Improved Functional Prediction of Proteins by Learning Kernel Combinations in Multilabel Settings

Volker Roth, Bernd Fischer

ETH Zurich, Institute of Computational Science
Universität-Str. 6, CH-8092 Zurich
{vroth,bernd.fischer}@inf.ethz.ch

1 Introduction

Kernel methods have been successfully applied to a variety of biological data analysis problems. One problem of using kernels, however, is the lacking interpretability of the decision functions. It has been proposed to address this problem by using *multiple* kernels together with some combination rules, where each of the kernels measures different aspects of the data. Methods for learning sparse kernel combinations have the potential to extract *relevant* measurements for a given task. Moreover, the use of multiple kernels addresses the problem of *data fusion* which is inherent in bioinformatics where data are often represented as strings or graphs. Kernels provide a suitable framework for combining such inhomogeneous data under a common matrix representation.

Here we present a method for learning kernel combinations which explicitly addresses the problem of *multilabel classification*. The main ingredient is an extension of nonlinear kernel discriminant analysis to sparse combinations of kernel matrices. The sparsity is obtained by way of *adaptive ridge* (AdR) regression which works as a kernel-specific regularizer.

Existing algorithms for combining kernels recast the problem as a *quadratically constrained quadratic program* (QCQP), [6], as a *semi-infinite linear program* (SLIP), [7], or within a *sequential minimization optimization* (SMO) framework, [1]. Methods for selecting kernel parameters have also been introduced in the context of Gaussian processes. Typical approaches of this kind maximize the marginal likelihood by way of gradient ascent. Kernels operating e.g. on strings, however, are not differentiable due to discrete parameters like the length of substrings. None of these approaches has been extended to handling multiple labels in a consistent way.

Another potential shortcoming of existing methods may be their lacking efficiency, at least for the QCQP formulation. For the latter there exists an $O(n^{4.5})$ algorithm and a $O(n^3)$ approximation, which both seem to impose unacceptable computational costs for large-scale problems. The complexity of both the SLIP- and the SMO method is unclear, but empirical results suggest that they work well in practice. For the proposed AdR method there exists an $O(n^2)$ approximation which can be implemented very efficiently even if not all kernel matrices can be hold in the memory.

The most important extension to existing methods, however, is the direct solution for *multilabel* learning tasks (i.e. problems in which an object can have more than one label). Whereas existing methods usually ignore the multilabel information, the AdR formulation can learn the correlations between classes introduced by objects with multiple labels. We show that this feature leads to an improved prediction of functional classes for yeast proteins.

2 Convex Kernel Combinations in Multilabel Classification

The method presented here is an extension of the *mixture discriminant analysis* (MDA) framework, which forms a link between Gaussian mixture models and discriminant analysis. [4]. The algorithm for solving multilabel classification problems emerges as a special case of this clustering approach.

Consider a Gaussian mixture model with K mixture components which share the covariance matrix Σ. The classical EM-algorithm provides a convenient method for maximizing the data likelihood. Following [4], the M-step can be carried out by linear discriminant analysis (LDA) which uses the "fuzzy labels" estimated in the preceding E-step. LDA is equivalent to an *optimal scoring* problem (cf. [5]), the basic ingredient of which is a linear regression procedure against the class-indicator variables. Since space here precludes a more detailed discussion, we concentrate in the following on the aspect of extending the model to learning combinations of different kernel matrices.

Adaptive ridge penalties and kernelization. A central ingredient of optimal scoring is the "blurred" $(n \times K)$ response matrix $\tilde{Z}$, whose rows consist of the current membership probabilities for each of the n training samples. Given an initial $(K \times K-1)$ *scoring* matrix $\Theta = (\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_K)$, a sequence of $K-1$ linear regression problems is solved by minimizing $\|\tilde{Z}\boldsymbol{\theta}_k - X\boldsymbol{\beta}_k\|_2^2$, where X is the $(n \times d)$ data matrix which contains the data vectors $\{\boldsymbol{x}_i\}_{i=1}^n \in \mathbb{R}^d$ as rows. Taking a Bayesian viewpoint of regression, we specify a prior distribution over the regression coefficients $\boldsymbol{\beta}$. This distribution has the form of an *Automatic Relevance Determination* (ARD) prior: $p(\boldsymbol{\beta}|\boldsymbol{\omega}) \propto \exp[-\sum_{i=1}^d \omega_i \beta_i^2]$. For each regression coefficient, the ARD prior contains a free hyperparameter ω_i, which encodes the "relevance" of the i-th variable in the linear regression. Adaptive ridge regression (AdR) [3] finds the hyper-parameters $\boldsymbol{\omega}$ by requiring that the mean prior variance is proportional to $1/\lambda$: $\frac{1}{d}\sum_{i=1}^d \frac{1}{\omega_i} = \frac{1}{\lambda}\ , \quad \omega_i > 0$, where λ is a predefined regularization constant that is typically selected via cross validation. The balancing procedure has the effect that some hyper-parameters ω_i go to infinity. As a consequence, the coefficients β_i are shrinked to zero and the corresponding input variables are discarded. Following [3] it is numerically advantageous to introduce new variables $\gamma_{j,i} = \sqrt{\omega_i/\lambda}\ \beta_{j,i}\ , \quad c_i = \sqrt{\lambda/\omega_i}$.

Denoting by $D_{\boldsymbol{c}}$ a diagonal matrix with elements c_i, we have to minimize

$$\sum_{k=1}^K \left\| \tilde{Z}\boldsymbol{\theta}_k - XD_{\boldsymbol{c}}\boldsymbol{\gamma}_k \right\|^2 + \lambda \boldsymbol{\gamma}_k^\top \boldsymbol{\gamma}_k \quad \text{s.t.} \quad \boldsymbol{c}^\top \boldsymbol{c} = d, \quad c_i > 0. \qquad (1)$$

Consider now the case of sharing weights over blocks of m regression coefficients: $m \cdot J = d$ (we have J such blocks): $\boldsymbol{c} = (c_1, \dots, c_1, \dots, c_J, \dots, c_J)^\top$.

Note that for given weights $\boldsymbol{c}$, eq. (1) defines a standard ridge-regression problem in the transformed data $X' = XD_{\boldsymbol{c}}$. It is well-known in the literature that the solution vectors $\hat{\boldsymbol{\gamma}}_k$ lie in the span of these input data, i.e. $\hat{\boldsymbol{\gamma}}_k = X'^\top \boldsymbol{\alpha}_k$, which allows us to identify the gram matrix $X'X'^\top$ as a Mercer kernel. Since we have assumed that a weight c_i is shared over a whole block of m features, we can decompose this kernel as a weighted sum of J individual kernels: $X'X'^\top = \sum_{j=1}^J c_j^2 X_{(j)} X_{(j)}{}^\top =: \sum_{j=1}^J c_j^2 K_j$. With this expression we have arrived at the desired framework for learning convex combinations of kernel matrices in each M-step of the EM-algorithm: minimize

$$\sum_{k=1}^{K} \left\| \tilde{Z}\boldsymbol{\theta}_k - (\sum_{j=1}^{J} c_j^2 K_j)\boldsymbol{\alpha}_k \right\|^2 + \lambda \boldsymbol{\alpha}_k^\top (\sum_{j=1}^{J} c_j^2 K_j)\boldsymbol{\alpha}_k \qquad (2)$$

subject to $\boldsymbol{c}^\top \boldsymbol{c} = \sum_{j=1}^{J} c_j^2 = d, \quad c_i > 0$. The minimizing vectors $\hat{\alpha}_k,\ k = 1, \dots, K$ can be found simultaneously in a very efficient way by employing *block conjugate gradient methods*, [2]. The optimal weights $\boldsymbol{c}$ are found iteratively by a fixed-point algorithm similar to that proposed in [3].

Multilabel classification. In multilabel classification problems, an object $\boldsymbol{x}_i$ can belong to more than one class, i.e. it might come with a set of labels Y_i. We treat these multilabels in a probabilistic way by assigning to each observation a set of class-membership probabilities. These probabilities might be given explicitly by the supervisor. Alternatively, they can be estimated uniformly as $1/|Y_i|$, if the class η is a member of the label set Y_i, and zero otherwise. In any case we end up with a set of membership probabilities for each object. Encoding these probabilities in the “blurred response matrix $\tilde{Z}$, we run one single M-step of the above clustering model.

Kernel discriminant analysis is a generative classifier which implicitly models the classes as Gaussians in the kernel feature space. The effect of multilabels on the classifier during the training phase can be understood intuitively as follows. If there are many objects in the training set which belong to both the classes C_i and C_j, the respective class centroids $\mu_{\{1,2\}}$ will be shrunk towards the averaged value $1/2 \cdot (\mu_1 + \mu_2)$. In this way, the classifier can learn the correlation of class labels.

For discriminant analysis it is straightforward to compute for each object a vector of assignment probabilities to the individual classes C_k, see e.g. [5]. In a traditional two-class scenario we would typically assign an object to class C_1 if the corresponding membership probability exceeds 1/2[1]. In multilabel scenarios, however, an object can belong to different classes so that we have to find a suitable way of thresholding the output probabilities. In analogy to the classical two-class case, we propose to first sort the assignment probabilities in decreasing order and assign an object $\boldsymbol{x}_i$ to the first k classes in this order such that $\sum_{j=1}^{k} p^{\text{sorted}}(C_j|\boldsymbol{x}_i) \geq 1/2$.

3 Functional categories of yeast proteins

This experiment concerns the prediction of functional categories associated with yeast proteins. On the top-level hierarchy, the functional catalogue provided by the *MIPS comprehensive yeast genome database* assigns 3588 yeast proteins with known function to 13 classes. Since a protein can have several functions, we recast the prediction problem in a multilabel framework. The proteins are represented by a collection of kernel matrices introduced in [6], consisting of (i) two kernels which analyze the domain structure: K_{pfdom} and an enriched variant $K_{\text{pf\_exp}}$; (ii) three diffusion kernels on interaction graphs: K_{mpi} (protein-protein interactions), K_{mgi} (genetic interactions), K_{tap} (co-participation in protein complexes); (iii) two kernels derived from cell cycle gene expression measurements: $K_{\text{exp\_d}}$ and a RBF variant $K_{\text{exp\_g}}$; (iv) a string alignment kernel K_{SW}. From each of these 8 original kernels we derive 3 additional Gaussian

[1] For simplicity we have ignored the class priors which might give rise to other thresholds.

RBF variants by first computing squared Euclidean distances between pairs of objects, $D_{ij}^2 = K_{ii} - 2K_{ij} + K_{jj}$, and than computing new kernels via $K_{ij}^{(l)} = \exp(-\sigma_l D_{ij}^2)$.

Performance evaluation. In [6], all 13 classes were trained separately in a one-against-all manner, where a gene is treated as a member of a certain class whenever it has a positive label for that class (irrespective of other possible labels!). The performance of these 13 individual classifiers has been evaluated in terms of *area under the ROC curves* (*auc*). Our method, on the contrary, respects the multiclass/multilabel structure of the problem by explicitly taking into account possible multilabels of genes. When comparing the performance of our method with the results in [6] in terms of class-specific *auc* values, however, we encounter the following problem: our probabilistic multiclass approach uses *fractional labels* $y_k = 1/m_i$, where m_i denotes the *label multiplicity* of the i-the gene, and outputs a probability vector for all classes. Thus, even in the optimal case, our classifier will assign a score of $1/m_i$ to a correct class. Since the test set contains genes with *different* label multiplicities, the classifier scores will reside on different scales and it will be impossible to find a common threshold for the classifier scores when computing a ROC curve.

To overcome this problem, we use two different measures: for each class C_k, $auc_{0/1}$ measures the area under the ROC curve only on the subset of genes which either do not belong to class C_k, or which are member of C_k with label multiplicity 1. For this subset (which for the yeast genome consists of $\approx$ 2/3 of the genes) we can directly compute a ROC curve, since there are no scaling problems. The measure auc_{weighted}, on the other hand, uses all test genes and avoids scaling problems by rescaling both the fractional label y_k for class C_k and the corresponding probabilistic classifier score p_k by the individual multiplicities, $y'_k = y_k \cdot m_i \Rightarrow y'_k \in \{0, 1\}, p'_k = p_k \cdot m_i$.

The obtained performances are depicted in figure 3 for 10 random splits into (80% training) / (20% test) data (detailed description below). It turns out that the AdR model significantly outperforms the method used in [6] in most of the classes. A closer analysis shows that this improvement is only partially due to the enlarged number of kernels (which can be managed very efficiently with our algorithm). The main influence factor for the improved prediction is the direct solution of the multi-class multilabel problem, whereas the method in [6] ignores the multilabel nature of the problem completely.

To highlight the different sources of improvements two experiments are conducted: first we use the 8 "basis" kernels from [6], see http://noble.gs.washington.edu/proj/yeast/. The left panel of figure 3 depicts three performance values (area under ROC curve) for each of the 13 classes: the result reported in [6] (represented as vertical bars since no variance measurements are provided) and the two measures auc_{weighted} and $auc_{0/1}$ for our multilabel approach (represented as box-plots with median line and lower/upper quartile). Each of the latter two significantly outperforms the former in 9 of the 13 classes (marked red). The measure $auc_{0/1}$ shows an improved median performance in *all* classes (some improvements are probably not significant, marked in orange), whereas auc_{weighted} has lower performance in three of the classes (probably also not significant, marked light blue).

In a second experiment we use the increased kernel set which additionally contains 3 RBF kernel variants of the 8 "basis" kernels. The right panel of figure 3 depicts the results, again in terms of auc_{weighted} and $auc_{0/1}$. We conclude that with respect to the class-specific *auc* measures there might be a slight (but statistically insignificant) improvement due to the enlarged kernel set. The improvement becomes more obvious when computing the *F1-measure* which is the harmonic mean of *precision* and *recall*. In order to compute the latter we first select for each gene the k most probable multilabels as described at the end of section 2 above. With these sets of labels we then compute both *precision* and *recall* up to the rank k and finally combine both to the *F1-measure*. With the enlarged kernel set *F1* increases to 0.59 as compared to 0.56 for the original kernels. The improvement is due to higher precision at a comparable recall value.

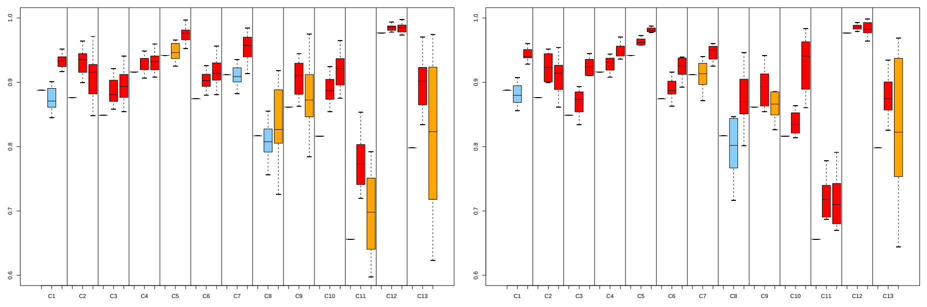

Fig. 1. Left: original kernels from [6]. For each class from left to right: results from [6], auc_{weighted} and $auc_{0/1}$. Red: signifi cant improvements; orange: probably insignifi cant improvements; light blue: probably insignifi cant worse performance. Right: enlarged kernel set.

Figure 3 depicts the learned kernel weights. Each box contains 4 bins corresponding to the original kernels from [6] and three Gaussian RBF kernel variants with decreasing kernel width. It is of particular interest that a RBF variant of the genetic interaction kernel K_{mgi} attains the highest weight of all kernels, whereas the original diffusion kernel on the interaction graph seems to contain almost no discriminative information (consistent with [6] where K_{mgi} is the least important kernel). This result shows that genetic interactions are highly important for predicting functional classes. The diffusion kernel K_{mgi}, however, does not encode this information in a suitable way. The reason for the improved performance of the RBF kernel variant might be the *local* nature of the Gaussian kernel function. A diffusion kernel encodes transition probabilities for a *random walk* model on a graph G. In the light of this random walk interpretation, the steep decay of the Gaussian kernel accentuates the local graph structure by additionally down-weighting transitions to farther nodes in the graph.

4 Discussion

The problem of learning kernel combinations has been addressed by reformulating classification as an indicator regression problem using adaptive ridge penalties. While the standard AdR model presented in [3] selects individual input features, our extensions concerning *weight sharing* and *kernelization* lead to a nonlinear model that combines/selects different *kernel matrices*.

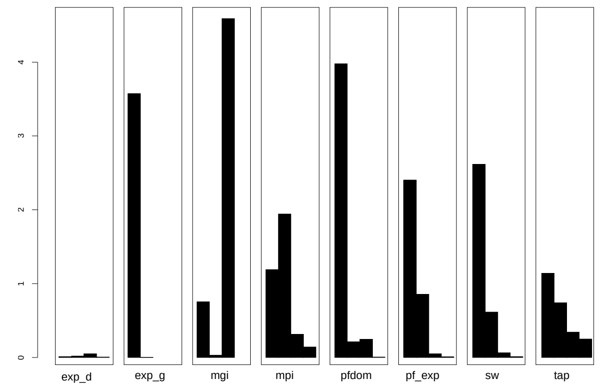

Fig. 2. Learned kernel weights in the experiment with the enlarged kernel set. Each box contains one of the 8 "basis" kernels and three Gaussian RBF variants (from left to right).

From the experiments we conclude that for the prediction of functional protein classes the AdR model compares favorably to the approach in [6]. Two aspects seem to be important: on the modeling side, our approach directly exploits the multilabel structure of the problem, rather than training one binary classifier per class and ignoring class correlations. Concerning the computational aspects, the efficiency of the method allows us to easily enlarge the set of kernels: as long as one single matrix can be hold in the main memory, the block conjugate gradient algorithm is extremely efficient. In the case of yeast proteins, the use of additional kernels has e.g. lead to the insight that genetic interactions are highly discriminative for functional predictions.

Due to a lack of space, details on using this approach in *unsupervised* settings will appear elsewhere.

References

1. F.R. Bach, G.R.G. Lanckriet, and M.I. Jordan. Multiple Kernel Learning, Conic Duality, and the SMO Algorithm. In *21st Intern. Conference on Machine Learning*, 2004.
2. AA. Dubrulle. Retooling the method of block conjugate gradients. *Electron. Trans. Numer. Anal.*, 12:216–233, 2001.
3. Y. Grandvalet. Least absolute shrinkage is equivalent to quadratic penalization. In L. Niklasson, M. Bodén, and T. Ziemske, editors, *ICANN'98*, pages 201–206. Springer, 1998.
4. T. Hastie and R. Tibshirani. Discriminant analysis by gaussian mixtures. *J. Royal Statistical Society B*, 58:158–176, 1996.
5. T. Hastie, R. Tibshirani, and A. Buja. Flexible discriminant analysis by optimal scoring. *J. American Statistical Association*, 89:1255–1270, 1994.
6. G.R.G. Lanckriet, M. Deng, N. Cristianini, M.I. Jordan, and W.S. Noble. Kernel-based data fusion and its application to protein function prediction in yeast. In *Pacifi c Symposium on Biocomputing*, pages 300–311, 2004.
7. Sören Sonnenburg, Gunnar Rätsch, and Christin Schäfer. A general and effi cient multiple kernel learning algorithm. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*. MIT Press, 2006. To appear.