Information Theoretic Kernel Integration

Yiming Ying†, Kaizhu Huang‡, and Colin Campbell†
†Department of Engineering Mathematics, University of Bristol,
Bristol BS8 1TR, United Kingdom
‡National Laboratory of Pattern Recognition, Institute of Automation,
The Chinese Academy of Sciences, 100190 Beijing, China

Abstract

In this paper we consider a novel information-theoretic approach to multiple kernel learning based on minimising a Kullback-Leibler (KL) divergence between the output kernel matrix and the input kernel matrix. There are two formulations which we refer to as *MKLdiv-dc* and *MKLdiv-conv*. We propose to solve MKLdiv-dc by a difference of convex (DC) programming method and MKLdiv-conv by a projected gradient descent algorithm. The effectiveness of the proposed approaches is evaluated on a benchmark dataset for protein fold recognition and a yeast protein function prediction problem.

1 Information-theoretic Data Integration

In this paper we consider the problem of integrating multiple data sources using a kernel-based approach. Recent trends in learning kernel combination are usually based on the margin maximization criterion used by Support Vector Machines (SVMs) or variants [5, 8, 9, 10, 14, 16, 17]. There, each data source can be represented by $\mathbf{x}^\ell = \{x_i^\ell : i \in \mathbb{N}_n\}$ for $\ell \in \mathbb{N}_m$ and the outputs are similarly denoted by $\mathbf{y} = \{y_i : i \in \mathbb{N}_n\}$. With kernel methods, for any $\ell \in \mathbb{N}_m$, each ℓ-th data source can be encoded into a candidate kernel matrix denoted by $\mathbf{K}_\ell = (K_\ell(x_i^\ell, x_j^\ell))_{ij}$. Depending on the type of data source used, the candidate kernel function $\mathbf{K}_\ell$ would be specified a priori, as a graph kernel for graph data or a string kernel for sequence data, for example. The composite kernel matrix is given by $\mathbf{K}_\lambda = \sum_{\ell\in\mathbb{N}_m} \lambda_\ell \mathbf{K}_\ell$. Hence, in this context the problem of data integration is reduced to learning a convex combination of candidate kernel matrices with *kernel coefficients* or *weights* denoted by λ.

We can quantify the similarity between $\mathbf{K}_\lambda$ and the output kernel $\mathbf{K}_\mathbf{y}$ through a Kullback-Leibler (KL) divergence or relative entropy term [3, 6, 7, 12, 13]. There is a simple bijection between the set of distance measures in these data spaces and the set of zero-mean multivariate Gaussian distributions [3]. Using this bijection, the difference between two distance measures, parameterized by $\mathbf{K}_\lambda$ and $\mathbf{K}_\mathbf{y}$, can be quantified by the relative entropy or Kullback-Leibler (KL) divergence between the corresponding multivariate Gaussians. Kernel matrices are generally positive semi-definite and thus can be regarded as the covariance matrices of the Gaussian distributions. Matching kernel matrices $\mathbf{K}_\lambda$ and $\mathbf{K}_\mathbf{y}$, can therefore be realized by minimizing a KL divergence between these two distributions. As described in [3, 6, 13], the Kullback-Leibler (KL) divergence (relative entropy) between a Gaussian distribution $\mathcal{N}(0, \mathbf{K}_\mathbf{y})$ with the output covariance matrix $\mathbf{K}_\mathbf{y}$ and a Gaussian distribution $\mathcal{N}(0, \mathbf{K}_\mathbf{x})$ with the input kernel covariance matrix $\mathbf{K}_\mathbf{x}$ is defined by

$$\mathrm{KL}\big(\mathcal{N}(0,\mathbf{K}_\mathbf{y})||\mathcal{N}(0,\mathbf{K}_\mathbf{x})\big) := \frac{1}{2}\mathrm{Tr}(\mathbf{K}_\mathbf{y}\mathbf{K}_\mathbf{x}^{-1}) + \frac{1}{2}\log|\mathbf{K}_\mathbf{x}| - \frac{1}{2}\log|\mathbf{K}_\mathbf{y}| - \frac{n}{2}. \tag{1}$$

Here, the notation $\mathrm{Tr}(\mathbf{B})$ denotes its trace. Though $\mathrm{KL}\big(\mathcal{N}(0,\mathbf{K}_\mathbf{y})||\mathcal{N}(0,\mathbf{K}_\mathbf{x})\big)$ is non-convex w.r.t. $\mathbf{K}_\mathbf{x}$, it has a unique minimum at $\mathbf{K}_\mathbf{x} = \mathbf{K}_\mathbf{y}$ if $\mathbf{K}_\mathbf{y}$ is positive definite, suggesting that minimizing the above KL-divergence encourages $\mathbf{K}_\mathbf{x}$ to approach $\mathbf{K}_\mathbf{y}$. If the input kernel matrix $\mathbf{K}_\mathbf{x}$ is represented by a linear combination of m candidate kernel matrices, i.e. $\mathbf{K}_\mathbf{x} = \mathbf{K}_\lambda = \sum_{\ell\in\mathbb{N}_m} \lambda_\ell \mathbf{K}_\ell$, the above

	MKLdiv-dc	MKLdiv-conv	SimpleMKL	VBKC	MKL-RKDA
All data sources	$\mathbf{73.36}$	$\mathbf{71.01}$	66.57	$\mathbf{68.1 \pm 1.2}$	$\mathbf{68.40}$
Uniform weighted	68.40	68.40	$\mathbf{68.14}$	$-$	66.06

Table 1: Performance with individual and all data sources. The results of VBKC are cited from [2]. The results not employed there are denoted by '$-$'. The best result for each kernel learning method is marked in bold.

KL-divergence based kernel learning is reduced to the following formulation:

$$\begin{aligned}&\arg\min_{\lambda\in\triangle} \text{KL}\left(\mathcal{N}(0,\mathbf{K_y})||\mathcal{N}(0,\mathbf{K}_\lambda)\right)\\ &= \arg\min_{\lambda\in\triangle} \text{Tr}\left(\mathbf{K_y}(\textstyle\sum_{\ell\in\mathbb{N}_m}\lambda_\ell\mathbf{K}_\ell+\sigma\mathbf{I}_n)^{-1}\right)+\log\left|\textstyle\sum_{\ell\in\mathbb{N}_m}\lambda_\ell\mathbf{K}_\ell+\sigma\mathbf{I}_n\right|,\end{aligned} \tag{2}$$

where $\mathbf{I}_n$ denotes the $n\times n$ identity matrix and $\sigma>0$ is a supplemented small parameter to avoid the singularity of $\mathbf{K}_\lambda$.

Since the KL-divergence is not symmetric with respect to $\mathbf{K_y}$ and $\mathbf{K}_\lambda$, another alternative approach to matching kernel matrices is given by

$$\begin{aligned}&\arg\min_{\lambda\in\triangle} \text{KL}\left(\mathcal{N}(0,\mathbf{K}_\lambda)||\mathcal{N}(0,\mathbf{K_y})\right)\\ &= \arg\min_{\lambda\in\triangle} \textstyle\sum_{\ell\in\mathbb{N}_m}\lambda_\ell\text{Tr}\left((\mathbf{K_y}+\sigma\mathbf{I}_n)^{-1}\mathbf{K}_\ell\right)-\log\left|\textstyle\sum_{\ell\in\mathbb{N}_m}\lambda_\ell\mathbf{K}_\ell+\sigma\mathbf{I}_n\right|,\end{aligned} \tag{3}$$

where parameter $\sigma>0$ is to avoid the singularity of $\mathbf{K_y}$. If there is no positive semi-definiteness restriction over $\mathbf{K}_\ell$, this formulation is a well-known convex *maximum-determinant problem*. Define $g(\lambda):=-\log\left|\sum_{\ell\in\mathbb{N}_m}\lambda_\ell\mathbf{K}_\ell+\sigma\mathbf{I}_n\right|$ and $f(\lambda):=\text{Tr}\left(\mathbf{K_y}(\sum_{\ell\in\mathbb{N}_m}\lambda_\ell\mathbf{K}_\ell+\sigma\mathbf{I}_n)^{-1}\right)$. Since from [1] functions $-\log|\mathbf{C}|$ and $\text{Tr}\left(\mathbf{K_y}\mathbf{C}^{-1}\right)$ are convex with respect to positive semi-definite matrices $\mathbf{C}$. Then it is easy to see that both f and g are convex with respect to $\lambda\in\triangle$. Consequently, problem (3) is convex and problem (2) is a *difference of convex problem*. For the convex problem (2) we can follow a projected gradient descent procedure and hence we refer to problem (2) as *MKLdiv-conv*. For the difference of convex (DC) problem (3) we use a concave-convex procedure [18], and hence we refer to problem (3) as *MKLdiv-dc*. Full details are given in our paper [15].

The KL-divergence criterion for kernel integration was also successfully used in [13, 7] which formulated the problem of supervised network inference as a kernel matrix completion problem. In terms of information geometry, formulation (2) corresponds to finding the m-projection of $\mathbf{K_y}$ over an e-flat submanifold. The convex problem (3) can be regarded as finding the e-projection of $\mathbf{K_y}$ over a m-flat submanifold. The formulation (2) also has a close relation with *Gaussian Process regression* [11]. Specifically, the Gaussian process prior can be written as $\mathbf{f}|\mathbf{x}\sim\mathcal{N}(\mathbf{f}|\,0,\mathbf{K}_\lambda)$. For the output vector $\mathbf{y}\in\mathbb{R}^n$, if we let $\mathbf{K_y}=\mathbf{y}\mathbf{y}^\top$ in the objective function of formulation (2), then one can easily check that, up to a constant term, the objective function in formulation (2) is the negative of the log likelihood of Gaussian process regression.

2 Experimental Evaluation

We evaluated both MKLdiv methods on two bioinformatics tasks: protein fold recognition and yeast protein function prediction. In these tasks, for a C-class classification, we recast the outputs $\mathbf{y}=\{y_i : i\in\mathbb{N}_n\}$ as $(y_{i1},\ldots,y_{iC})$ such that $y_{ip}=1$ if x_i is in class p and otherwise -1. Hence the outputs are represented by an $n\times C$ *indicator matrix* $\mathbf{Y}=(y_{ip})_{i,p}$ whose p-th column vector is denoted by $\mathbf{y}_p$. The output kernel matrix $\mathbf{K_y}=\mathbf{Y}\mathbf{Y}^\top$. Also, we first compute the kernel weights using the MKLdiv methods given above and then feed these into a one-against-all multi-class SVM to make predictions.

2.1 Protein Fold Recognition

We evaluated MKLdiv on a well-known protein fold prediction dataset [4]. This benchmark dataset (based on SCOP PDB-40D) has 27 SCOP fold classes with 311 proteins for training and 383 for testing. There are a total of 12 different data sources such as Amino Acid Composition (C), Predicted Secondary Structure (S), Hydrophobicity (H), Polarity (P), Polarizability (Z), PseAA $\lambda=1$ (L1), PseAA $\lambda=4$ (L4), PseAA $\lambda=14$ (L14), PseAA $\lambda=30$ (L30), SW with BLOSUM62 (SW1) and SW with PAM50 (SW2), etc. As in [2], we employ linear kernels (Smith-Waterman scores) for SW1 and SW2 and second order polynomial kernels for the other data sources. Ding and Duchbak [4] reported an original test set accuracy (TSA) of 56%, subsequently improved to 70% by Damoulas and Girolami [2] using a Bayesian multi-class multi-kernel algorithm. We additionally compare our MKLdiv methods with kernel learning based on a one-against-all multiclass SVM trained using the SimpleMKL software package [10], kernel learning for regularized discriminant analysis (MKL-RKDA) [14] and a probabilistic Bayesian model for kernel learning (VBKC) [2].

In Table 1 we see that the performance of MKLdiv-dc and MKLdiv-conv inclusive of all data sources achieves a test set accuracy of 73.36% and 71.01% respectively, consistently outperforming all individual performances and the uniformly weighted composite kernel (68.40%). Our methods also outperforms SimpleMKL (68.14%) and MKL-RKDA (68.40%). As depicted in the subfigure (b) of Figure 1, the kernel weights of MKLdiv-dc and MKLdiv-conv include some less informative data sources such as PZL1,L4,L14,L30 etc., with small (but not zero) kernel weights. In contrast, as shown in (e) and (g) of Figure 1, SimpleMKL and MKL-RKDA completely discard these less informative data sources. However, as shown in (d) and (f) of Figure 1, SimpleMKL and MKL-RKDA achieve poorer performance, less than 70%, while MKLdiv-dc achieves 73.36% and MKLdiv-conv achieves 71.01%. This suggests that MKLdiv-dc provides a more reasonable balance over the entire set of data sources.

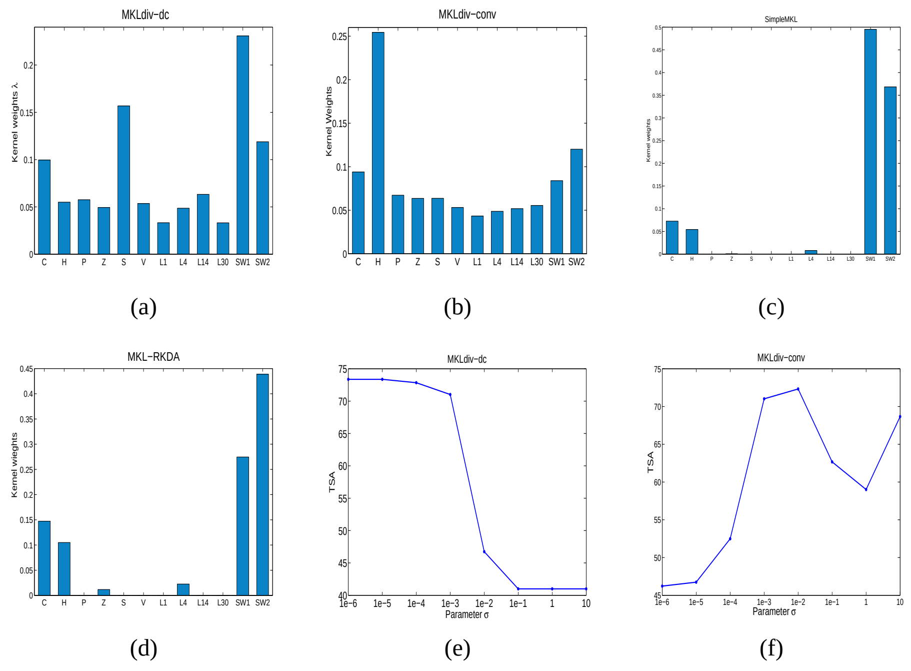

Figure 1: Kernel weights: (a) MKLdiv-dc, (b) MKLdiv-conv, (c) SimpleMKL and (d) MKL-RKDA. Test set accuracy versus different values of σ on the protein fold recognition dataset: (a) MKLdiv-dc and (b) MKLdiv-conv.

Sensitivity against Parameter σ

The initial purpose of introducing σ is to avoid the singularity of the input kernel matrix or the output kernel matrix. However, in practice we found that, in the convex formulation MKLdiv-conv, values of σ have a great influence on performance for protein fold recognition. Subfigures (e)-(f) Figure 1 depicted the test set accuracy versus values of σ where we see in (e) of Figure 1 that the test set accuracy of MKLdiv-dc is relatively stable for small values of σ's compared with MKLdiv-conv. This generally suggests that the parameter σ has a great impact on performance of MKLdiv-conv. This could be because the output kernel matrix $\mathbf{K_y} = \mathbf{YY}^\top$ is of low rank (rank one in binary classification) and thus adding a small matrix $\sigma \mathbf{I}_n$ in the formulation MKLdiv-conv could dramatically change the information of the output kernel matrix.

2.2 Yeast Protein Classification

We next extend our investigation of MKLdiv-dc and MKLdiv-conv to a yeast membrane protein classification problem [8]. This binary classification task has 2316 examples with eight input kernel matrices. MKLdiv-dc yields a ROC score of 0.9189 ± 0.0171 which is competitive with the result in [8]. MKLdiv-conv, however, achieved a ROC score of 0.9016 ± 0.0161 which is worse than MKLdiv-dc. The performance of MKLdiv-dc is also slightly better than the performance of the uniformly weighted kernel 0.9084 ± 0.0177 excluding the noise kernel and 0.8979 ± 0.0120 including the noise kernel.

Conclusion

In this paper we developed a novel information-theoretic approach to learning a linear combination of kernel matrices based on the KL-divergence [13, 7, 12, 6, 3], especially focused on the protein fold recognition problem.

Generally, it is difficult to determine which criterion is better for multiple kernel combination since this problem is highly data-dependent. For the information-theoretic approaches MKLdiv-dc and MKLdiv-conv, although MKLdiv-dc is not convex and its DC algorithm tends to find a local minima, in practice we would recommend MKLdiv-dc for the following reasons. Firstly, as mentioned above MKLdiv-dc has a close relation with the kernel matrix completion problem using information geometry [13, 7] and the maximization of the log likelihood of Gaussian Process regression [11], which partly explains the success of MKLdiv-dc. Secondly, we empirically observed that MKLdiv-dc outperforms MKLdiv-conv in protein fold recognition and yeast protein function prediction. Finally, as we showed in Figure 1, the performance of MKLdiv-conv is quite sensitive to the parameter σ and the choice of σ remains a challenging problem. MKLdiv-dc is relatively stable with respect to small values of σ and we can fix σ to be a very small number e.g. $\sigma = 10^{-5}$.

References

[1] Borwein, J.M. and Lewis, AS: *Convex Analysis and Nonlinear Optimization: Theory and Examples*, CMS Books in Mathematics, Springer-Verlag, New York, 2000.

[2] Damoulas, T. and Girolami, M: Probabilistic multi-class multi-kernel learning: On protein fold recognition and remote homology detection, *Bioinformatics*, **24**(10), 1264-1270, 2008.

[3] Davis, JV, Kulis, B, Jain, P, Sra, S and Dhillon IS: Information-theoretic metric learning. *ICML* ,2007.

[4] Ding, C. and Dubchak, I, Multi-class protein fold recognition using support vector machines and neural networks. *Bioinformatics*, **17**, 349–358, 2001.

[5] Girolami M. and Rogers S: Hierarchic Bayesian Models for Kernel Learning, *ICML*, 2005.

[6] Lawrence, N.D. and Sanguinetti, G, Matching kernels through Kullback-Leibler divergence minimization, *Technical Report CS-04-12*, Department of Computer Science, University of Sheffield ,2005.

[7] Kato, T., Tsuda, K., and Asai, K. Selective integration of multiple biological data for supervised network inference. *Bioinformatics*, **21**: 2488–2495, 2005.

[8] Lanckriet, G.R.G., De Bie T., Cristianini N., Jordan M.I., and Noble WS: A statistical framework for genomic data fusion. *Bioinformatics*, **20**(16): 2626-2635, 2004.

[9] Micchelli CA and Pontil M: Learning the kernel function via regularization, *J. of Machine Learning Research*, **6**: 1099–1125, 2005.

[10] Rakotomamonjy A, Bach F, Canu S, Grandvalet Y: SimpleMKL, *J. of Machine Learning Research*, 2008.

[11] Rasmussen C E and Williams C. *Gaussian Processes for Machine Learning*, the MIT Press, 2006.

[12] Sun L, Ji S, and Ye J: Adaptive diffusion kernel learning from biological networks for protein function prediction, *BMC Bioinformatics*, **9**:162, 2008.

[13] Tsuda, K., Akaho, S., and Asai, K. The em algorithm for kernel matrix completion with auxiliary data. *Journal of Machine Learning Research*, **4**: 67–81 2003.

[14] Ye J, Ji S, and Chen J: Multi-class discriminant kernel learning via convex programming, *J. of Machine Learning Research*, **9**: 719–758, 2008.

[15] Ying, Y., Huang, K. and Campbell, C: Enhanced Protein Fold Prediction through a Novel Data Integration Approach. *BMC Bioinformatics*, 10:267, 2009.

[16] Ying Y and Zhou DX: Learnability of Gaussians with flexible variances, *J. of Machine Learning Research*, **8**: 249–276, 2007.

[17] Ying Y and Campbell C: Generalization bounds for learning the kernel, *COLT*, 2009.

[18] Yuille, A.L. and Rangarajan, A. (2003) The concave convex procedure. *Neural Computation*, **15**: 915–936.