# Multitask Multiple Kernel Learning (MT-MKL)

**Christian Widmer**
FML, Max Planck Society
Spemannstr. 39
72076 Tübingen, Germany
cwidmer@tuebingen.mpg.de

**Nora C. Toussaint**
ZBIT, Eberhard-Karls-Universität Tübingen
Sand 14
72076 Tübingen, Germany
toussaint@informatik.uni-tuebingen.de

**Yasemin Altun**
MPI for Biological Cybernetics
Spemannstr. 38
72076 Tübingen, Germany
altun@tuebingen.mpg.de

**Gunnar Rätsch**
FML, Max Planck Society
Spemannstr. 39
72076 Tübingen, Germany
raetsch@tuebingen.mpg.de

## Abstract

The lack of sufficient training data is the limiting factor for many Machine Learning applications in Computational Biology. If data is available for several different but related problem domains, Multitask Learning algorithms can be used to learn a model based on all available information. However, combining information from several tasks requires careful consideration of the degree of similarity between tasks. We propose to use the recently published $q$-Norm Multiple Kernel Learning algorithm to simultaneously learn or refine the similarity matrix between tasks along with the Multitask Learning classifier by formulating the Multitask Learning problem as Multiple Kernel Learning. We demonstrate the performance of our method on two problems from Computational Biology. First, we show that our method is able to improve performance on a splice site dataset with given hierarchical task structure by refining the task relationships. Second, we consider an MHC-I dataset, for which we assume no knowledge about the degree of task relatedness. Here, we are able to learn the task similarity *ab initio*. Our framework is very general as it allows to incorporate prior knowledge about tasks relationships if available, but is also able to identify task similarities in absence of such prior information. Both variants show promising results in applications from Computational Biology.

## Background

In this paper, we investigate Multitask Learning scenarios where there exists a latent structural relation across tasks. In particular, we model the relatedness between tasks by defining *meta*-tasks. Here, each meta-task corresponds to a subset of all tasks, representing the common properties of the tasks within this subset. Then, the model of each task can be derived by a convex combination of the meta-tasks it belongs to. Moreover, the latent structure over tasks can be represented by a collection of the meta-tasks. Information is transferred between two tasks $t, t'$ with respect to their relatedness according to the latent structure (number of meta-tasks in which $t, t'$ co-occur and the importance of each of these meta-tasks defined by the mixture weights).

Clearly, such an approach is highly sensitive to the chosen structure and in the absence of prior knowledge, learning the latent structure is a crucial component of MTL with non-uniform relatedness. Starting from a special case, where there exists a single meta-task consisting of all tasks (standard MTL), we show that inferring the latent structure can be cast as a Multiple Kernel Learning problem, where the base kernels are defined with respected to *Dirac* kernels [4] that establish relatedness of all possible task combinations and hence correspond to all possible meta-tasks.

In a single-task supervised learning scenario, a sample of example-label pairs $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1,\dots,n}$ is given, where the $\boldsymbol{x}_i$ live in an input space $\mathcal{X}$ and $y_i \in \{-1, 1\}$ (for binary classification). The goal is to learn a function $f$ such that $f(\boldsymbol{x}_i) \approx y_i$ that generalizes well on unseen data. Before we describe our formulation of MTL as MKL approach, we briefly review the formulation of the used Multitask Learning algorithm. For a detailed description of $q$-norm MKL, see [5].

**Multitask Learning**

In MTL, we are given one labeled sample $\mathcal{D}_t$ for each of $T$ tasks. Similar to the single-task supervised learning scenario, we are now interested in obtaining $T$ hypotheses $f_t$, one for each task. We start from a regularization-based Multitask Learning method that was similarly proposed in the context of SVMs [1–3]. The basic idea is that models $\mathbf{w}_t$ are learned simultaneously for all tasks. Information transfer between tasks is achieved by sharing a general component $\mathbf{w}_0 = \frac{1}{T}\sum_{t=1}^{T}\mathbf{w}_t$ and enforcing similarity of each $\mathbf{w}_t$ to $\mathbf{w}_0$ in the joint optimization problem via regularization. We use the following formulation, leaving out some constants for readability

$$\min_{\mathbf{w}_1,\ldots,\mathbf{w}_T} \frac{1}{2}\sum_{t=1}^{T}||\mathbf{w}_t||^2 + \sum_{t=1}^{T}||\mathbf{w}_t - \mathbf{w}_0||^2 + C\sum_{t=1}^{T}\sum_{(\boldsymbol{x},y)\in\mathcal{D}_t}\ell\left(\langle\boldsymbol{x},\mathbf{w}_t\rangle, y\right),$$

where $\ell$ is the hinge loss, $\ell(z,y)=\max\{1-yz,0\}$.

It was shown in [3], that the dual formulation of the above corresponds to the standard SVM using a modified kernel function:

$$\max_{\alpha} -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \tilde{K}(\mathbf{x}_i,\mathbf{x}_j) + \sum_{i=1}^{n}\alpha_i$$
$$\text{s.t.} \quad \alpha^T\mathbf{y}=0, \quad 0\le\alpha_i\le C\ \forall i\in\{1,n\},$$

where $K_B$ denotes the base kernel that captures the interactions between examples from all tasks and

$$\tilde{K}(\boldsymbol{x}_i,\boldsymbol{x}_j) = K_B(\boldsymbol{x}_i,\boldsymbol{x}_j) + \delta_{t(i),t(j)}K_B(\boldsymbol{x}_i,\boldsymbol{x}_j). \tag{1}$$

Here, $t(i)$ denotes the task of example $\boldsymbol{x}_i$. In the above formulation, $\tilde{K}$ is composed of the general kernel $K_B$ and the kernel $\delta_{t(i),t(j)}K_B(\boldsymbol{x}_i,\boldsymbol{x}_j)$ that captures only intra-domain interactions. In [4], the latter kernel is referred to as *Dirac* kernel. A slightly more general formulation of $\tilde{K}$ is the following, which allows to adjust the trade-off between the general kernel and the task-specific kernel:

$$\tilde{K}(\boldsymbol{x}_i,\boldsymbol{x}_j) = \beta_1 K_B(\boldsymbol{x}_i,\boldsymbol{x}_j) + \beta_2\delta_{t(i),t(j)}K_B(\boldsymbol{x}_i,\boldsymbol{x}_j),$$

where $\beta_1,\beta_2\ge 0$ and $\beta_1+\beta_2=1$.

Clearly, $\tilde{K}$ is a convex combination of base kernels and thus a valid kernel. MKL is a technique to learn the individual weights of a weighted linear combination of kernels. Thus, it seems natural to utilize MKL to learn an optimal weighting for $\tilde{K}$.

## Multitask Multiple Kernel Learning

To be able to use MKL for Multitask Learning, we need to reformulate the Multitask Learning problem as a weighted linear combination of kernels. Motivated by Equation 1, the basic idea of our decomposition is to define task-based block masks on the kernel matrix of $K_B$. Given a list of tasks $\mathcal{T}=\{t_1,...,t_T\}$, we define a kernel $K_S$ on a subset of tasks $S\subseteq\mathcal{T}$ as follows:

$$K_S(x,y) = \begin{cases} K_B(x,y), & \text{if } t(x)\in S \wedge t(y)\in S \\ 0, & \text{else}\end{cases}$$

where $t(x)$ denotes the task of example $x$. Here, each $S$ corresponds to a *meta-task* as defined in the introduction. In the most general formulation, we define a collection $\mathcal{I}=\{S_1,...,S_p\}$ of an arbitrary number $p$ of task sets $S_i$, which defines the latent structure over tasks. This collection leads to the following linear combination of kernels, which is positive semi-definite:

$$\hat{K}(x,y) = \sum_{S_i\in\mathcal{I}}\beta_i K_{S_i}(x,y)$$

Using $\hat{K}$, we can readily utilize existing MKL methods to learn the $\beta_i$. This corresponds to identifying the groups of tasks $S_i$ for which sharing information leads to improved performance. After training using MKL, we have obtained a classifier $f_t$ for each task $t$:

$$f_t(y) = \sum_{i=0}^{N}\alpha_i y_i \sum_{S_j\in\mathcal{I};t\in S_j}\beta_j K_{S_j}(x_i,y),$$

where $N$ is the total number of training examples of all tasks combined.

What remains to be discussed is how to define a collection $\mathcal{I}$ of candidate subsets $S_i$ (i.e. meta-tasks), which are subsequently to be weighted by MKL. We consider two scenarios, one where we assume to have access to a hierarchical structure relating the tasks at hand and one, where we assume no prior knowledge given about task relations. Generally, however, it is possible to utilize prior domain knowledge indicating how tasks are related to design an appropriate $\mathcal{I}$.

**Powerset MT-MKL**

With no prior information given, a natural choice is to take into account all possible subsets of tasks. Given a set tasks $\mathcal{T}$, this corresponds to considering the power set $\mathcal{P}$ of $\mathcal{T}$ (excluding the empty set) $\mathcal{I}_{\mathcal{P}} = \{S | S \in \mathcal{P}(\mathcal{T}) \wedge S \neq \emptyset\}$.

Clearly, this gives us an exponential number (i.e. $2^T$) of task sets $S_i$ of which only a few will be relevant. To identify the relevant task sets, we propose to use an L1-regularized MKL approach to yield a sparse solution. Most subset weights will be set to zero, yielding only a few relevant subsets with weights greater than zero. We expect that the examples in these subsets come from similar distributions and that it is therefore beneficial to consider interactions between them, when obtaining a multitask predictor.

**Hierarchical MT-MKL**

In the second scenario, we assume that we are given a tree structure $\mathcal{G}$ that relates our tasks at hand (see Figure 1). In this context, a task $t_i$ corresponds to a leaf in $\mathcal{G}$. Assuming hierarchical relations between tasks is particularly relevant to Computational Biology where often different tasks correspond to different organisms. In this context, we expect that the longer the common evolutionary history between two organisms, the more beneficial it is to share information between these organisms in a MTL setting. We can exploit the hierarchical structure $\mathcal{G}$ to determine which subsets might play a role for Multitask

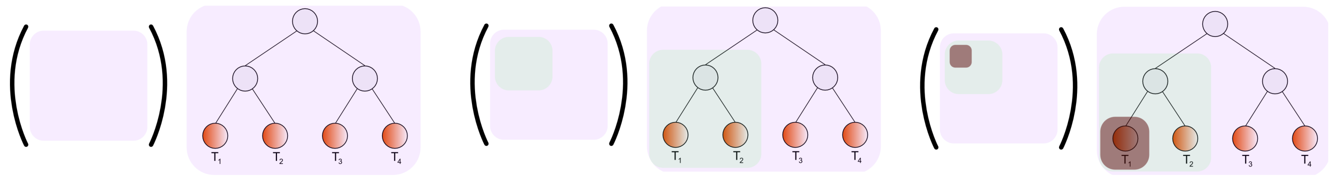

Figure 1: Example of a hierarchical decomposition.

Learning. In other words, we use the hierarchy to restrict the space of task sets. Let $leaves(n) = \{l | l \text{ is descendant of } n\}$ be the set of leaves below the sub-tree rooted at node $n$. Then, we can give the following definition for the hierarchically decomposed kernel function

$$\hat{K}(x, y) = \sum_{n_i \in \mathcal{G}} \beta_i K_{leaves(n_i)}$$

As an example, consider the kernel defined by a hierarchical decomposition according to Figure 1.

Clearly, the number of $\beta_i$ corresponds to the number of nodes. For a perfect binary tree this leads to $2m - 1$ nodes, where $m$ is the number of leaves/tasks. We expect that learning the contributions of the individual levels of the taxonomy makes sense for cases, where the edge lengths of $\mathcal{G}$ are unequal.

## Results and discussion

We performed experiments in two settings. In the first setting, we considered a set of MHC-I (major histocompatibility complex) proteins. Here, we assume we are not given any prior information to relate them. In the second setting, we used splice site data from 15 organisms and assumed that the task relationship is given by a hierarchical structure according to their evolutionary history. The examples are string data over an alphabet $\{A, T, G, C\}$ (DNA) in the splicing case and the alphabet of 20 amino acids in the MHC-I case. To incorporate string features, we used the Weighted Degree String Kernel [7], which amongst other kernels such as the Spectrum Kernel [6], has been successfully employed in problems from Computational Biology. In addition to the two MKL-based methods, we evaluated three following baseline methods: *Union*: One global model is obtained on the union of examples from all tasks, *Plain*: For each task, a model is trained independently, not taking into account any out-of-domain information, *Vanilla MTL*: In the vanilla approach, we fix all weights at $\beta_i = 1$.

**MHC-I binding prediction using Powerset MT-MKL**

In this experiment, the task is to predict whether a peptide binds to a certain MHC molecule (binder) or not (non-binder). It has been previously shown that sharing information between related molecules (alleles) and thus casting the problem in a Multitask Learning scenario, can be beneficial [4]. In the MHC setting, different tasks correspond to different MHC proteins. We report the area under the precision recall curve (auPRC) in Table 1.

| auPRC | Plain | Union | Vanilla MTL | Powerset MT-MKL |
|---|---|---|---|---|
| cross-validation | 0.668 | 0.637 | 0.676 | 0.692 |
| test set | 0.671 | 0.576 | 0.679 | 0.699 |

Table 1: Results for the MHC experiment in auPRC for the model selection step and the final prediction on the test set. Reported is the average performance over all tasks.

**Splice-site prediction using hierarchical MT-MKL**

In this setting, we take into account a given hierarchy relating the $15$ organisms in our data set. The data set consists of $6000$ examples for $15$ tasks, each at a positive to negative ratio of 1:100, similar to the one used in [8]. The data is split into $4$ splits, three splits with $333$ examples each and a large test split with $5000$ examples. The dataset was created that way to establish a scenario where positive training examples are extremely rare. For the *Vanilla MTL* method, we use the given hierarchy $\mathcal{G}$ to define the initial task sets, but not further optimize their individual influence.

We report the area under the precision recall curve (auPRC) in Table 2.

| auPRC | Plain | Union | Vanilla MTL | Hierarchical MT-MKL |
|---|---|---|---|---|
| cross-validation | 0.043 | 0.092 | 0.087 | 0.100 |
| test set | 0.059 | 0.153 | 0.169 | 0.190 |

Table 2: Results for the splice site experiment in auPRC for the model selection step and the final prediction on the test set. This table shows only the performance for the best-performing variant of Hierarchical MT-MKL with norm $q = 2$.

## Conclusions

We have presented a principle way of formulating Multitask Learning as a Multiple Kernel Learning approach. Following the basic idea of task-set-wise decomposition of the kernel matrix, we present a hierarchical decomposition and a power set based approach. These two methods allow us to elegantly identify or refine structure relating the tasks at hand in one global optimization problem. We expect our methods to work particularly well in cases, where edge weights differ within the hierarchical structure or where the task structure is unknown.

## References

[1] H. Daumé. Frustratingly easy domain adaptation. In *ACL*. The Association for Computer Linguistics, 2007.

[2] T. Evgeniou, C.A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

[3] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In Won Kim, Ron Kohavi, Johannes Gehrke, and William DuMouchel, editors, *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*, pages 109–117. ACM, 2004.

[4] L. Jacob and J.P. Vert. Efficient peptide-MHC-I binding prediction for alleles with few known binders. *Bioinformatics*, 24(3):358, 2008.

[5] Marius Kloft, Ulf Brefeld, Sören Sonnenburg, Pavel Laskov, Klaus-Robert Müller, and Alexander Zien. Efficient and accurate lp-norm multiple kernel learning. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 997–1005. MIT Press, 2009.

[6] C. Leslie, E. Eskin, and W. S. Noble. The spectrum kernel: A string kernel for SVM protein classification. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 564–575, 2002.

[7] G. Rätsch and S. Sonnenburg. *Accurate Splice Site Detection for Caenorhabditis elegans.* MIT Press, 2004.

[8] G. Schweikert, C. Widmer, B. Schölkopf, and G. Rätsch. An empirical analysis of domain adaptation algorithms. In *Advances in Neural Information Processing System, NIPS*, volume 22, Vancouver, B.C., 2008.