On Pairwise Kernels: An Efficient Alternative and Generalization Analysis

Hisashi Kashima[1], Satoshi Oyama[2],
Yoshihiro Yamanishi[3], and Koji Tsuda[4]

[1] IBM Research, Tokyo Research Laboratory,
[2] Kyoto University, Graduate School of Informatics
[3] Mines ParisTech, Centre for Computational Biology
[4] Max Planck Institute for Biological Cybernetics

Abstract. 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 become successful in various fields. In this paper, we propose an efficient alternative which we call *Cartesian kernel*. While the existing pairwise kernel (which we refer to as 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. Experimental results show the Cartesian kernel is much faster than the existing pairwise kernel, and at the same time, competitive with the existing pairwise kernel in predictive performance. We discuss the generalization bounds by the two pairwise kernels by using eigenvalue analysis of the kernel matrices.
Keywords: 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. Models for pairwise prediction should take a pair of instances as its input, and output the relationship between the two instances. In this paper, we focus on pairwise classification problem, where the task is to predict whether or not a relation exists between given two nodes, and we apply kernel methods [1] to this problem. To apply kernel methods to pairwise classification, we need to define a kernel function between two pairs of instances. Interestingly, three research groups have independently proposed an exactly same pairwise kernel by combining two instance-wise kernel functions [2–4]. The proposed pairwise kernel matrix is considered as a Kronecker product of two instance-wise kernel matrices. However, the pairwise kernel is 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 its applications.

In this paper, we propose a new pairwise kernel called *Cartesian kernel* as a more efficient alternative to the existing pairwise kernel (which we refer to as *Kronecker kernel*). The proposed kernel is defined as a Kronecker sum of two instance-wise kernel matrices, and therefore more computational- and space-efficient than the existing pairwise kernel. The experimental results using numbers of real network data show that the proposed pairwise kernel is much faster than the existing pairwise kernel, and at the same time, competitive with the existing pairwise kernel in predictive performance. Finally, we give the generalization bounds of the two pairwise kernels by using eigenvalue analysis of the kernel matrices [5, 6].

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 [2–4]. The standard binary classification problem aims to 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 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 value. In the context of link prediction, the F can be regarded as the adjacency matrix for a network including $V^{(1)}$ and $V^{(2)}$ as its nodes. The $[\mathsf{F}]_{i_1,i_2} = +1$ indicates that there is a link between $v^{(1)}_{i_1} \in V^{(1)}$ and $v^{(2)}_{i_2} \in V^{(2)}$, the $[\mathsf{F}]_{i_1,i_2} = -1$ indicates that there is no link, and $[\mathsf{F}]_{i_1,i_2} = 0$ 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. $V^{(1)} = V^{(2)} := V$, the F is considered as a $|V| \times |V|$ adjacency matrix F for a set $V = (v_1, v_2, \ldots, v_{|V|})$. If the network is undirected, the F becomes symmetric. If the network is directed, the F is asymmetric, and $[\mathsf{F}]_{i_1,i_2}$ 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 $\mathsf{K}^{(1)}$ and $\mathsf{K}^{(2)}$ for $V^{(1)}$ and $V^{(2)}$, respectively. In exchangeable cases, $\mathsf{K}^{(1)} = \mathsf{K}^{(2)} := \mathsf{K}$. Note that those kernel matrices are positive semi-definite.

Since we are interested in classification of pairs of instances, we need a kernel function between two pairs of instances if we apply kernel methods [1] to this problem. In many case, it is rather easy to design kernels for two basic instances, so we 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 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 to define the pairwise similarity as the product of two instance-wise similarities as

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

Since products of Mercer kernels are also Mercer kernels [1], 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}) is symmetrized as

$$k_\otimes^{\mathsf{SYM}}((v_{i_1}, v_{i_2}), (v_{j_1}, v_{j_2})) = [\mathsf{K}]_{i_1,j_1}[\mathsf{K}]_{i_2,j_2} + [\mathsf{K}]_{i_1,j_2}[\mathsf{K}]_{i_2,j_1}. \tag{2}$$

The prediction of a kernel machine for a pair $(v_{i_1}^{(1)}, v_{i_2}^{(2)})$ is given as $[\mathsf{F}]_{i_1,i_2} = \sum_{(j_1,j_2)} \alpha(v_{j_1}^{(1)}, v_{j_2}^{(2)}) k_\otimes((v_{i_1}^{(1)}, v_{i_2}^{(2)}), (v_{j_1}^{(1)}, v_{j_2}^{(2)}))$, where αs are the model parameters of the kernel machine. In exchangeable and symmetric cases, it becomes $[\mathsf{F}]_{i_1,i_2} = \sum_{(j_1,j_2):j_1<j_2} \alpha(v_{j_1}, v_{j_2}) k_\otimes^{\mathsf{SYM}}((v_{i_1}, v_{i_2}), (v_{j_1}, v_{j_2}))$.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 [7] of two instance-wise kernel matrices as $\mathsf{K}_\otimes = \mathsf{K}^{(2)} \otimes \mathsf{K}^{(1)}$, where its $(v_{i_1}^{(1)} + v_{i_2}^{(2)}|V^{(2)}|), v_{j_1}^{(1)} + v_{j_2}^{(2)}|V^{(2)}|)$-th element is $k_\otimes((v_{i_1}^{(1)}, v_{i_2}^{(2)}), (v_{j_1}^{(1)}, v_{j_2}^{(2)}))$. The pairwise kernel matrix can be interpreted as a weighted adjacency matrix of the Kronecker product graph [8] of the two graphs whose weighted adjacency matrices are the instance-wise kernel matrices. Therefore, we refer to this pairwise kernel as *Kronecker kernel* to distinguish from the one we will propose in the next section.

3 Cartesian kernel: a new pairwise kernel

In this section, we propose a more efficient pairwise kernel. At the end of the previous section, we mentioned the relationship between the existing pairwise kernel and a Kronecker product graph. So, it is natural to imagine that we can design another pairwise kernel based on another kind of product graph. In this paper, we adopt another kind of product graph called Cartesian product graph [8]. 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 $(v_{i_1}^{(1)}, v_{i_2}^{(2)})$ and $(v_{j_1}^{(1)}, v_{j_2}^{(2)})$ 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 $v_{i_1}^{(1)}$ and $v_{j_1}^{(1)}$ in $G^{(1)}$ and there is a link between $v_{i_2}^{(2)}$ and $v_{j_2}^{(2)}$ in $G^{(2)}$. On the other hand, in Cartesian product graphs, a link between these two pairs exists if and only if $v_{i_1}^{(1)} = v_{j_1}^{(1)}$ in $G^{(1)}$ and there is a link between $v_{i_2}^{(2)}$ and $v_{j_2}^{(2)}$ in $G^{(2)}$, or there is a link between $v_{i_1}^{(1)}$ and $v_{j_1}^{(1)}$ in $G^{(1)}$ and $v_{i_2}^{(2)} = v_{j_2}^{(2)}$ in $G^{(2)}$. We can notice that the condition for a link existing in Cartesian product graphs is more strict than that 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

$$k_\otimes((v_{i_1}^{(1)}, v_{i_2}^{(2)}), (v_{j_1}^{(1)}, v_{j_2}^{(2)})) = [\mathsf{K}^{(1)}]_{i_1,j_1}\delta(i_2 = j_2) + \delta(i_1 = j_1)[\mathsf{K}^{(2)}]_{i_2,j_2}, \tag{3}$$

where δ is an indicator function, which returns 1 when its argument is true and 0 otherwise. Since δ is considered as an identity kernel, the above 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}) is symmetrized as

$$\begin{aligned} k_\otimes^{\mathsf{SYM}}((v_{i_1}, v_{i_2}), (v_{j_1}, v_{j_2})) = {} & [\mathsf{K}]_{i_1,j_1}\delta(i_2 = j_2) + \delta(i_1 = j_1)[\mathsf{K}]_{i_2,j_2}] \\ & + [\mathsf{K}]_{i_1,j_2}\delta(i_2 = j_1) + \delta(i_1 = j_2)[\mathsf{K}]_{i_2,j_1}. \end{aligned} \tag{4}$$

The kernel matrix of the Cartesian kernel is equivalently written as the Kronecker *sum* [7] of two instance-wise kernel matrices as $\mathsf{K}_{\oplus} = \mathsf{K}^{(2)} \oplus \mathsf{K}^{(1)}$, where the Kronecker sum operation is defined as $\mathsf{K}^{(2)} \oplus \mathsf{K}^{(1)} = \mathsf{K}^{(2)} \otimes \mathsf{I} + \mathsf{I} \otimes \mathsf{K}^{(1)}$. At the first sight, the size of the Cartesian kernel matrix is the same as that of the Kronecker kernel. But, 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 mention computational efficiency of the Cartesian kernel. While the Kronecker kernel can give a score greater than 0 between arbitrary two pairs, the Cartesian kernel can give a non-zero value only to the pairs which share one of their instances. This fact indicates that the Cartesian kernel is much faster than the Kronecker kernel.

4 Experiments

In this section, we show several experimental results in which we compare the Kronecker kernel and the Cartesian kernel.

We used three data sets for network prediction. Two of them are biological networks, and one of them is a social network. All data are symmetric networks. The first data set [9] contains the metabolic pathway network of the yeast S. Cerevisiae in the KEGG/PATHWAY database [10]. 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[5]. The second data set is a protein-protein interaction network constructed by von Mering et al. [11]. We followed Tsuda and Noble [12], 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[6]. 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[7]. All of the element-wise kernel matrices are normalized so that all of their diagonals are 1. The models were trained by using PUMMA [13], an on-line learning algorithm whose solutions asymptotically converge to those by the support vector machine with squared hinge loss. The hyperparameter for regularization was set to $C = 1$. All of the training data were processed thee times in the training phase. The results were evaluated in AUC by 5-fold cross validation with 20% of all of the pairwise relationships as training data.

Now, we show the predictive performances and execution times by the two pairwise kernels. In Fig. 1, the gray bars indicate the AUCs by the Kronecker kernel, and the

[5] Available at http://web.kuicr.kyoto-u.ac.jp/supp/yoshi/ismb05/.

[6] Available at http://noble.gs.washington.edu/proj/maxent/.

[7] Available at http://ai.stanford.edu/˜gal/data.html.

black bars represent the AUCs by the Cartesian kernel. The error bars indicate the standard deviations of the AUC values. In the upper figure of 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 the lower figures, the left pair of the AUC bars is for the protein-protein interaction networks, and the right one is for the co-authoring network. We can observe 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 in the co-authoring network is not clear, and we could not find the reason in the theoretical analysis in the following section. But it might be relaled to network sparsity (the co-authoring network is the most sparse), or differences between natures of biological networks and social networks. Figure 2 shows the average training time for each data set in log scale. We can see that the Cartesian kernel is at most 16 times faster than the Kronecker kernel. The differences are remarkable when the network size is large. Based on the above results, we conclude that the Cartesian kernel is a promising alternative to the Kronecker kernel especially for large data sets.

5 Generalization bounds for the pairwise kernels

In this section, we discuss generalization bounds for the Kronecker kernel and the Cartesian kernel. 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 [5, 6]. To compute the generalization bounds, we need to compute the eigenvalues of the Kronecker product $\mathsf{K}_\otimes$ or the Kronecker sum $\mathsf{K}_\oplus$ of the two instance-wise kernel matrices. It is difficult to directly compute the eigenvalues for the Kronecker product or the Kronecker sum, since their size are very huge.However, we can compute them from the eigenvalues of the instance-wise kernel matrices by using the following theorem [7].

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

Figure 3 (left) shows the eigenvalues of the Kronecker kernel matrix and the Cartesian kernel matrix derived from phylogenetic profiles for the KEGG metabolic network. Also, Figure 3 (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. 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 $R(h) = \sum_{(x,y)\in Z} \delta(h(x)y \leq 0)P(x, y)$. Given a

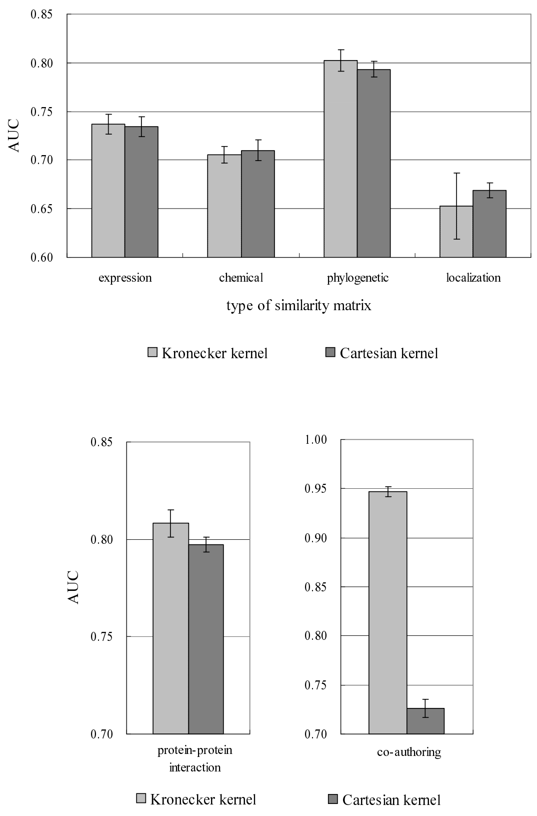


Fig. 1. Summary of results for the KEGG metabolic network (upper), the protein-protein interaction network (lower-left) and the co-authoring network (lower-right).

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 [6] gives an upper bound for the expected risk.

Theorem 2 *Let $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_M$ be the set of eigenvalues of the kernel matrix derived from the set of all possible data points. Consider the hypothesis class $\mathcal{F}(c)_B = \{\langle w, x\rangle + b : \|w\| \leq c, |b| \leq B\}$. Then the following inequality holds simultaneously for all $\gamma \in (8\Upsilon(n), 1]$:*

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

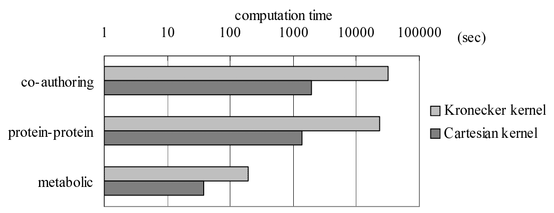
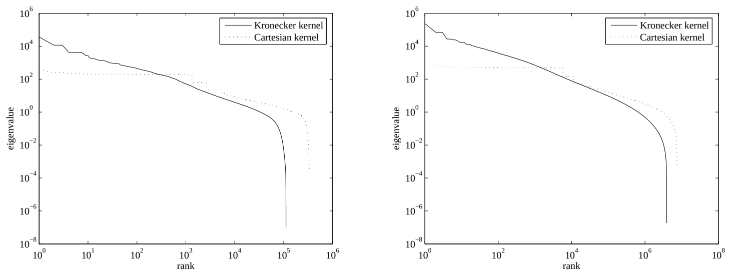


Fig. 2. Comparison of computation time.



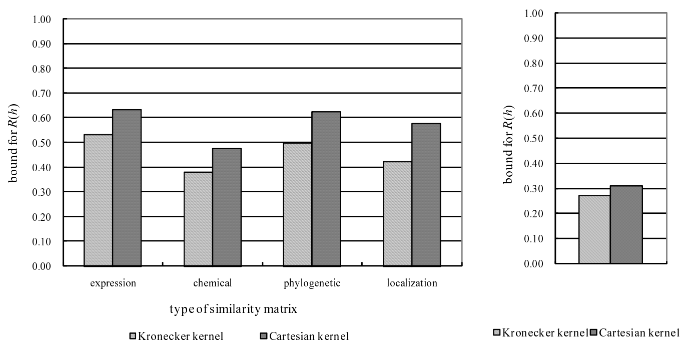
metabolic (phylogenetic) co-authoring

Fig. 3. Eigenvalues of the Kronecker kernel matrices and the Cartesian kernel matrices.

where

$$\begin{aligned}
\Upsilon(n) &= \min_{j\in\{1,\ldots,n-1\}} 6\cdot 2^{-\frac{(j-1)}{k(2^j-1)}}\left(\lambda_1,\cdots,\lambda_{k(2^j-1)}\right)^{\frac{1}{2k(2^j-1)}} c(n,j)\\
k(l) &= \min\left\{k\in\{1,\ldots,M\} : \lambda_{k+1}\le\left(\lambda_1\cdots\lambda_k l^2\right)^{\frac{1}{k}}\right\}\\
c(n,j) &= \min\left(1, 1.86\sqrt{\log_2\left(M/(n-j)+1\right)/(n-j)}\right)
\end{aligned}\quad .$$

The generalization bounds computed for the metabolic network and the co-authoring network are shown in Fig. 4. The bounds for the Kronecker kernels are smaller than the bounds for the Cartesian kernels. Those theoretical results are consistent with the experimental results in the previous section, 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. As mentioned in the preceding work [5], the bounds are not very tight. Also, the theoretical result gives no explanation of the large performance difference in the co-authoring network. In future work, we will investigate tighter bounds including the several possibilities for improvements mentioned in the preceding work [5].



metabolic co-authoring

Fig. 4. Bounds of the expected risks for the Kronecker kernel and the Cartesian kernel.

References

1. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press New York, NY, USA (2004)
2. Basilico, J., Hofmann, T.: Unifying collaborative and content-based filtering. In: Proceedings of the 21st International Conference on Machine Learning (ICML). (2004)
3. Ben-Hur, A., Noble, W.S.: Kernel methods for predicting protein-protein interactions. Bioinformatics **21**(Suppl. 1) (2005) i38–i46
4. Oyama, S., Manning, C.D.: Using feature conjunctions across examples for learning pairwise classifiers. In: Proceedings of the 15th European Conference on Machine Learning (ECML). (2004) 322–333
5. Schölkopf, B., Shawe-Taylor, J., Smola, A., Williamson, R.: Generalization bounds via eigenvalues of the gram matrix. Technical Report 99-035, NeuroColt (1999)
6. Kroon, R.: Support vector machines, generalization bounds and transduction, masters thesis, stellenbosch university (2003)
7. Laub, A.J.: Matrix Analysis for Scientists and Engineers. Society for Industrial and Applied Mathematics (2005)
8. Imrich, W., Klavzar, S.: Product Graphs: Structure and Recognition. Wiley (2000)
9. Yamanishi, Y., Vert, J.P., Kanehisa, M.: Supervised enzyme network inference from the integration of genomic data and chemical information. Bioinformatics **21** (2005) i468–i477
10. Kanehisa, M., Goto, S., Kawashima, S., Okuno, Y., Hattori, M.: The KEGG resources for deciphering the genome. Nucleic Acids Research **32** (2004) D277–D280
11. von Mering, C., Krause, R., Snel, B., Cornell, M., Olivier, S., Fields, S., Bork, P.: Comparative assessment of large-scale data sets of protein-protein interactions. Nature **417** (2002) 399–403
12. Tsuda, K., Noble, W.S.: Learning kernels from biological networks by maximizing entropy. Bioinformatics **20**(Suppl. 1) (2004) i326–i333
13. Ishibashi, K., Hatano, K., Takeda, M.: Online learning of approximate maximum p-norm margin classifiers with biases. In: Proceedings of the 21st Annual Conference on Learning Theory (COLT 2008). (2008)