IEICE TRANS. INF. & SYST., VOL.E93–D, NO.10 OCTOBER 2010

PAPER Special Section on Data Mining and Statistical Science

Cartesian Kernel: An Efficient Alternative to the Pairwise Kernel

Hisashi KASHIMA[†a)], **Satoshi OYAMA**[††], ***Members***, **Yoshihiro YAMANISHI**[†††], ***and*** **Koji TSUDA**[††††], ***Nonmembers***

SUMMARY Pairwise classification has many applications including network prediction, entity resolution, and collaborative filtering. The pairwise kernel has been proposed for those purposes by several research groups independently, and has been used successfully in several fields. In this paper, we propose an efficient alternative which we call a *Cartesian kernel*. While the existing pairwise kernel (which we refer to as the Kronecker kernel) can be interpreted as the weighted adjacency matrix of the Kronecker product graph of two graphs, the Cartesian kernel can be interpreted as that of the Cartesian graph, which is more sparse than the Kronecker product graph. We discuss the generalization bounds of the two pairwise kernels by using eigenvalue analysis of the kernel matrices. Also, we consider the N-wise extensions of the two pairwise kernels. Experimental results show the Cartesian kernel is much faster than the Kronecker kernel, and at the same time, competitive with the Kronecker kernel in predictive performance.

key words: *kernel methods, pairwise kernels, link prediction*

1. Introduction

Most phenomena in the world can be represented by sets of entities and sets of static and dynamic relationships among the entities. Such relationships include friendships among people, actions such as someone clicking an on-line advertisement, and physical interactions among proteins.

Supervised pairwise prediction aims to predict such pairwise relationships based on known relationships. It has many applications including network prediction, entity resolution, and collaborative filtering. For example, the network prediction problem has been studied in the context of predicting biological networks such as protein-protein interaction networks and gene regulatory networks in the bioinformatics area [1]–[3], and also in the context of link mining [4] in the data mining community.

Models for pairwise prediction should take a pair of instances as input, and output a relationship between the two instances. In this paper, we focus on the pairwise classification problem [*], where the task is to predict whether or not a relationship exists between two given nodes, and we apply kernel methods [5] to this problem. To apply kernel methods to pairwise classification, we need to define a kernel function between two arbitrary pairs of instances. Interestingly, three research groups have independently proposed essentially the same pairwise kernel by combining two instance-wise kernel functions [6]–[8]. Essentially the same kernels are also proposed in the context of multi-task learning [9], [10], where a pair-wise instance consists of a data instance and the task that the instance belongs to. In those existing works, the pairwise kernel matrix is considered as a Kronecker product of the two instance-wise kernel matrices. However, the kernel methods using the pair-wise kernel are significantly time-and-space-consuming since the pairwise kernel matrix is huge. For this reason, only sampled training data have been used in most of these applications.

In this paper, we propose a new pairwise kernel called a *Cartesian kernel* as a more efficient alternative to the existing pairwise kernel (which we refer to as the *Kronecker kernel*). The proposed kernel is defined as a Kronecker sum of two instance-wise kernel matrices, and therefore more efficient in both computation time and memory than the existing pairwise kernel. The experimental results using several real network data show that the proposed pairwise kernel is much faster than the Kronecker kernel, and at the same time, is competitive with the Kronecker kernel in predictive performance. Finally, we give the generalization bounds of the two pairwise kernels by using eigenvalue analysis of the kernel matrices [11], [12]. We also give the N-wise extensions of the pairwise kernels.

2. Pairwise Classification Problem and the Pairwise Kernel

In this section, we introduce the definition of the (binary) pairwise classification problem and review the existing pairwise kernel independently proposed by three research groups [6]–[8].

The standard binary classification problem aims to

Manuscript received November 26, 2009.
Manuscript revised March 24, 2010.

[†]The author is with the Department of Mathematical Informatics, Graduate School of Information Science and Technology, The University of Tokyo, Tokyo, 113–8656 Japan.
[††]The author is with the Graduate School of Information Science and Technology, Hokkaido University, Sapporo-shi, 060–0814 Japan.
[†††]The author is with Mines ParisTech, Centre for Computational Biology, 35 rue Saint-Honore, F-77305 Fontainebleau Cedex, France, Institut Curie, F-75248, Paris, France, and INSERM U900, F-75248, Paris, France.
[††††]The author is with AIST Computational Biology Research Center, Tokyo, 135–0064 Japan.
a) E-mail: kashima@mist.i.u-tokyo.ac.jp
DOI: 10.1587/transinf.E93.D.2672

[*]Although we focus on binary classification problems in this paper, use of the kernels presented in this paper is not limited to binary classification problems, but applicable to other cases such as multi-class classification, regression, and ranking.

learn a function $f : V \to \{+1, -1\}$, where V indicates the set of all possible instances. On the other hand, in the (binary) pairwise classification, the goal is to learn a function $f : V^{(1)} \times V^{(2)} \to \{+1, -1\}$, where $V^{(1)}$ and $V^{(2)}$ are two sets of all possible instances.

Let us assume that we are given a $|V^{(1)}| \times |V^{(2)}|$ class matrix $\boldsymbol{F}$ whose elements have one of $+1$ (positive class), -1 (negative class), and 0 (unknown). Our task is to fill in the unknown parts of the class matrix which have 0 values.

In the context of link prediction, the $\boldsymbol{F}$ can be regarded as the adjacency matrix for a network including $V^{(1)}$ and $V^{(2)}$ as its nodes. $[\boldsymbol{F}]_{i_1,i_2} = +1$ indicates that there is a link between $v^{(1)}_{i_1} \in V^{(1)}$ and $v^{(2)}_{i_2} \in V^{(2)}$, $[\boldsymbol{F}]_{i_1,i_2} = -1$ indicates that there is no link, and $[\boldsymbol{F}]_{i_1,i_2} = 0$ indicates that we do not know if there is a link. If the two sets are exclusive, i.e. $V^{(1)} \cap V^{(2)} = \phi$, the network is regarded as a bipartite graph. On the other hand, if the two sets are exchangeable, i.e. $V^{(1)} = V^{(2)} := V$, $\boldsymbol{F}$ is considered as a $|V| \times |V|$ adjacency matrix $\boldsymbol{F}$ for the set $V = (v_1, v_2, \ldots, v_{|V|})$. If the network is undirected, $\boldsymbol{F}$ will be symmetric. If the network is directed, $\boldsymbol{F}$ will be asymmetric, and $[\boldsymbol{F}]_{i_1,i_2}$ indicates whether or not a link exists from $v_{i_1} \in V$ to $v_{i_2} \in V$.

In addition to the adjacency matrix, we have two kernel matrices $\boldsymbol{K}^{(1)}$ and $\boldsymbol{K}^{(2)}$ for $V^{(1)}$ and $V^{(2)}$, respectively. A kernel matrix indicates similarities among nodes. For example, the (i, j)-th element of $\boldsymbol{K}^{(1)}$ indicates the similarity between the i-th node and the j-th node in $V^{(1)}$. Following the standard assumption of kernel methods [5], those kernel matrices are positive semi-definite so that the corresponding kernel functions are interpreted as inner products of two feature vectors of two nodes. In exchangeable cases, we assume $\boldsymbol{K}^{(1)} = \boldsymbol{K}^{(2)} := \boldsymbol{K}$.

3. Pairwise Kernels

In this section, we review the existing pairwise kernel independently proposed by three research groups [6]–[8].

Since we are interested in the classification of pairs of instances, we need a kernel function between two pairs of instances in order to apply kernel methods [5] to this problem. In many cases, it is rather easy to design kernels for two basic instances, so we can construct pairwise kernels by using these instance-wise kernels as building blocks.

Assume that we want to define a similarity between two pairs of instances $(v^{(1)}_{i_1}, v^{(2)}_{i_2})$ and $(v^{(1)}_{j_1}, v^{(2)}_{j_2})$. It is natural to say two pairwise relationships are similar if elements from the two relationships are similar. In other words, they are similar to each other if $v^{(1)}_{i_1}$ and $v^{(1)}_{j_1}$ are similar, and at the same time, $v^{(2)}_{i_2}$ and $v^{(2)}_{j_2}$ are similar. This idea motivates defining the pairwise similarity as the product of two instance-wise similarities with

$$k_{\otimes}((v^{(1)}_{i_1}, v^{(2)}_{i_2}), (v^{(1)}_{j_1}, v^{(2)}_{j_2})) = [\boldsymbol{K}^{(1)}]_{i_1,j_1}[\boldsymbol{K}^{(2)}]_{i_2,j_2}. \quad (1)$$

Since the products of Mercer kernels are also Mercer kernels [5], the above similarity measure is also a Mercer kernel if the element-wise kernels are Mercer kernels. In exchangeable and symmetric cases, the pairwise kernel between (v_{i_1}, v_{i_2}) and (v_{j_1}, v_{j_2}) can be made symmetric using

$$k^{\mathrm{SYM}}_{\otimes}((v_{i_1}, v_{i_2}), (v_{j_1}, v_{j_2})) = [\boldsymbol{K}]_{i_1,j_1}[\boldsymbol{K}]_{i_2,j_2} + [\boldsymbol{K}]_{i_1,j_2}[\boldsymbol{K}]_{i_2,j_1}. \quad (2)$$

The prediction of a kernel machine for a pair $(v^{(1)}_{i_1}, v^{(2)}_{i_2})$ is given as

$$[\boldsymbol{F}]_{i_1,i_2} = \sum_{(j_1,j_2)} \alpha(v^{(1)}_{j_1}, v^{(2)}_{j_2}) k_{\otimes}((v^{(1)}_{i_1}, v^{(2)}_{i_2}), (v^{(1)}_{j_1}, v^{(2)}_{j_2})),$$

where the αs are the model parameters of the kernel machine. In exchangeable and symmetric cases, it becomes

$$[\boldsymbol{F}]_{i_1,i_2} = \sum_{(j_1,j_2):j_1<j_2} \alpha(v_{j_1}, v_{j_2}) k^{\mathrm{SYM}}_{\otimes}((v_{i_1}, v_{i_2}), (v_{j_1}, v_{j_2})).$$

Note that $\alpha(v_{j_1}, v_{j_2})$ for (v_{j_1}, v_{j_2}) such that $j_1 \geq j_2$ is not used because of symmetry.

The kernel matrix for the pairwise kernel is equivalently written as the Kronecker product [13] of two instance-wise kernel matrices as

$$\boldsymbol{K}_{\otimes} = \boldsymbol{K}^{(2)} \otimes \boldsymbol{K}^{(1)}, \quad (3)$$

where the $(v^{(1)}_{i_1} + v^{(2)}_{i_2}|V^{(2)}|), v^{(1)}_{j_1} + v^{(2)}_{j_2}|V^{(2)}|)$-th element is $k_{\otimes}((v^{(1)}_{i_1}, v^{(2)}_{i_2}), (v^{(1)}_{j_1}, v^{(2)}_{j_2}))$.

The pairwise kernel matrix can be interpreted as a weighted adjacency matrix of the Kronecker product graph [14] for the two graphs whose weighted adjacency matrices are the instance-wise kernel matrices. Therefore, we refer to this pairwise kernel as the *Kronecker kernel* to distinguish it from the kernel we will propose in the next section.

4. Cartesian Kernel: A New Pairwise Kernel

In this section, we propose a new efficient pairwise kernel. At the end of the previous section, we mentioned the relationship between the existing pairwise kernel and a Kronecker product graph. Therefore it is natural to imagine that we can design another pairwise kernel based on another kind of product graph. In this paper, we adopt the product graph called the Cartesian product graph [14].

Assume that we have two graphs $G^{(1)}$ and $G^{(2)}$ whose sets of nodes are $V^{(1)}$ and $V^{(2)}$, respectively. The product graph of $G^{(1)}$ and $G^{(2)}$ has nodes $V^{(1)} \times V^{(2)}$, each of whose nodes is defined as a pair of nodes from the original graphs. Let $(v^{(1)}_{i_1}, v^{(2)}_{i_2})$ and $(v^{(1)}_{j_1}, v^{(2)}_{j_2})$ be two node pairs in the product graph. In Kronecker product graphs, a link between these two pairs exists if and only if there is a link between $v^{(1)}_{i_1}$ and $v^{(1)}_{j_1}$ in $G^{(1)}$ and there is a link between $v^{(2)}_{i_2}$ and $v^{(2)}_{j_2}$ in $G^{(2)}$.

On the other hand, in Cartesian product graphs, a link between these two pairs exists if and only if $v^{(1)}_{i_1} = v^{(1)}_{j_1}$ in $G^{(1)}$ and there is a link between $v^{(2)}_{i_2}$ and $v^{(2)}_{j_2}$ in $G^{(2)}$, or there is a link between $v^{(1)}_{i_1}$ and $v^{(1)}_{j_1}$ in $G^{(1)}$ and $v^{(2)}_{i_2} = v^{(2)}_{j_2}$ in $G^{(2)}$.

We can see that the condition for a link existing in Cartesian product graphs is more strict than for Kronecker product graphs.

Inspired by the definition of the Cartesian product graph, we define the *Cartesian kernel* between $(v_{i_1}^{(1)}, v_{i_2}^{(2)})$ and $(v_{j_1}^{(1)}, v_{j_2}^{(2)})$ as

$$k_{\otimes}((v_{i_1}^{(1)}, v_{i_2}^{(2)}), (v_{j_1}^{(1)}, v_{j_2}^{(2)})) = [\boldsymbol{K}^{(1)}]_{i_1,j_1}\delta(i_2 = j_2) + \delta(i_1 = j_1)[\boldsymbol{K}^{(2)}]_{i_2,j_2}, \quad (4)$$

where δ is an indicator function, which returns 1 when its argument is true and 0 otherwise. Since δ can be regarded as an identity kernel, this similarity measure is also a Mercer kernel if the element-wise kernels are Mercer kernels. In exchangeable and symmetric cases, the kernel between (v_{i_1}, v_{i_2}) and (v_{j_1}, v_{j_2}) can be symmetrized as

$$k_{\otimes}^{\mathrm{SYM}}((v_{i_1}, v_{i_2}), (v_{j_1}, v_{j_2})) = [\boldsymbol{K}]_{i_1,j_1}\delta(i_2 = j_2) + \delta(i_1 = j_1)[\boldsymbol{K}]_{i_2,j_2} + [\boldsymbol{K}]_{i_1,j_2}\delta(i_2 = j_1) + \delta(i_1 = j_2)[\boldsymbol{K}]_{i_2,j_1}. \quad (5)$$

The kernel matrix of the Cartesian kernel is equivalently written as the Kronecker *sum* [13] of two instance-wise kernel matrices as

$$\boldsymbol{K}_{\oplus} = \boldsymbol{K}^{(2)} \oplus \boldsymbol{K}^{(1)},$$

where the Kronecker sum operation is defined as

$$\boldsymbol{K}^{(2)} \oplus \boldsymbol{K}^{(1)} = \boldsymbol{K}^{(2)} \otimes \boldsymbol{I} + \boldsymbol{I} \otimes \boldsymbol{K}^{(1)}.$$

At first sight, the size of the Cartesian kernel matrix seems to be the same as the Kronecker kernel. However the number of the non-zero elements in the kernel matrix is much smaller, since the Cartesian kernel is based on the Kronecker products of an element-wise kernel matrix and an identity matrix.

Finally, we describe the computational efficiency of the Cartesian kernel. While the Kronecker kernel can give a score greater than 0 between any two pairs, the Cartesian kernel can give a non-zero value only to those pairs that share one of their instances. The computational cost of one evaluation of a model output largely depends on the number of non-zero entries, since we can skip multiplications of model parameters and zero kernel values. In addition, we have no multiplication in the Cartesian kernel while the Kronecker kernel has one (or two in symmetric cases). Those facts suggest that the Cartesian kernel will be much faster to compute than the Kronecker kernel.

5. Generalization Bounds for the Pairwise Kernels

To compare the theoretical behaviors of the Kronecker kernel and the Cartesian kernel, we investigate the generalization bounds for them. It is known that the generalization bound for a kernel machine such as a support vector machine is given by using the distribution of the eigenvalues of the kernel matrix [11], [12].

In the following analysis, we assume that the set of all possible data points $X = \{x_1, x_2, \ldots, x_M\}$ ($X = V^{(1)} \times V^{(2)}$ in our case of pairwise classification) are known in advance of the training phase. Let $Y = \{1, -1\}$ be the set of labels. We also assume that $(x, y) \in Z = X \times Y$ follows a certain distribution $P(x, y)$. The *expected risk* of a hypothesis $h \in \mathcal{H}$ is given by

$$R(h) = \sum_{(x,y)\in Z} \delta(h(x)y \le 0)P(x, y).$$

Given a set of labeled samples $\{(x_i, y_i) : i \in \{1, \ldots, m\}\}$ from Z with size $m < M$, the *empirical margin risk* for a certain margin γ is defined by the rate of the samples with $h(x_i)y_i < \gamma$:

$$R_s^{\gamma}(h) = \frac{1}{m}\sum_{i=1}^{m} \delta(h(x_i)y_i < \gamma).$$

Then the following theorem [12] gives an upper bound for the expected risk.

Theorem 1. Let $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_M$ be the set of eigenvalues of the kernel matrix derived from the set of all possible data points. Consider the hypothesis class $\mathcal{F}(c)_B = \{\langle w, x\rangle + b : \|w\| \le c, |b| \le B\}$. Then the following inequality holds simultaneously for all $\gamma \in (8\Upsilon(n), 1]$:

$$P_{s\in Z^m}\left(\exists h \in \mathcal{H}(c)_B : R(h) \ge R_s^{\gamma}(h) + \sqrt{\left(n\ln 2 + \ln\left(\frac{\lceil c\rceil}{\theta\gamma}\right)\left\lceil\frac{8B}{\gamma}\right\rceil\right)/(2m)}\right) \le \theta,$$

where

$$\Upsilon(n) = \min_{j\in\{1,\ldots,n-1\}} 6\cdot 2^{-\frac{(j-1)}{k(2^{j-1})}}(\lambda_1\cdots\lambda_{k(2^{j-1})})^{\frac{1}{2k(2^{j-1})}}c(n, j),$$

$$k(l) = \min\left\{k \in \{1, \ldots, M\} : \lambda_{k+1} \le \left(\lambda_1\cdots\lambda_k/l^2\right)^{\frac{1}{k}}\right\},$$

$$c(n, j) = \min\left(1, 1.86\sqrt{\log_2(M/(n-j)+1)/(n-j)}\right).$$

To compute the generalization bounds, we need to compute the eigenvalues of the Kronecker product $\boldsymbol{K}_{\otimes}$ or the Kronecker sum $\boldsymbol{K}_{\oplus}$ of the two instance-wise kernel matrices. It is difficult to directly compute the eigenvalues for the Kronecker product or the Kronecker sum, since they are very large [†]. However, we can compute them from the eigenvalues of the instance-wise kernel matrices by using the following theorem [13].

Theorem 2. Let $\{\lambda_i^{(1)}\}$ and $\{\lambda_j^{(2)}\}$ be the sets of eigenvalues of kernel matrices $\boldsymbol{K}^{(1)}$ and $\boldsymbol{K}^{(2)}$, respectively. The set of eigenvalues of the Kronecker product $\boldsymbol{K}^{(2)} \otimes \boldsymbol{K}^{(1)}$ is $\{\lambda_i^{(1)}\lambda_j^{(2)}\}$ and the set of eigenvalues of the Kronecker sum $\boldsymbol{K}^{(2)} \oplus \boldsymbol{K}^{(1)}$ is $\{\lambda_i^{(1)} + \lambda_j^{(2)}\}$.

[†]For examples, the size of a pairwise kernel matrix for the metabolic network is $446,224 \times 446,224$, and that for the co-authoring network is $8,208,225 \times 8,208,225$.

6. *N*-wise Extensions of the Pairwise Kernels

Finally, we mention the N-wise extensions of the two pairwise kernels, since it is a natural question whether we can generalize the pairwise kernels for N-wise prediction.

Let us assume that we have N sets, $V^{(1)}, V^{(2)}, \ldots, V^{(N)}$, and want to define a kernel function between two N-wise relationships $I = (v_{i_1}^{(1)}, v_{i_2}^{(2)}, \ldots, v_{i_N}^{(N)}) \in V^{(1)} \times V^{(2)} \times \ldots \times V^{(N)}$ and $J = (v_{j_1}^{(1)}, v_{j_2}^{(2)}, \ldots, v_{j_N}^{(N)}) \in V^{(1)} \times V^{(2)} \times \ldots \times V^{(N)}$.

Similarly to the pairwise case, the N-wise extension of the Kronecker kernel (1) can be defined as

$$k_{\otimes}(I, J) = \prod_{\ell=1}^{N} [\boldsymbol{K}^{(\ell)}]_{i^{(\ell)}, j^{(\ell)}},$$

where $k^{(\ell)}$ is the instance-wise kernel function for the ℓ-th set $V^{(\ell)}$. For the Cartesian kernel, its N-wise counterpart can be given as

$$k_{\oplus}(I, J) = \sum_{\ell'=1}^{N} k^{(\ell')}(i^{(\ell')}, j^{(\ell')}) \prod_{\ell \neq \ell'} \delta(i^{(\ell)} = j^{(\ell)}).$$

The prediction for an N-let I is written as

$$f(I) = \sum_{J} \alpha(J) k_{\circledast}(I, J),$$

where $\circledast$ is replaced by $\otimes$ when we use the Kronecker kernel and by $\oplus$ when we use the Cartesian kernel.

Also, the kernel function can be represented as an $\prod_\ell |V^{(\ell)}| \times \prod_\ell |V^{(\ell)}|$ kernel matrix as

$$\mathsf{K}_{\otimes} = \mathsf{K}^{(N)} \otimes \mathsf{K}^{(N-1)} \otimes \cdots \otimes \mathsf{K}^{(1)}$$

for the Kronecker kernel, and

$$\mathsf{K}_{\oplus} = \mathsf{K}^{(N)} \oplus \mathsf{K}^{(N-1)} \oplus \cdots \oplus \mathsf{K}^{(1)}.$$

for the Cartesian kernel.

In exchangeable and symmetric cases, the N-wise kernel has to take into account all of the possible matchings, i.e. all of the permutations of I and J. Therefore, Eq. (2) and Eq. (5) are generalized as

$$k_{\circledast}^{\mathrm{SYM}}(I, J) = \sum_{J' = (j'_1, j'_2, \ldots, j'_N) \in P(J)} k_{\circledast}(I, J'),$$

where $\circledast \in \{\otimes, \oplus\}$ according to the employed kernel, and $P(J)$ is the set of all of the possible permutations of J. Also in prediction, J and its permutations are not distinguished, so we only consider $J = (j^{(1)}, \ldots, j^{(N)})$ such that $j^{(1)}, \ldots, j^{(N)}$ are in order of increasing. Denoting such a set of Js by O, the prediction for an N-let I is given as

$$f(I) = \sum_{J \in O} \alpha(J) k_{\circledast}^{\mathrm{SYM}}(I, J),$$

where $\circledast \in \{\otimes, \oplus\}$ according to the employed kernel. Note that $\alpha(J)$ for $J \notin O$ are not assigned any values.

7. Experiments

In this section, we show several experimental to show that the Cartesian kernel is more efficient than the Kronecker kernel while keeping reasonable predictive performance by using several real-world data sets. We also compare the generalization bounds derived in Sect. 5 and the actual performances on the data sets.

7.1 Data Sets

We used four data sets for network prediction. Three of them are biological networks, and one of them is a social network. Also, three data sets are symmetric networks, and one is an asymmetric network.

The first data set [3] contains the metabolic pathway network of the yeast S. Cerevisiae in the KEGG/PATHWAY database [15]. Proteins are represented as nodes, and a symmetric edge indicates that the two proteins are enzymes that catalyze successive reactions. The number of nodes in the network is 618, and the number of links is 2,782. In this data set, four element-wise kernel matrices based on gene expressions, chemical information, localization sites, and phylogenetic profiles are given. We used them as the kernel matrices or the similarity matrices [†].

The second data set is a protein-protein interaction network constructed by von Mering et al. [16]. We followed Tsuda and Noble [17], and used the medium confidence network. This network contains $2,617$ nodes and $11,855$ symmetric links. Each protein is given a 76-dimensional binary vector, each of whose dimensions indicates whether or not the protein is related to a particular function. We used the inner products of the vectors as the element-wise kernel matrix [††].

The third data set is a social network representing the co-authorships in the NIPS conferences, containing $2,865$ nodes and $4,733$ links. Authors correspond to nodes, and a symmetric link between two nodes indicates that there is at least one co-authored paper by the corresponding authors. Each author is given a feature vector, each of whose dimensions corresponds to occurrences of a particular word in the author's papers. We used the inner product of the vectors as the element-wise kernel matrix [†††].

The fourth data set is the drug-target interaction network data constructed by Yamanishi et al. [18]. This data set includes four drug-target interaction networks, each of which is an asymmetric network representing interactions among drugs and one of the four target protein classes, i.e. enzymes, ion channels, G-protein-coupled receptors (GPCRs), and nuclear receptors [††††]. The numbers of known interactions involving enzymes, ion channels, GPCRs, and nuclear receptors are 2,926, 1,476, 635, and 90, respectively.

[†] http://web.kuicr.kyoto-u.ac.jp/supp/yoshi/ismb05/
[††] http://noble.gs.washington.edu/proj/maxent/
[†††] http://ai.stanford.edu/~gal/data.html
[††††] http://web.kuicr.kyoto-u.ac.jp/supp/yoshi/drugtarget/

The numbers of drugs are 445, 210, 223, and 54, respectively. The numbers of target proteins are 664, 204, 95, and 26, respectively. Each of the drugs is given as a labeled graph representing its chemical structure, and each target protein is given as a sequence of amino acids. We computed the values of the element-wise kernel function among the chemical structures of the drugs by using the SIMCOMP algorithm [19], and the values of the kernel function among the sequences of the target proteins by using a local alignment kernel based on the Smith-Waterman scores [20]. All of these element-wise kernel matrices are normalized so that all of their diagonals are 1.

7.2 Evaluation Method

Since the kernel matrices for the pair-wise prediction problems are inherently large, naive applications of the standard SVM implementations are not realistic, and we should resort to more efficient on-line type training methods which process the data one by one. Among them, PUMMA [21] whose solutions asymptotically converge to those by the support vector machine with squared hinge loss. Its computational cost is almost as same as that of perceptron, the most simplest on-line algorithm (which performed poorer than PUMMA in our pre-experiments.) The hyperparameter for regularization was set to $C = 1$, since it gave good prediction results to both kernel in our pre-experiments. All of the training data was processed thee times in the training phase.

We used AUC (Area Under the ROC Curve) a predictive performance measure, which indicates the probability of a randomly-picked positive test example given a higher score than a randomly-picked negative test example. The results were evaluated by 5-fold cross validation with 20% of all of the pairwise relationships as training data.

7.3 Results

Now, we show the predictive performances and execution times for the two pairwise kernels. Overall, the Cartesian kernel performs almost as well as the Kronecker kernel, but the Cartesian kernel is much faster than the Kronecker kernel.

Figures 1, 2, and 3 indicate the results for the metabolic networks, the protein-protein interaction network, and the social network, and the drug-target interaction networks, respectively. The gray bars indicate the AUCs of the Kronecker kernel, and the black bars represent the AUCs of the Cartesian kernel. The error bars indicate the standard deviations of the AUC values. In Fig. 1, each of the pairs of AUC bars indicates the results when we used gene expressions, chemical information, phylogenetic profiles, or localization sites for element-wise kernel matrices. In Fig. 2, the left pair of the AUC bars is for the protein-protein interaction networks, and the right pair is for the co-authoring network. In Fig. 3, each of the pairs of AUC bars indicates the results for the enzyme class, the GPCR class, the ion channel class, or the nuclear receptor class.

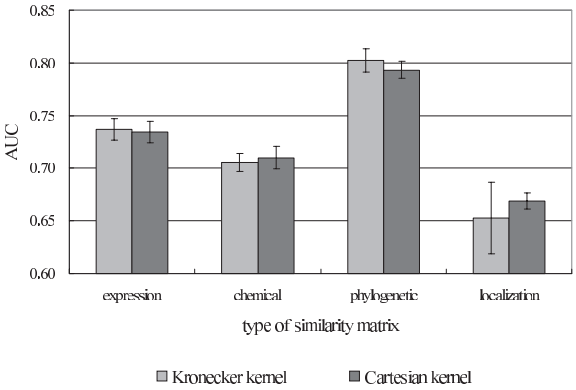

Fig. 1 Summary of results for the KEGG metabolic network.

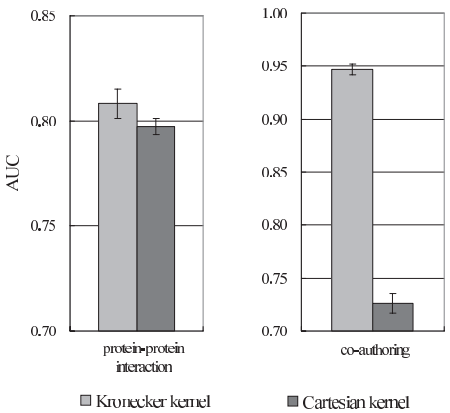

Fig. 2 Summary of results for the protein-protein interaction network and the co-authoring network.

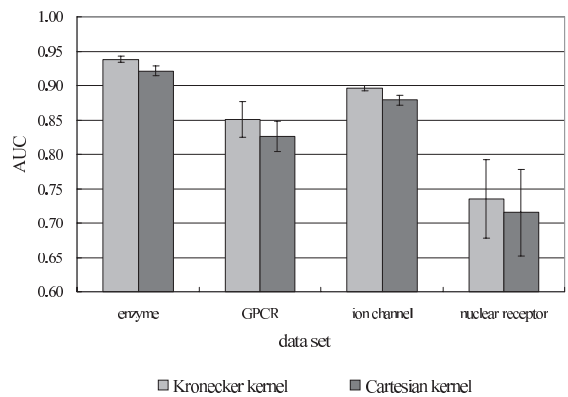

Fig. 3 Summary of results for the drug-target interaction networks.

We can see that the predictive performance of the Cartesian kernel is competitive with that of the Kronecker kernel except for the co-authoring network. The reason for the degraded performance by the Cartesian kernel for the co-authoring network is not clear, and we could not strongly support any reason in the theoretical analysis in the following section. However, it might be relaled to the network sparsity (since the co-authoring network is the most sparse), or due to differences in the innate traits of biological net-

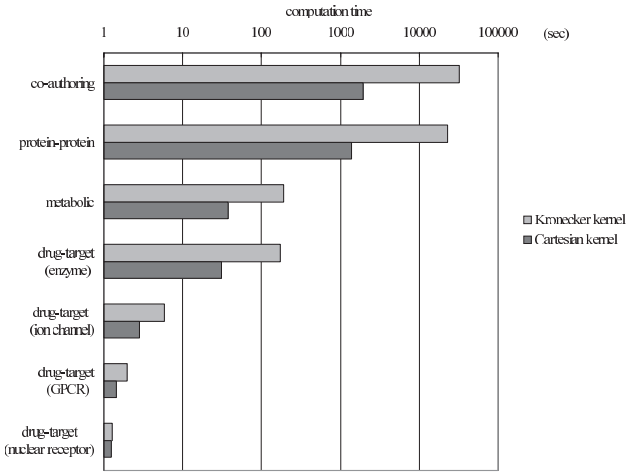

Fig. 4 Comparison of computation time. The pairs of the AUC bars are sorted in decreasing order of data set size.

works and social networks.

Now we turn to the execution times for the two pairwise kernels. Figure 4 shows the average training time for each data set on a logarithmic scale. We can see that the Cartesian kernel is up to 16 times faster than the Kronecker kernel. The differences are most remarkable when the network size is large.

Based on these results, we conclude that the Cartesian kernel is a promising alternative to the Kronecker kernel especially for large data sets.

7.4 Generalization Bounds

Figure 5 (left) shows the eigenvalues of the Kronecker kernel matrix and the Cartesian kernel matrix derived from phylogenetic profiles for the KEGG metabolic network. Also, Fig. 5 (right) shows the eigenvalues for the co-authoring network. The two pairwise kernels, the Kronecker kernel and the Cartesian kernel, yield very different eigenvalue distributions. Although we do not show the eigenvalues for the other networks, the general trend is that the high-ranked eigenvalues of the Kronecker kernel are larger than those of the Cartesian kernel, while the low-ranked eigenvalues of the Cartesian kernel are larger than those of the Kronecker kernel. In other words, the eigenvalues of the Kronecker product decay faster than those of the Kronecker sum.

The generalization bounds computed for the metabolic network and the co-authoring network are shown in Fig. 6. The bounds for the Kronecker kernels are smaller than the bounds for the Cartesian kernels. These theoretical results are consistent with the previous experimental results, where the Kronecker kernels were slightly superior to the Cartesian kernels, although there were a few exceptions and the differences were subtle in the most of the cases. This is probably because the bounds are not very tight, as mentioned in the preceding work [11]. Also, the theoretical result gives no explanation of the large performance difference in the co-authoring network. In future work, we will inves-

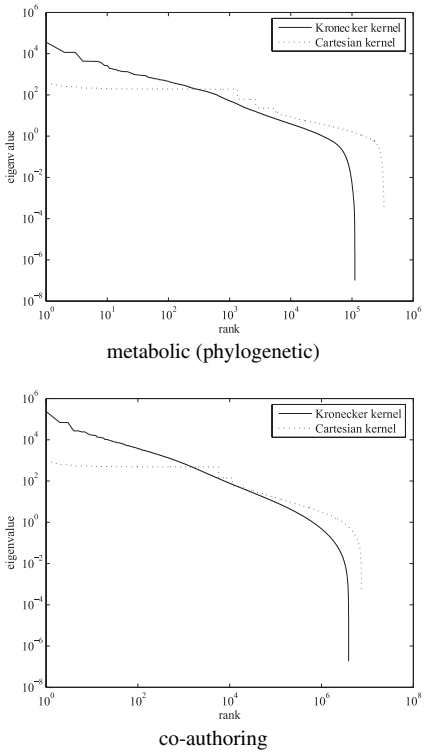

Fig. 5 Eigenvalues of the Kronecker kernel matrices and the Cartesian kernel matrices.

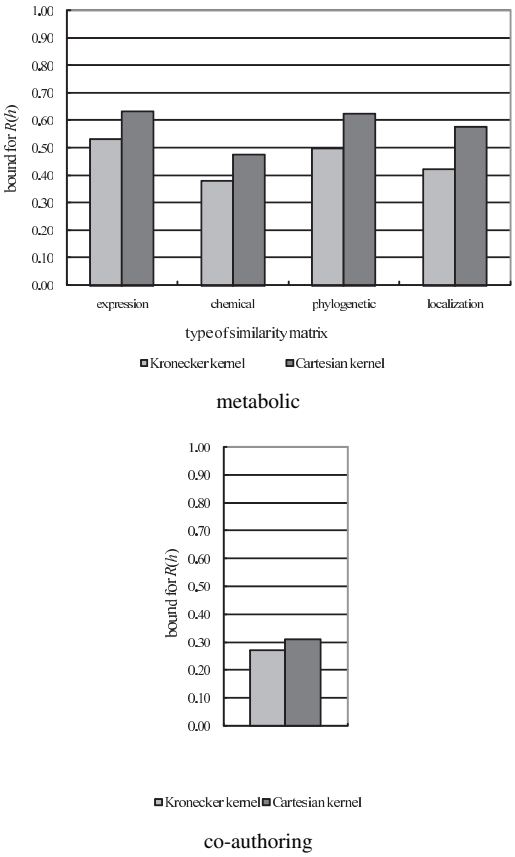

Fig. 6 Bounds of the expected risks for the Kronecker kernel and the Cartesian kernel.

tigate tighter bounds including the several possibilities for improvements mentioned in the preceding work [11].

8. Conclusion

We proposed a new pairwise kernel called the *Cartesian kernel* as a more efficient alternative to the existing pairwise kernel called the Kronecker kernel. While the existing pairwise kernel matrix is defined as a Kronecker product of two instance-wise kernel matrices, the proposed kernel is defined as a Kronecker sum of the two matrices, and therefore it needs much less memory to store the kernel matrix and less computational time than the Kronecker kernel. We also investigated the generalization bounds for the two pairwise kernels and gave also their N-wise extensions. Experimental results showed the proposed kernel is much faster than the Kronecker kernel, and at the same time, competitive with the predictive performance of the Kronecker kernel.

Although we focused on pairwise link prediction based only on node information represented as kernel matrices, there are several methods that utilize only structural information such as link metrics [22] and matrix factorization [23], [24]. One of the future work is to combine both of node information and topological information, like [25], [26].

Our discussion in this paper was mainly in the context of link prediction, while the pairwise prediction problem has been studied in at least three contexts, link prediction, collaborative filtering [27], and multi-task learning [28]. Application to those domains is an interesting future direction.

References

[1] T. Kato, K. Tsuda, and K. Asai, "Selective integration of multiple biological data for supervised network inference," Bioinformatics, vol.21, no.10, pp.2488–2495, 2005.
[2] J.P. Vert and Y. Yamanishi, "Supervised graph inference," Advances in Neural Information Processing Systems 15, 2005.
[3] Y. Yamanishi, J.P. Vert, and M. Kanehisa, "Supervised enzyme network inference from the integration of genomic data and chemical information," Bioinformatics, vol.21, pp.i468–i477, 2005.
[4] L. Getoor and C.P. Diehl, "Link mining: A survey," SIGKDD Explorations, vol.7, no.2, pp.3–12, 2005.
[5] J. Shawe-Taylor and N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press New York, NY, 2004.
[6] J. Basilico and T. Hofmann, "Unifying collaborative and content-based filtering," Proc. 21st International Conference on Machine Learning (ICML), 2004.
[7] A. Ben-Hur and W.S. Noble, "Kernel methods for predicting protein-protein interactions," Bioinformatics, vol.21, no.Suppl.1, pp.i38–i46, 2005.
[8] S. Oyama and C.D. Manning, "Using feature conjunctions across examples for learning pairwise classifiers," Proc. 15th European Conference on Machine Learning (ECML), pp.322–333, 2004.
[9] T. Evgeniou, C. Micchelli, and M. Pontil, "Learning multiple tasks with kernel methods," J. Machine Learning Research, vol.6, no.1, p.615, 2006.
[10] E. Bonilla, F. Agakov, and C. Williams, "Kernel multi-task learning using task-specific features," 11th International Conference on Artificial Intelligence and Statistics (AISTATS), 2007.
[11] B. Schölkopf, J. Shawe-Taylor, A. Smola, and R. Williamson, "Generalization bounds via eigenvalues of the gram matrix," Tech. Rep. 99-035, NeuroColt, 1999.
[12] R. Kroon, Support vector machines, generalization bounds and transduction, Masters Thesis, Stellenbosch University, 2003.
[13] A.J. Laub, Matrix Analysis for Scientists and Engineers, Society for Industrial and Applied Mathematics, 2005.
[14] W. Imrich and S. Klavzar, Product Graphs: Structure and Recognition, Wiley, 2000.
[15] M. Kanehisa, S. Goto, S. Kawashima, Y. Okuno, and M. Hattori, "The KEGG resources for deciphering the genome," Nucleic Acids Research, vol.32, pp.D277–D280, 2004.
[16] C. von Mering, R. Krause, B. Snel, M. Cornell, S. Olivier, S. Fields, and P. Bork, "Comparative assessment of large-scale data sets of protein-protein interactions," Nature, vol.417, pp.399–403, 2002.
[17] K. Tsuda and W.S. Noble, "Learning kernels from biological networks by maximizing entropy," Bioinformatics, vol.20, no.Suppl.1, pp.i326–i333, 2004.
[18] Y. Yamanishi, M. Araki, A. Gutteridge, W. Honda, and M. Kanehisa, "Prediction of drug-target interaction networks from the integration of chemical and genomic spaces," Bioinformatics, vol.24, pp.i232–i240, 2008.
[19] M. Hattori, Y. Okuno, S. Goto, and M. Kanehisa, "Development of a chemical structure comparison method for integrated analysis of chemical and genomic information in the metabolic pathways," J. American Chemical Society, vol.125, pp.11853–11865, 2003.
[20] T. Smith and M. Waterman, "Identification of common molecular subsequences," J. Molecular Biology, vol.147, pp.195–197, 1981.
[21] K. Ishibashi, K. Hatano, and M. Takeda, "Online learning of approximate maximum p-norm margin classifiers with biases," Proc. 21st Annual Conference on Learning Theory (COLT), 2008.
[22] D. Liben-Nowell and J. Kleinberg, "The link prediction problem for social networks," Proc. 12th International Conference on Information and Knowledge Management (CIKM), pp.556–559, 2004.
[23] D. Lee and H. Seung, "Algorithms for non-negative matrix factorization," Advances in Neural Information Processing Systems 13, 2001.
[24] N. Srebro, J. Rennie, and T. Jaakkola, "Maximum-margin matrix factorization," Advances in Neural Information Processing Systems 17, 2005.
[25] M.A. Hasan, V. Chaoji, S. Salem, and M. Zaki, "Link prediction using supervised learning," Workshop on Link Discovery: Issues, Approaches and Applications (LinkKDD), 2005.
[26] J. O'Madadhain, J. Hutchins, and P. Smyth, "Prediction and ranking algorithms for event-based network data," SIGKDD Explorations, vol.7, no.2, pp.23–30, 2005.
[27] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An open architecture for collaborative filtering of Netnews," The Conference on Computer Supported Cooperative Work (CSCW), pp.175–186, 1994.
[28] R. Caruana, "Multitask learning," Mach. Learn., vol.28, no.1, pp.41–75, 1997.

Hisashi Kashima is an associate professor of Department of Mathematical Informatics, the University of Tokyo. Before joining to the faculty, he was a research staff member of Data Analytics Group in Tokyo Research Laboratory of IBM Research during 1999–2009. His research interest includes machine learning and data mining. He received his B.Eng., M.Eng., and Ph.D. degrees from Kyoto University in 1997, 1999, and 2007, respectively.

Satoshi Oyama is an associate professor in the Graduate School of Information Science and Technology, Hokkaido University, Japan. He has been working on machine learning and data mining and their applications to the Web, including domain-specific search, relation discovery, and object identification. He received his B.Eng., M.Eng., and Ph.D. degrees from Kyoto University in 1994, 1996, and 2002, respectively. He was a research fellow of the Japan Society for the Promotion of Science from 2001 to 2002. He was an assistant professor in the Graduate School of Informatics at Kyoto University from 2002 to 2009. He was a visiting assistant professor in the Department of Computer Science at Stanford University from 2003 to 2004.

Yoshihiro Yamanishi is a faculty member at Centre for Computational Biology, Mines ParisTech, France. He is also a researcher in the department of Bioinformatics and Computational Systems Biology of Cancer, Mines ParisTech - Institut Curie - INSERM U900. He is working on statistics and machine learning for bioinformatics, chemoinformatics, and genomic drug discovery. He obtained his B.S. degree in 1999 and M.S. degree in 2001 from Okayama University, and Ph.D in 2005 from Kyoto University in Japan. He was a post-doctoral research fellow at Center for Geostatistics, Ecole des Mines de Paris from 2005 to 2006. He was an assistant professor at Institute for Chemical Research, Kyoto University from 2006 to 2007.

Koji Tsuda is Senior Research Scientist at AIST Computational Biology Research Center. After completing his Dr.Eng. in Kyoto University in 1998, he joined former Electrotechnical Laboratory (ETL), Tsukuba, Japan, as Research Scientist. When ETL is reorganized as AIST in 2001, he joined newly established Computational Biology Research Center, Tokyo, Japan. In 2000–2001, he worked at GMD FIRST (current Fraunhofer FIRST) in Berlin, Germany, as Visiting Scientist. In 2003–2004 and 2006–2008, he worked at Max Planck Institute for Biological Cybernetics, Tuebingen, Germany, first as Research Scientist and later as Project Leader. His scientific interests are in the fields of machine learning, data mining, kernel methods and bioinformatics.