenvalues from the adjacency matrices of g_i and g_j requires computation time of $O(|g|^3)$. In addition, we require $O(|g|^2)$ computation time to compute Eq. (10) from the eigenvalues. Therefore, the SPEC kernel $k(g_i, g_j)$ can be computed in $O(|g|^3)$. □

4 Optimizing Graph Spectra for Large Graph Classification

In the previous section, we gave the details of the SPEC kernel, constructed especially for the classification of graphs with large common induced subgraphs. In this section, we propose a new algorithm, called OPTSPEC to obtain high classification accuracy for graphs using the SPEC kernel.

4.1 Basic Idea for Optimizing Graph Spectra

The SPEC kernel $k(g_i, g_j)$ between graphs g_i and g_j is computed from eigenvalues of adjacency matrices $M(g_i)$ and $M(g_j)$ defined by Eq. (1), and the eigenvalues depend on a matrix generator vector $\boldsymbol{x}$ defining elements of $M(g_i)$ and $M(g_j)$. Therefore, $k(g_i, g_j)$ depends on a matrix generator vector $\boldsymbol{x}$. Even if the value of only a single element of $\boldsymbol{x}$ is changed, it leads to a change in the graph spectra of g_i and g_j, and thus the intersection of the ranges (7) is also changed, as well as the value of the kernel function $k(g_i, g_j)$. In other words, by choosing a suitable matrix generator vector $\boldsymbol{x}$, there is a possibility to accurately measure the similarity between two arbitrary graphs using the Interlace Theorem.



Coding table

Graph sub-structure	x_1	x_2
1 1 1	2	3
1 1 2	3	2
1 1 3	0	5
1 2 1	1	0
1 2 3	4	2
1 3 1	0	1
1 3 3	5	2
3 1 3	4	0
3 2 3	1	5

Graph spectra of g_1 and g_2 under x_1

$\gamma'(g_1)$ = {-7.76, -2.78, 0, 2.78, 7.76}

$\gamma'(g_2)$ = {-6.44, -3.67, 0, 0, 3.67, 6.44}

Graph spectra of g_1 and g_2 under x_2

$\gamma''(g_1)$ ={-7.41, -7, 1.34, 3, 10.07}

$\gamma''(g_2)$ ={-5.21, -4.11, -0.34, 0, 3.58, 6.08}

Fig. 3. Example of using different matrix generator vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$

Figure 3 shows an example of a difference between two matrix generator vectors in the computation of graph spectra. Given graphs g_1 and g_2, two graph spectra $\gamma'(g_1)$ and $\gamma'(g_2)$ are calculated using the matrix generator vector $\boldsymbol{x}_1$ shown in the second column of the table in Fig. 3, and two graph spectra $\gamma''(g_1)$ and $\gamma''(g_2)$ are calculated using another vector $\boldsymbol{x}_2$ shown in the third column of the table in Fig. 3. From this example, we can see that the graph spectra of g_1 and g_2 are completely different. Moreover, while the largest common induced subgraph of g_1 and g_2 is the subgraph consisting of 3 vertices (v_{12}, v_{13}, and v_{14} in g_1 and v_{22}, v_{23}, and v_{24} in g_2), we cannot conclude this fact by using $\gamma'(g_1)$ and $\gamma'(g_2)$ and the Interlace Theorem. The ranges of $\gamma'_1(g_s)$ for a subgraph g_s of size 4 in g_1 and g_2 are [-7.76,-2.78] and [-6.44,-2.78], respectively, and their intersection for $\gamma'_1(g_s)$ is [-6.44,-2.78]. Similarly, the intersections for $\gamma'_2(g_s)$, $\gamma'_3(g_s)$, and $\gamma'_4(g_s)$ are calculated as [-2.78,0], [0,2.78], and [2.78,6.44], respectively. Since the intersections for four elements of $\gamma'(g_s)$ exist, we cannot conclude the correct size of the largest induced subgraph g_s between g_1 and g_2. On the other hand, the ranges of $\gamma''_1(g_s)$ for a subgraph g_s of size 4 in g_1 and g_2 are [-7.41,-7] and [-5.21,-0.34], respectively. These non-intersecting ranges exclude a common induced subgraph of size 4 in g_1 and g_2.

As shown in the above example, if we choose an unsuitable matrix generator vector $\boldsymbol{x}$, it is hard to correctly know the existence of a common induced subgraph of two arbitrary graphs g_i and g_j and the maximum size of their common induced subgraph. Thus, the importance of a proper selection of $\boldsymbol{x}$ is very clear. In the next subsection, we propose a method for choosing a suitable matrix generator vector $\boldsymbol{x}$ based on an optimization technique.

4.2 Algorithm for Optimizing Graph Spectra for Classification

The data handled in this paper is graphs whose class labels are determined by whether or not they contain some large induced subgraphs belonging to a given graph set S, and the problem we intend to solve in this paper is the construction

a classifier for the graphs using an SVM and the SPEC kernel. Although using a suitable matrix generator vector $\boldsymbol{x}$ is expected to achieve high performance of the SVM with the SPEC kernel, obtaining the suitable matrix generator vector is difficult by hand-tuning. We, then, empirically optimize the matrix generator vector $\boldsymbol{x}$ using training examples together with the training of the SVM. If a suitable matrix generator vector is chosen to compute the SPEC kernel, we can correctly measure the similarity between graphs with the kernel, and the number of misclassified training examples is reduced using an SVM with the SPEC kernel. Therefore, we optimize the matrix generator vector $\boldsymbol{x}$ so that the number of examples misclassified by an SVM is reduced using training examples.

We propose the OPTSPEC algorithm for optimizing a matrix generator vector based on the framework of Generalized Multiple Kernel Learning (GMKL) [14]. GMKL minimizes the number of misclassified training examples by alternately learning parameters of an SVM and a parameter $\boldsymbol{x}$ of the kernel function. Given a set of training examples, the OPTSPEC algorithm aims at minimizing the distance between boundaries on support vectors and the number of misclassified training examples in the classification of graphs as follows.

$$\min_{\boldsymbol{x}} T(\boldsymbol{x}) \; s.t. \; \boldsymbol{x} \geq 0 \;\; \texttt{(outer loop)} \tag{11}$$

$$where \; T(\boldsymbol{x}) = \min_{\boldsymbol{w}} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \frac{C}{2}\sum_{i=1}^{n} \max(0, 1 - y_i f(g_i)) + r(\boldsymbol{x}), \;\; \texttt{(inner loop)} \tag{12}$$

where $\boldsymbol{w}$ is a parameter learned by the SVM, C is a constant value specified by the user, f is a function to be learned as mentioned in Section 2, and r is a regularizer to incorporate a scale parameter within $T(\boldsymbol{x})$ in form of $r(\boldsymbol{x}) = ||\boldsymbol{x}||^2 - 1$. In OPTSPEC, the constraint $\boldsymbol{x} \geq 0$ is relaxed so that the learned kernel is PSD as mentioned in [14]. The optimal kernel is learned by optimizing over $\boldsymbol{x}$ in the outer loop, while the matrix generator vector $\boldsymbol{x}$ remains fixed and the parameter $\boldsymbol{w}$ are learned using an SVM in the inner loop. In the calculation of the outer loop, the matrix generator vector $\boldsymbol{x}$ is updated to another matrix generator vector $\boldsymbol{x} - \alpha(\frac{\partial T}{\partial x_1}, \frac{\partial T}{\partial x_2}, \cdots, \frac{\partial T}{\partial x_{|\boldsymbol{x}|}})^T$ using the Steepest Descent method. In this computation, since $T(\boldsymbol{x})$ which contains the number of misclassified training examples $\frac{1}{2}\sum_{i=1}^{n} \max(0, 1 - y_i f(g_i))$ in its second term is a discrete function, we cannot calculate its differentials. To overcome this difficulty, we employ sensitivity analysis to calculate $\frac{\partial T}{\partial x_i} = \frac{T(\boldsymbol{x} + \tau \Delta \boldsymbol{x}_i) - T(\boldsymbol{x})}{\tau \Delta \boldsymbol{x}_i^T \Delta \boldsymbol{x}_i}$ $(i = 1, \cdots, |\boldsymbol{x}|)$ for each element x_i, one by one, where

$$\Delta \boldsymbol{x}_i^T = \begin{array}{c} \begin{array}{ccccccccc} 1 & \cdots & i-1 & i & i+1 & \cdots & |\boldsymbol{x}| \end{array} \\ \begin{pmatrix} 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{pmatrix} \end{array}.$$

The computations in the inner loop using an SVM and the outer loop using the Steepest Descent method are alternately continued, while the number of misclassified examples is reduced.

5 Experiments

The proposed method was implemented in Java. All experiments were done on an Intel Xeon L5240 3 GHz computer with 2 GB memory and running Microsoft Windows 2008 Server. We compared the accuracy of the training and the prediction performance of OPTSPEC with those of SVMs using the Random Walk Kernel and Neighborhood Hash Kernel. In the remainder of this paper, for simplicity, we refer to the SVMs using Random Walk Kernel and Neighborhood Hash Kernel as RWK-SVM and NHK-SVM, respectively. We varied parameters $\lambda = \{0.9, 0.8, \cdots, 0.2, 0.1, 0.01, 0.001\}$ which represents the termination probability for the random walk kernel and $R = \{1, 2, \cdots, 9, 10, 20, \cdots, 90, 100, 150, 200\}$ which represents maximum order of neighborhood hash for Neighborhood Hash Kernel. For RWK-SVM and NHK-SVM, we show the best prediction accuracy for various λ and R in the next subsections. For learning from the kernel matrices generated by the above graph kernels, we used the LIBSVM package[3] using 10-fold cross validation. The performance of the proposed method was evaluated using artificial and real-world data.

Table 1. Default parameters for the data generation program

	Avg. size of graphs	Proportion of size of induced subgraphs	Prob. of edge existence	# of vertex and edge labels
Default values	$\|g\| = 100$	$p_V = 0.7$	$p = 5\%$	$\|L\| = 3$

5.1 Experiments on Artificial Datasets

We generated artificial datasets of graphs using the four parameters listed in Table 1. For each dataset, 50 graphs, each with an average of $|g|$ vertices, were generated. Two vertices in a graph were connected with probability p of the existence of an edge, and one of $|L|$ labels was assigned to each vertex or edge in the graph. In parallel with the dataset generation and using the same parameters p and $|L|$, three graphs g_{s1}, g_{s2}, and g_{s3} with an average of $p_V \times |g|$ vertices were also generated for embedding in the 50 graphs as common induced subgraphs. g_{s1} was randomly embedded in half of the 50 graphs. The embedding process was then repeated using g_{s2} and g_{s3}. Finally, the class label 1 was assigned to the graphs containing g_{s3}, which was the last to be embedded, while the class label -1 was assigned to the other graphs. Even under this tough condition in which graphs have high similarity, *i.e.*, they contain parts of g_{s1} and/or g_{s2} as common induced subgraphs, a good classifier should be able to classify graphs labeled according to whether or not they contain g_{s3} correctly.

First, we varied only p_V to generate various datasets with the other parameters set to their default values. The proportion of subgraphs embedded in each dataset to the average size of the graphs was varied from 0.1 to 0.9. The values in Table 2 denote the average and standard deviation of the accuracy of the three

[3] `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

Table 2. Results for various p_V: accuracy (standard deviation)

p_V	NHK-SVM		RWK-SVM		OPTSPEC		BoostOPTSPEC	
	Training	Test	Training	Test	Training	Test	Training	Test
0.1	95% (4%)	58% (17%)	52% (1%)	34% (13%)	68% (10%)	60% (16%)	82% (10%)	56% (28%)
0.3	74% (6%)	30% (24%)	52% (2%)	28% (13%)	67% (10%)	57% (17%)	79% (10%)	54% (26%)
0.5	80% (6%)	50% (22%)	52% (2%)	30% (13%)	83% (9%)	72% (19%)	93% (4%)	70% (25%)
0.7	99% (2%)	54% (22%)	52% (2%)	30% (13%)	86% (20%)	74% (31%)	96% (4%)	86% (14%)
0.9	100% (0%)	48% (24%)	51% (2%)	32% (13%)	98% (6%)	96% (8%)	100% (0%)	92% (10%)

Table 3. Results for various $|g|$: accuracy (standard deviation)

$\|g\|$	NHK-SVM		RWK-SVM		OPTSPEC		Boost OPTSPEC	
	Training	Test	Training	Test	Training	Test	Training	Test
36	99% (2%)	70% (22%)	89% (3%)	86% (13%)	92% (6%)	88% (22%)	99% (1%)	94% (14%)
60	90% (3%)	60% (18%)	52% (2%)	28% (13%)	90% (10%)	64% (25%)	98% (2%)	82% (15%)
100	99% (2%)	54% (22%)	52% (2%)	52% (2%)	86% (20%)	74% (31%)	96% (4%)	86% (14%)
180	91% (2%)	68% (20%)	51% (1%)	38% (6%)	85% (15%)	72% (27%)	98% (3%)	80% (16%)
360	100% (0%)	38% (28%)	–	–	93% (3%)	90% (14%)	–	–

classifiers in the experiments. RWK-SVM did not perform well with either the training or test datasets. Although the accuracy of NHK-SVM is high for the training datasets, it was unable to classify graphs correctly in the test datasets. On the other hand, the accuracy of OPTSPEC is high for the test dataset, especially with a high p_V, compared with both NHK-SVM and RWK-SVM. Since the accuracy of OPTSPEC is constantly higher than 50%, we combined our OPTSPEC algorithm with AdaBoost [3] to create a strong learner. The experimental results of OPTSPEC combined with AdaBoost are shown in the last two columns of Table 2. By combining OPTSPEC with AdaBoost, the classification performance of OPTSPEC was enhanced particularly for the training data.

In the remaining sections, we set p_V to 0.7 as its default value to emphasize the classification performance of OPTSPEC for graphs labeled by whether or not they contain g_{s3}. Table 3 gives the experimental results for datasets generated by varying the values of $|g|$ and keeping the other parameters set to their default values. The average size $|g|$ of the graphs in the datasets was varied between 36 and 360. In the table, "–" indicates that results were not obtained due to intractable computation times exceeding 3 hours. The accuracy of OPTSPEC is comparable with that of NHK-SVM and RWK-SVM for small $|g|$. However, as $|g|$ increases, so the accuracy of NHK-SVM and RWK-SVM decreases, while the accuracy of OPTSPEC remains high. This arises from the fact that NHK-SVM and RWK-SVM make use of only small structures in the graphs to calculate their kernel functions, and therefore, they are unable to perform well with graphs containing large common induced subgraphs. On the other hand, in OPTSPEC, the possibility of a large common graph is identified in the computation of SPEC and then the matrix generator vector $\boldsymbol{x}$ is optimized based on GMKL.

Tables 4 and 5 give the experimental results for datasets generated by varying values of $|L|$ and p, respectively, while keeping the other parameters set to their default values. The number of labels $|L|$ in the graphs was varied between 1 and 5 in Table 4, while the probability p of the existence of an edge between

Table 4. Results for various $|L|$: accuracy (standard deviation)

$\|L\|$	NHK-SVM		RWK-SVM		OPTSPEC		Boost OPTSPEC	
	Training	Test	Training	Test	Training	Test	Training	Test
1	76% (3%)	40% (16%)	52% (1%)	28% (10%)	83% (18%)	55% (19%)	82% (10%)	56% (28%)
2	82% (5%)	44% (17%)	52% (2%)	32% (13%)	83% (7%)	74% (25%)	95% (3%)	80% (19%)
3	99% (3%)	54% (22%)	52% (2%)	52% (2%)	84% (11%)	70% (22%)	96% (4%)	86% (14%)
4	84% (3%)	38% (24%)	53% (1%)	26% (13%)	86% (20%)	74% (31%)	99% (3)%	96% (8%)
5	95% (2%)	70% (22%)	52% (2%)	30% (16%)	98% (3%)	98% (6%)	100% (0%)	96% (8%)

Table 5. Results for various p: accuracy (standard deviation)

p	NHK-SVM		RWK-SVM		OPTSPEC		Boost OPTSPEC	
	Training	Test	Training	Test	Training	Test	Training	Test
2.5	88% (3%)	46% (22%)	52% (1%)	32% (10%)	93% (4%)	80% (16%)	99% (2%)	86% (14%)
5	99% (2%)	54% (22%)	52% (2%)	52% (2%)	84% (11%)	70% (22%)	96% (4%)	86% (14%)
10	89% (3%)	66% (28%)	52% (1%)	36% (8%)	92% (4%)	92% (10%)	99% (2%)	92% (14%)
20	76% (5%)	44% (23%)	52% (1%)	34% (13%)	94% (5%)	80% (16%)	98% (4%)	92% (14%)

two vertices was varied between 2.5 and 20 in Table 5. Table 4 shows that the accuracies of OPTSPEC and Boost OPTSPEC remains high for both training and test datasets as the number of labels in the graphs increases. This originates from the fact that the size of the matrix generator vector $\boldsymbol{x}$ becomes large when the number of labels in the graphs increases, which leads to high probability of discovering a suitable vector $\boldsymbol{x}$. Table 5 shows that the accuracies of OPTSPEC and Boost OPTSPEC are high. This is because most of the elements in the adjacency matrices become non-zero when p increases, which also leads to high probability of discovering a suitable vector $\boldsymbol{x}$. On the other hand, similar to the previous experiments, accuracy results for RWK-SVM are low, and the accuracy of NHK-SVM for the test data is also low compared with that of OPTSPEC, while the accuracy of NHK-SVM being high for the training datasets.

5.2 Experiment with Real-World Graphs

To assess the practicability of our proposed method, we experimented on the email-exchange history data of the Enron company [2]. We transformed the mail exchange history per week data to one graph, and, having preprocessed the data as described below, obtained a dataset consisting of 123 graphs for 123 weeks. Each person in the company is represented by a single vertex labeled with his or her position in the company, for example "CEO", "Director", "Employee", "Lawyer", "Manager", "President", "Trader" and "Vice President". An edge connecting two vertices is included if the corresponding individuals exchanged emails for a week. The maximum size of the graphs in the dataset is 70, and the average edge existence probability is 4.9%. Since our aim is to evaluate the proposed method using large graphs, we chose 50 large graphs from the 123, corresponding to 50 continuous weeks.

Because it is very difficult to understand the common graphs contained in the graphs, we transformed the dataset by randomly choosing two graphs g_{s1} and g_{s2} from the 50 graphs of Enron data, and embedded these graphs within the

Table 6. Results for the Enron Dataset: accuracy (standard deviation)

Dataset	NHK-SVM		RWK-SVM		OPTSPEC		Boost OPTSPEC	
	Training	Test	Training	Test	Training	Test	Training	Test
D_1	99% (2%)	60% (18%)	52% (1%)	28% (16%)	85% (14%)	72% (20%)	98% (3%)	92% (14%)
D_2	100% (0%)	62% (21%)	52% (1%)	30% (13%)	91% (7%)	88% (13%)	99% (1%)	92% (14%)
D_3	100% (0%)	68% (20%)	51% (0%)	40% (0%)	96% (9%)	92% (10%)	99% (4%)	98% (6%)

other graphs. This embedding process is similar to the generation of artificial datasets described in the previous subsection. Accordingly, the graphs containing g_{s2} were given the class label 1, while the remaining graphs were given the class label -1. Since the embedded graphs were chosen from the Enron data, they also express the email exchanges for one week, and have an identical character to the other graphs in the dataset. Therefore, the characters of the embedded graphs do not change much. To obtain the exact performance of the proposed method, we chose three pairs of g_{s1} and g_{s2}, and embedded the three pairs in the graphs to create three datasets, respectively. We denote the datasets as D_1, D_2, and D_3, respectively. The numbers of vertices and edges of the embedded graphs in each dataset are as follows:

- D_1: g_{s1}: 25 vertices, 25 edges. g_{s2}: 28 vertices, 26 edges.
- D_2: g_{s1}: 25 vertices, 25 edges. g_{s2}: 22 vertices, 16 edges.
- D_3: g_{s1}: 22 vertices, 16 edges. g_{s2}: 27 vertices, 26 edges.

Table 6 gives the accuracy of the four methods with the three datasets. OPTSPEC and Boost OPTSPEC outperformed NHK-SVM and RWK-SVM on all three datasets. We confirmed that Boost OPTSPEC performed very well in improving the accuracy of OPTSPEC with the real-world data. In contrast, although NHK-SVM classified the training data with high accuracy, *i.e.*, over 85%, it did not predict the test graphs very well.

6 Discussion

Based on the various experiments, Boost OPTSPEC achieved the best performance for datasets with a variety of different specifications. For graphs whose class label is decided by whether or not they contain some large induced subgraphs in a given set, Boost OPTSPEC predicted the class of the test graphs with very high accuracy. By comparing the accuracy of Boost OPTSPEC and OPTSPEC, it is clear that the boosting method effectively increases the performance of OPTSPEC, thus enabling a powerful classifier for classifying graphs containing large common induced subgraphs. In contrast, although NHK-SVM showed high accuracy in the training process, the classifier was unable to perform the prediction well in almost all the experiments. RWK-SVM and NHK-SVM classifiers performed well only in the experiments with small graphs. The reason for this is that the two classifiers make use of very small structures in the graphs to measure the similarity between them, and therefore do not perform well in cases where the graphs contain large common induced subgraphs.

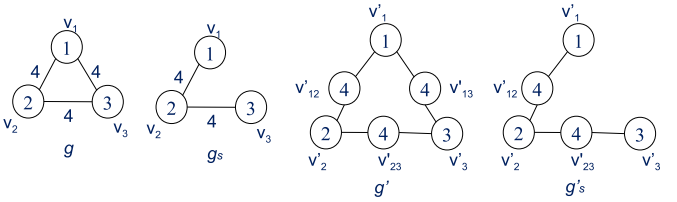


Fig. 4. Conversion of a graph g to another graph g'

In this paper, we focused on the common induced subgraphs within the graphs. Our SPEC kernel can be applied to classify graphs whose classes are determined by whether or not the graphs contain some specified common graph as general subgraphs using the following matrix representation. Given a graph $g = (V, E, L, l)$, the graph is converted to another graph $g' = (V', E', L', l')$, where $V' = V \cup E$, $E' \subseteq V \times E$, $L' = L$ and an edge e' between $v \in V$ and $e \in E$ exists in g' if v is directly linked to another vertex by e in g.

$$l'(v') = \begin{cases} l(v) & \texttt{if } v' \texttt{ corresponds to a vertex } v \texttt{ in } g, \\ l(e) & \texttt{otherwise if } v' \texttt{ corresponds to an edge } e \texttt{ in } g. \end{cases}$$

Since a general subgraph in g corresponds to an induced subgraph in g', the graph g' is represented by an adjacency matrix of size $(|V| + |E|) \times (|V| + |E|)$ using Eq. (1). Computing a kernel matrix using this matrix representation requires $O((|V| + |E|)^3)$ computation time. For example, the graph g with 3 vertices and 3 edges in Fig. 4 is converted to the another graph g' with 6 vertices. In this conversion as shown in Fig. 4, the graph g_s which is a general subgraph of g becomes an induced subgraph g'_s of g'. Therefore, the Interlace Theorem holds between adjacency matrices $M(g'_s)$ and $M(g')$, and the SPEC kernel can be computed from the adjacency matrices.

On the other hand, when given data are directed graphs, an adjacency matrix $M(g)$ of each directed graph g is converted to $M'(g) = \begin{pmatrix} 0 & M(g) \\ M(g)^T & 0 \end{pmatrix}$ to create a symmetric matrix. Therefore, the Interlace Theorem holds between adjacency matrices $M'(g_s)$ and $M'(g)$ where g_s is an induced subgraph of g, and the SPEC kernel can be computed from the adjacency matrices. In this case, computing the kernel matrix is $O((2|V|)^3) = O(|V|^3)$. Therefore, the proposed method can be applied to directed graphs whose classes are determined by whether or not the graph contains some common graphs as general subgraphs.

7 Conclusion

In this paper, we proposed a novel graph kernel named as SPEC based on graph spectra and the Interlace Theorem. We also proposed the OPTSPEC algorithm for optimizing the SPEC kernel used in an SVM for graph classification. We developed a graph classification program and confirmed the performance and practicability of the proposed method through computational experiments using artificial and real-world datasets.

Acknowledgment

We would like to thank Mr. Shohei Hido of IBM Research and Prof. Kouzo Ohara of Aoyama Gakuin University for their help and advice.

References

1. Alon, N., Krivelevich, M., Vu, V.H.: On the Concentration of Eigenvalues of Random Symmetric Matrices. Israel Journal of Mathematics 131(1), 259–267 (2001)
2. Enron Email Dataset, `http://www.cs.cmu.edu/~enron/`
3. Freund, Y., Schapire, R.E.: A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. Journal of Computer and System Sciences 55(1), 119–139 (1997)
4. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman, New York (1979)
5. Gärtner, T., Lloyd, J.W., Flach, P.A.: Kernels and Distances for Structured Data. Machine Learning 57(3), 205–232 (2004)
6. Hido, S., Kashima, H.: A Linear-Time Graph Kernel. In: Proc. of Int'l Conf. on Data Mining, pp. 179–188 (2009)
7. Hwang, S.: Cauchy's Interlace Theorem for Eigenvalues of Hermitian Matrices. American Mathematical Monthly 111, 157–159 (2004)
8. Ikebe, Y., Inagaki, T., Miyamoto, S.: The monotonicity theorem, Cauchy's interlace theorem, and the Courant-Fischer theorem. American Mathematical Monthly 94, 352–354 (1987)
9. Kashima, H., Inokuchi, A.: Kernels for graph classification. In: Proc. of ICDM Workshop on Active Mining (2002)
10. Kashima, H., Tsuda, K., Inokuchi, A.: Marginalized Kernels Between Labeled Graphs. In: Proc. of Int'l Conf. on Machine Learning, pp. 321–328 (2003)
11. Schölkopf, B., Tsuda, K., Vert, J.: Kernel Methods in Computational Biology. The MIT Press, Cambridge (2004)
12. Schölkopf, B., Smola, J.: Learning with kernels. MIT Press, Cambridge (2002)
13. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press, Cambridge (2004)
14. Varma, M., Rakesh Babu, B.: More Generality in Efficient Multiple Kernel Learning. In: Proc. of Int'l Conf. on Machine Learning, vol. 134 (2009)
15. Vishwanathan, S.V.N., Borgwardt, K.M., Schraudolph, N.N.: Fast Computation of Graph Kernels. In: Proc. of Annual Conf. on Neural Information Processing Systems, pp. 1449–1456 (2006)