Multi-task Multiple Kernel Learning

Christian Widmer[*] **Marius Kloft**[†] **Gunnar Rätsch**
Computational Biology Center
Memorial Sloan-Kettering Cancer Center
415 E 68th street
New York, NY 10065, USA
{cwidmer,kloft,raetsch}@cbio.mskcc.org

Abstract

We present a general regularization-based framework for Multi-task learning (MTL), in which the similarity between tasks can be learned or refined using ℓ_p-norm Multiple Kernel learning (MKL). Based on this very general formulation (including a general loss function), we derive the corresponding dual formulation using Fenchel duality applied to Hermitian matrices. We show that numerous established MTL methods can be derived as special cases from both, the primal and dual of our formulation. Furthermore, we derive a dual-coordinate descend optimization strategy for the hinge-loss variant of our formulation and provide convergence bounds for our algorithm. We analyze various aspects of our algorithm such as predictive performance and ability to reconstruct task relationships on data sets from computational biology.

1 Introduction

In computational biology we often study the problem of building statistical models from data in order to predict, analyze, and ultimately understand biological systems. Regardless of the problem at hand, be it the recognition of sequence signals such as splice sites, the prediction of protein-protein interactions, or the modeling of metabolic networks, we frequently have access to data sets for multiple organisms, tissues or cell-lines. Thus, our goal is to develop methods that aim at *optimally* combining the data from different domains - such as organisms, tissues or cell lines - in order to improve the generalization performance of the statistical models built for each these sources. Here, we present a *general framework* for regularization-based multitask learning that subsumes several existing approaches and show how to learn or refine task similarity using *ℓ_p-norm Multiple Kernel Learning* [1, 2], an extension of regular multiple kernel learning [3, 4, 5]. In particular, this puts our previous work [6] on solid grounds. We also develop an efficient optimization solver. The C++ implementation of which we freely disseminate within the SHOGUN toolbox [7]. Exploiting efficient feature hashing techniques we able to train with millions of data points and thousands of tasks at the same time.

2 A Unifying View of Regularized Multi-Task Learning

Our main goal is to provide a method to learn task structure using non-sparse Multiple Kernel Learning, providing a middle ground between assuming known task relationships and learning the entire task structure from scratch. This can be formalized as follows:

[*] Also with Machine Learning Group, Technische Universität Berlin, Franklinstr. 28/29, 10587 Berlin, Germany.

[†] Also with Learning Theory Group, Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, NY 10012, USA.

Problem 1 (Primal problem)**.** *Solve*

$$\inf_{\boldsymbol{\theta}\in\Theta_p,\boldsymbol{W}\in\mathcal{H}} \quad \mathfrak{R}_{\boldsymbol{\theta}}(\boldsymbol{W}) \;+\; C\,\mathfrak{L}\big(A(\boldsymbol{W})\big)\,,$$

where

$$\mathfrak{R}_{\boldsymbol{\theta}}(\boldsymbol{W}) := \frac{1}{2}\sum_{m=1}^{M}\frac{\|W_m\|_{Q_m}^2}{\theta_m}\,, \quad \|W_m\|_{Q_m}^2 := \operatorname{tr}(W_m Q_m W_m^*)$$

$$\mathfrak{L}(A(\boldsymbol{W})) := \sum_{i=1}^{n} l\big(A_i(\boldsymbol{W})\big)\,,\;\; A(\boldsymbol{W}) := \big(A_i(\boldsymbol{W})\big)_{1\le i\le n}\,,\;\; A_i(\boldsymbol{W}) := y_i\sum_{m=1}^{M}\big\langle \boldsymbol{w}_{m\tau(i)}, \varphi_m(x_i)\big\rangle\,.$$

The central idea of the above formulation is to define *Multitask regularizer* $\mathfrak{R}_{\boldsymbol{\theta}}$ based on M task similarity measures Q_m. These will subsequently be weighted by tuning the corresponding Θ_m parameters using MKL. We formulate this idea in terms of general loss functions in $\mathfrak{L}(A(\boldsymbol{W}))$ and thus open the door to deriving a whole family of algorithms.

2.1 Dualization

Dual representations of optimization problems deliver insight into the problem, which can be used in practice to, for example, develop optimization algorithms (so done in Section 3 of this paper). In this section, we derive a dual representation of our unifying primal optimization problem, i.e., Problem 1. Our dualization approach is based on Fenchel-Rockafellar duality theory. The basic results of Fenchel-Rockafellar duality theory for Hilbert spaces are reviewed in Appendix F. We present two dual optimization problems: one that is dualized with respect to $\boldsymbol{W}$ only (i.e., considering $\boldsymbol{\theta}$ as being fixed) and one that completely removes the dependency on $\boldsymbol{\theta}$.

2.1.1 Dual Optimization Problems

We may now apply Fenchel's duality theorem (cf. Theorem F.3 in Appendix F), which gives the following dual MTL problem:

Problem 2 (Dual problem—partially dualized minimax formulation)**.** *Solve*

$$\inf_{\boldsymbol{\theta}\in\Theta_p}\;\sup_{\boldsymbol{\alpha}\in\mathbb{R}^n} \quad -\,\mathfrak{R}^*_{\boldsymbol{\theta}}(A^*(\boldsymbol{\alpha})) \;-\; C\,\mathfrak{L}^*(-\boldsymbol{\alpha}/C)\,, \tag{1}$$

where

$$\begin{aligned}
&\mathfrak{R}^*_{\boldsymbol{\theta}}(A^*(\boldsymbol{\alpha})) = \frac{1}{2}\sum_{m=1}^{M}\theta_m\,\|A_m^*(\boldsymbol{\alpha})\|^2_{Q_m^{-1}}\,, \quad \mathfrak{L}^*(\boldsymbol{\alpha}) = \sum_{i=1}^{n} l^*(\alpha_i)\,,\\
&A^*(\boldsymbol{\alpha}) := \big(A_m^*(\boldsymbol{\alpha})\big)_{1\le m\le M}\,, \quad A_m^*(\boldsymbol{\alpha}) = \Big(\textstyle\sum_{i\in I_t}\alpha_i y_i \varphi_m(x_i)\Big)_{1\le t\le T}\,.
\end{aligned} \tag{2}$$

The above problem involves minimization with respect to (the primal variable) $\boldsymbol{\theta}$ and maximization with respect to (the dual variable) $\boldsymbol{\alpha}$. The optimization algorithm presented later in this paper will optimize is based on this minimax formulation. However, we may completely remove the dependency on $\boldsymbol{\theta}$, which sheds further insights into the problem, which will later be exploited for optimization, i.e., to control the duality gap of the computed solutions.

To remove the dependency on $\boldsymbol{\theta}$, we first note that Problem 2 is convex (even affine) in $\boldsymbol{\theta}$ and concave in $\boldsymbol{\alpha}$ and thus, by Sion's minimax theorem, we may exchange the order of minimization and maximization:

$$\begin{aligned}
\text{Eq. (1)} \;&=\; \inf_{\boldsymbol{\theta}\in\Theta_p}\;\sup_{\boldsymbol{\alpha}\in\mathbb{R}^n} \quad -\frac{1}{2}\sum_{m=1}^{M}\theta_m\,\|A_m^*(\boldsymbol{\alpha})\|^2_{Q_m^{-1}} \;-\; C\,\mathfrak{L}^*(-\boldsymbol{\alpha}/C)\\
&=\; \sup_{\boldsymbol{\alpha}\in\mathbb{R}^n} \quad -\sup_{\boldsymbol{\theta}\in\Theta_p}\;\frac{1}{2}\sum_{m=1}^{M}\theta_m\,\|A_m^*(\boldsymbol{\alpha})\|^2_{Q_m^{-1}} \;+\; C\,\mathfrak{L}^*(-\boldsymbol{\alpha}/C)\\
&=\; \sup_{\boldsymbol{\alpha}\in\mathbb{R}^n} \quad -\frac{1}{2}\Big\|\Big(\|A_m^*(\boldsymbol{\alpha})\|^2_{Q_m^{-1}}\Big)_{1\le m\le M}\Big\|_{p^*} \;+\; C\,\mathfrak{L}^*(-\boldsymbol{\alpha}/C)
\end{aligned}$$

where the last step is by the definition of the dual norm, i.e., $\sup_{\boldsymbol{\theta}\in\Theta_p} \langle \boldsymbol{\theta}, \widetilde{\boldsymbol{\theta}} \rangle = \|\widetilde{\boldsymbol{\theta}}\|_{p^*}$ and $p^* := p/(p-1)$ denotes the conjugated exponent. We thus have the following alternative dual problem.

Problem 3 (Dual problem—completely dualized formulation)**.** *Solve*

$$\sup_{\boldsymbol{\alpha}\in\mathbb{R}^n} \; -\frac{1}{2}\left\| \left(\|A_m^*(\boldsymbol{\alpha})\|^2_{Q_m^{-1}} \right)_{1\le m\le M} \right\|_{p^*} \; + \; C\,\mathfrak{L}^*(-\boldsymbol{\alpha}/C)$$

where

$$\mathfrak{L}^*(\boldsymbol{\alpha}) = \sum_{i=1}^n l^*(\alpha_i)\,, \quad A_m^*(\boldsymbol{\alpha}) = \left(\sum\nolimits_{i\in I_t} \alpha_i y_i \varphi_m(x_i) \right)_{1\le t\le T}.$$

3 Algorithms

In this section, we present efficient optimization algorithms to solve the primal and dual problems, i.e., Problems 1 and 2, respectively. We distinguish the cases of linear and non-linear kernel matrices. For non-linear kernels, we can simply use existing MKL implementations. These standard algorithms are discussed in Appendix E. For linear kernels, we develop a specifically tailored large-scale algorithm that allows us to train on problems with millions of data points and dimensions. We present this algorithm in the next section. If the kernel admits a sparse, efficiently computable feature representation, for example, for certain string kernels and polynomial kernels of degree 2 or 3, and of course linear kernels, this algorithm can be employed. The algorithm is embedded into the COFFIN framework [8] and integrated into the SHOGUN large-scale machine learning toolbox [7].

3.1 A Large-scale Algorithm for Linear or String Kernels and Beyond

For specific kernels such as linear kernels and string kernels—and, more generally, any kernel admitting an efficient feature space representation—, we can derive a specifically tailored large-scale algorithm. This requires substantial work, which is presented below.

3.1.1 Overview

From a top-level view the upcoming algorithm underlies the core idea of alternating the following two steps:

1. the $\boldsymbol{\theta}$ step, where the kernel weights are improved
2. the $\boldsymbol{W}$ step, where the remaining primal variables are improved.

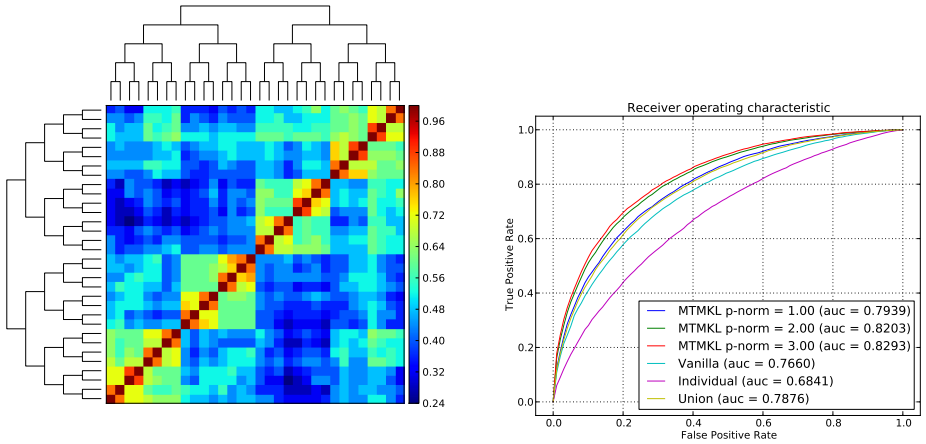
Algorithm 1 (BLUEPRINT OF THE LARGE-SCALE OPTIMIZATION ALGORITHM). The MKL module (θ step) is wrapped around the MTL modul ($\boldsymbol{W}$ step).

input: data $x_1,\ldots,x_n \in \mathcal{X}$ and labels $y_1,\ldots,y_n \in \{-1,1\}$ associated with tasks $\tau(1),\ldots,\tau(n) \in \{1,\ldots,T\}$; feature vectors $\phi_1(x_i),\ldots,\phi_M(x_i)$; task similarity matrices $Q_1,\ldots,Q_M$; optimization precision ε
initialize $\theta_m := \sqrt[q]{1/M}$ for all $m = 1,\ldots,M$, initialize $\boldsymbol{W} = \mathbf{0}$
while optimality conditions are not satisfied within tolerance ϵ **do**
    $\boldsymbol{W}$ *descend step:* compute a new $\boldsymbol{W}$ such that the objective $\mathfrak{R}_{\boldsymbol{\theta}}(\boldsymbol{W}) + C\,\mathfrak{L}(\boldsymbol{W})$ decreases
        e.g., $\boldsymbol{W} := \operatorname{argmin}_{\widetilde{\boldsymbol{W}}} \; \mathfrak{R}_{\boldsymbol{\theta}}(\widetilde{\boldsymbol{W}}) + C\,\mathfrak{L}(\widetilde{\boldsymbol{W}})$
    $\boldsymbol{\theta}$ *step:* compute the minimizer $\boldsymbol{\theta} := \operatorname{argmin}_{\widetilde{\boldsymbol{\theta}}\in\Theta_p} \; \mathfrak{R}_{\widetilde{\boldsymbol{\theta}}}(\boldsymbol{W}) + C\,\mathfrak{L}(\boldsymbol{W})$ according to (E.1)
end while
output: ϵ-accurate optimal hypothesis $\boldsymbol{W}$ and kernel weights $\boldsymbol{\theta}$

These steps are illustrated in Algorithm Table 1. We observe from the table that the variables are split into the two sets $\{\theta_m | m = 1,\ldots,M\}$ and $\{\boldsymbol{w}_{mt} | m = 1,\ldots,M, t = 1,\ldots,T\}$. The algorithm then alternatingly optimizes with respect to one or the other set until the optimality conditions are approximately satisfied. Note that similar algorithms have been used in the context of the group lasso and multiple kernel learning by, e.g., [9], [10], and [2]. A derivation of the two steps can be found in the Appendix (see Section E.1 and E.2). The resulting large-scale algorithm is summarized in Algorithm Table E.1.

4 Experiments

In this section, we evaluate Hierarchical MT-MKL on an artificial data set motivated by biological evolution. At the core of this example is the binary classification of examples generated from two 100-dimensional isotropic Gaussian distributions with a standard deviation of 20. The difference of the mean vectors μ_{pos} and μ_{neg} is captured by a difference vector μ_d. We set $\mu_{pos} = 0.5\mu_d$ and $\mu_{neg} = -0.5\mu_d$. To turn this into a MTL setting, we start with a single $\mu_d = (1, \ldots, 1)^T$ and apply mutations to it. These mutations correspond to flipping the sign of m dimensions in μ_d, where $m = 5$. Inspired by biological evolution, mutations are then applied in a hierarchical fashion according to a binary tree of depth 4 (corresponding to $2^4 = 32$ leaves). Starting at the root node, we apply subsequent mutations to the μ_d at the inner nodes and work down the tree until each leaf carries its own μ_d. We sample 10 training points and 1000 test points for each class and for each of the 32 tasks. The similarity between the μ_d at the leaves is computed by taking the dot product between all pairs and is shown in Figure 1(a). Clearly, this information is valuable when deciding which tasks (corresponding to leaves in this context) should be coupled and will be referred to as the *true* task similarity matrix in the following. The hierarchy is encoded by a set of task similarity matrices. For each node we define a task similarity matrix whose entries are set to one if two tasks are in the same subtree (i.e. both leaves are descendants of the current node) and zero otherwise. We subsequently learn a weighting for these similarity matrices using MT-MKL. We compare MT-MKL with $p = 1, 2, 3$ to the following baseline methods: *Union* that combines data from all tasks into a single group, *Individual* that treats each task separately and *Vanilla MTL* that uses MTL with the same weight for all matrices. We report the mean (averaged over tasks) ROC curve for each of the above methods in Figure 1(b). We observe that the baseline *Individual* performs worst by a



(a) Task similarity matrix

(b) Performance HMTKL vs Baselines

Figure 1: In 1(a), we show the similarity matrix between all 32 tasks as generated by the procedure outlined in the main text. Comparison of MT-MKL to baselines *Vanilla MTL*, *Union*, *Individual* is shown in 1(b), where ROC curves are averaged over tasks for each method.

large margin, suggesting that combining information from several tasks is clearly beneficial for this data set. Next, we observe that a simple way of combining tasks (i.e. *Union*) already considerably improves performance. Furthermore, we observe that learning weights of hierarchically inferred task grouping in fact improves performance compared to *Vanilla* for non-sparse MT-MKL (i.e. $p = 2, 3$). Of all methods, non-sparse MT-MKL is most accurate for all recall values.

5 Conclusion

We have presented a general framework for regularization-based MTL that supports the learning of task similarities from data by means of ℓ_p-norm MKL. We formulate a general primal problem and derive the corresponding dual, which we then use to obtain an efficient solver based on dual-coordinate descend. We demonstrate the performance of our framework in an experiment using an data set, designed to resemble a biological setting. We expect to present further biological experiments at the time of the workshop.

References

[1] M. Kloft, U. Brefeld, S. Sonnenburg, P. Laskov, K.-R. Müller, and A. 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.

[2] M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien. Lp-norm multiple kernel learning. Journal of Machine Learning Research, 12:953–997, Mar 2011.

[3] G. Lanckriet, N. Cristianini, L. E. Ghaoui, P. Bartlett, and M. I. Jordan. Learning the kernel matrix with semi-definite programming. JMLR, 5:27–72, 2004.

[4] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In Proc. 21st ICML. ACM, 2004.

[5] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. Journal of Machine Learning Research, 7:1531–1565, July 2006.

[6] C. Widmer, N.C. Toussaint, Y. Altun, and G. Rätsch. Inferring latent task structure for Multitask Learning by Multiple Kernel Learning. BMC bioinformatics, 11 Suppl 8(Suppl 8):S5, January 2010.

[7] S. Sonnenburg, G. Rätsch, S. Henschel, C. Widmer, A. Zien, F. de Bona, C. Gehl, A. Binder, and V. Franc. The SHOGUN machine learning toolbox. Journal of Machine Learning Research, 2010.

[8] Sören Sonnenburg and Vojtech Franc. Coffin: A computational framework for linear SVMs. In Johannes Fürnkranz and Thorsten Joachims, editors, ICML, pages 999–1006. Omnipress, 2010.

[9] V. Roth and B. Fischer. The group-lasso for generalized linear models: uniqueness of solutions and efficient algorithms. In Proceedings of the Twenty-Fifth International Conference on Machine Learning (ICML 2008), volume 307, pages 848–855. ACM, 2008.

[10] Z. Xu, R. Jin, H. Yang, I. King, and M. Lyu. Simple and efficient multiple kernel learning by group lasso. In Proceedings of the 27th Conference on Machine Learning (ICML 2010), 2010.

[11] M. Kloft, U. Rückert, and P. L. Bartlett. A unifying view of multiple kernel learning. In Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD), 2010. To appear. ArXiv preprint: `http://arxiv.org/abs/1005.0437`.

[12] T. Joachims. Making large–scale SVM learning practical. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, Advances in Kernel Methods — Support Vector Learning, pages 169–184, Cambridge, MA, 1999. MIT Press.

[13] S. V. N. Vishwanathan, Zhaonan sun, Nawanol Ampornpunt, and Manik Varma. Multiple kernel learning and the smo algorithm. In Advances in Neural Information Processing Systems 23, pages 2361–2369, 2010.

[14] Mehmet Gönen and Ethem Alpaydin. Multiple kernel learning algorithms. Journal of Machine Learning Research, 12:2211–2268, 2011.

[15] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. J. Optim. Theory Appl., 109(3):475–494, June 2001.

[16] Bernhard Schölkopf, Sebastian Mika, Christopher J. C. Burges, Phil Knirsch, Klaus R. Müller, Gunnar Rätsch, and Alexander J. Smola. Input space versus feature space in kernel-based methods. IEEE Transactions on Neural Networks, 10(5):1000–1017, 1999.

[17] Alexandre d'Aspremont. Smooth optimization with approximate gradient. SIAM Journal on Optimization, 19(3):1171–1183, 2008.

[18] Heinz H. Bauschke and Patrick L. Combettes. Convex analysis and monotone operator theory in Hilbert spaces. CMS Books in mathematics. Springer, New York, 2011.

[19] H. Bauschke and Y. Lucet. What is a fenchel conjugate? Notices of the AMS, 59:44–46, 2012.

[20] R. M. Rifkin and R. A. Lippert. Value regularization and Fenchel duality. J. Mach. Learn. Res., 8:441–479, 2007.

[21] Andreas Argyriou, Charles A. Micchelli, and Massimiliano Pontil. When is there a representer theorem? vector versus matrix regularizers. J. Mach. Learn. Res., 10:2507–2529, December 2009.

Appendix

A Notation

Let $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be a set of training pattern/label pairs. In multiple task learning, each training example (x_i, y_i) is associated with a task $\tau(i) \in \{1, \ldots, T\}$. Furthermore, we assume that for each $t \in \{1, \ldots, T\}$ the instances associated with task t are independently drawn from a probability distribution P_t over a measurable space $\mathcal{X}_t \times \mathcal{Y}_t$. We denote the set of indices of training points of the tth task by $I_t := \{i \in \{1, \ldots, n\} : \tau(i) = t\}$. The goal is to find, for each task $t \in \{1, \ldots, T\}$, a prediction function $f_t : \mathcal{X} \to \mathbb{R}$. In this paper, we consider composite functions of the form $f_t : x \mapsto \sum_{m=1}^{M} \langle \boldsymbol{w}_{mt}, \varphi_m(x) \rangle$, $1 \leq t \leq T$, where $\varphi_m : \mathcal{X} \to \mathcal{H}_m$, $1 \leq m \leq M$, are mappings into reproducing Hilbert spaces $\mathcal{H}_1, \ldots, \mathcal{H}_M$, encoding multiple views of the multi-task learning problem via kernels $k_m(x, \tilde{x}) = \langle \varphi_m(x), \varphi_m(\tilde{x}) \rangle$, and $\boldsymbol{W} := (\boldsymbol{w}_{mt})_{1 \leq m \leq M,\, 1 \leq t \leq T}$, $w_{mt} \in \mathcal{H}_m$ are parameter vectors of the prediction function.

For simplicity of notation, we concentrate on binary prediction, i.e., $\mathcal{Y} = \{-1, 1\}$, and encode the loss of the prediction problem via a loss term $\mathfrak{L}(\boldsymbol{W}) := \sum_{i=1}^{n} l(y_i f_{\tau(i)}(x_i))$, where $l : \mathbb{R} \to \mathbb{R} \cup \{\infty\}$ is a loss function, assumed to be closed convex, lower bounded, and finite at 0. To consider sophisticated couplings between the tasks, we introduce so-called *task-similarity matrices* $Q_1, \ldots, Q_M \in \mathrm{GL}_n(\mathbb{R})$ with $Q_m = (q_{mst})_{1 \leq s,t \leq T}$, $Q_m^{-1} = \big(q_{mst}^{(-1)}\big)_{1 \leq s,t \leq T}$ and consider the regularizer $\mathfrak{R}_{\boldsymbol{\theta}}(\boldsymbol{W}) = \frac{1}{2} \sum_{m=1}^{M} \|W_m\|_{Q_m}^2 / \theta_m$ (setting $1/0 := \infty$, $0/0 := 0$) with $\|W_m\|_{Q_m} := \mathrm{tr}(W_m Q_m W_m^*) = \sqrt{\sum_{s,t=1}^{T} q_{mst} \langle \boldsymbol{w}_{ms}, \boldsymbol{w}_{mt} \rangle}$, where $W_m = (\boldsymbol{w}_{m1}, \ldots, \boldsymbol{w}_{mT}) \in \bigoplus_{t=1}^{T} \mathcal{H}_m =: \mathcal{H}_m^T$ with adjoint W_m^* and $\mathrm{tr}(\cdot)$ denotes the trace class operator of the tensor Hilbert space $\mathcal{H}_m \otimes \mathcal{H}_m$. Note that also the direct sum $\mathcal{H} := \bigoplus_{m=1}^{M} \mathcal{H}_m^T$ is a Hilbert space, which will allow us to view $\boldsymbol{W} \in \mathcal{H}$ as an element in a Hilbert space. The parameters $\boldsymbol{\theta} = (\theta_m)_{1 \leq m \leq M} \in \Theta_p$, $\Theta_p := \{\boldsymbol{\theta} \in \mathbb{R}^M : \theta_m \geq 0, 1 \leq m \leq M, \|\boldsymbol{\theta}\|_p \leq 1\}$, are adaptive weights of the views, where $\|\boldsymbol{\theta}\|_p = \sqrt[p]{\sum_{m=1}^{M} |\theta_m|^p}$ denotes the ℓ_p-norm. Here $\boldsymbol{\theta} \succeq \mathbf{0}$ denotes $\theta_m \geq 0$, $m = 1, \ldots, M$.

B Computation of Conjugates and Adjoint Map

To apply Fenchel's duality theorem, we need to compute the adjoint map A^* of the linear map $A : \mathcal{H} \to \mathbb{R}^n$, $A(\boldsymbol{W}) = \big(A_i(\boldsymbol{W})\big)_{1 \leq i \leq n}$, as well as the convex conjugates of $\mathfrak{R}$ and $\mathfrak{L}$. See Appendix F for a review of the definitions of the convex conjugate and the adjoint map. First, we notice that, by the basic identities for convex conjugates of Prop. F.4 in Appendix F, we have that

$$\big(C\mathfrak{L}(\boldsymbol{\alpha})\big)^* \;=\; C\mathfrak{L}^*(\boldsymbol{\alpha}/C) \;=\; C\Big(\sum\nolimits_{i=1}^{n} l(\alpha_i/C)\Big)^* \;=\; C\sum\nolimits_{i=1}^{n} l^*(\alpha_i/C)\,.$$

Next, we define $A^* : \mathbb{R}^n \to \mathcal{H}$ by $A^*(\boldsymbol{\alpha}) = \big(\sum_{i \in I_t} \alpha_i y_i \varphi_m(x_i)\big)_{1 \leq m \leq M, 1 \leq t \leq T}$, and verify that, for any $\boldsymbol{W} \in \mathcal{H}$ and $\boldsymbol{\alpha} \in \mathbb{R}^n$, it holds

$$\begin{aligned}
\langle \boldsymbol{W}, A^*(\boldsymbol{\alpha}) \rangle &= \Big\langle \big(w_{mt}\big)_{1 \leq m \leq M, 1 \leq t \leq T}\,, \Big(\sum\nolimits_{i \in I_t} \alpha_i y_i \varphi_m(x_i)\Big)_{1 \leq m \leq M, 1 \leq t \leq T} \Big\rangle \\
&= \sum_{m=1}^{M} \sum_{t=1}^{T} \sum_{i \in I_t} \alpha_i y_i \, \langle w_{mt}, \varphi_m(x_i) \rangle \\
&= \sum_{i=1}^{n} \sum_{m=1}^{M} \alpha_i y_i \, \big\langle w_{m\tau(i)}, \varphi_m(x_i) \big\rangle \\
&= \langle A(\boldsymbol{W}), \boldsymbol{\alpha} \rangle \;.
\end{aligned}$$

Thus, A^* as defined above is indeed the adjoint map. Finally, we compute the conjugate of $\mathfrak{R}$ with respect to $\boldsymbol{W}$, where we consider $\boldsymbol{\theta}$ as a constant. We write $r_m(W_m) := \frac{1}{2} \|W_m\|_{Q_m}^2$ and note that,

by Prop. F.4,

$$\mathfrak{R}^*_{\boldsymbol{\theta}}(\boldsymbol{W}) \;=\; \left(\sum_{m=1}^{M} \theta_m^{-1} r_m(W_m)\right)^* \;=\; \sum_{m=1}^{M} \theta_m^{-1} r_m^*(\theta_m W_m)\,.$$

Furthermore,

$$r_m^*(W_m) = \sup_{V_m \in \mathcal{H}_m^T} \underbrace{\langle V_m, W_m\rangle - \frac{1}{2}\operatorname{tr}(V_m Q_m V_m)}_{=:\psi(V_m)}\,. \tag{B.1}$$

The supremum is attained when $\nabla_{V_m}\psi(V_m) = 0$ so that in the optimum $V_m = Q_m^{-1}W_m$. Resubstitution into (B.1) gives $r_m^*(W_m) = \frac{1}{2}W_m Q_m^{-1} W_m = \frac{1}{2}\left\|W_m\right\|^2_{Q_m^{-1}}$, so that we have

$$\mathfrak{R}^*_{\boldsymbol{\theta}}(\boldsymbol{W}) = \frac{1}{2}\sum_{m=1}^{M} \theta_m \left\|W_m\right\|^2_{Q_m^{-1}}\,.$$

C Specific Instantiations of the Framework

In this section, we show that several regularization-based multi-task learning machines are subsumed by the generalized primal and dual formulations of Problems 1–2. We will thus need to compute dual losses and dual regularizers.

C.1 Hinge Loss

Many existing multi-task learning machines deploy the hinge loss $l(a) = \max(0, 1-a)$. Employing the hinge loss in Problem 1, yields the loss term

$$\mathfrak{L}(A(\boldsymbol{W})) = \sum_{i=1}^{n} \max\left(0, 1 - y_i \sum\nolimits_{m=1}^{M} \left\langle \boldsymbol{w}_{m\tau(i)}, \varphi_m(x_i)\right\rangle\right)\,.$$

Furthermore, we observe that the conjugate of the hinge loss is $l^*(a) = a$, if $-1 \leq a \leq 0$ and ∞ elsewise, which is readily verified by elementary calculus. Thus, we have

$$-C\,\mathfrak{L}^*(-\boldsymbol{\alpha}/C) = -C\sum_{i=1}^{n} l^*(-\alpha_i/C) = \sum_{i=1}^{n}\alpha_i \tag{C.1}$$

provided that $\forall i = 1,\ldots,n : 0 \leq \alpha_i \leq C$; otherwise we $-C\,\mathfrak{L}^*(-\boldsymbol{\alpha}/C) = -\infty$. Hence, we obtain the following dual problem,

$$\inf_{\boldsymbol{\theta}\in\Theta_p}\;\sup_{\mathbf{0}\preceq\boldsymbol{\alpha}\preceq\mathbf{C}}\;\; -\,\mathfrak{R}^*_{\boldsymbol{\theta}}(A^*(\boldsymbol{\alpha})) \;+\; \sum_{i=1}^{n}\alpha_i\,.$$

D General Algorithms for Non-linear Kernels

A very convenient way to numerically solve the proposed framework is to simply exploit existing MKL implementations. To see this, recall that if we use the multi-task kernels $\tilde{k}_1,\ldots,\tilde{k}_M$ as the set of multiple kernels, Problem 2 is equivalent to

$$\inf_{\boldsymbol{\theta}\in\Theta_p}\;\sup_{\boldsymbol{\alpha}\in\mathbb{R}^n}\;\; -\frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \sum_{m=1}^{M}\tilde{k}_m(x_i,x_j) \;-\; C\,\mathfrak{L}^*(-\boldsymbol{\alpha}/C)\,,$$

which is exactly the optimization problem of ℓ_p-norm multiple kernel learning as described in [1, 11, 2].

We may thus build on existing research in the field of MKL and use one the prevalent efficient implementations to solve ℓ_p-norm MKL. Most of the ℓ_p-norm MKL solvers are specifically tailored to the hinge loss. Proven implementations are, for example, the interleaved optimization method of [2], which is directly integrated into the SVMLight modul [12] of the SHOGUN toolbox such that the $\boldsymbol{\theta}$-step is performed after each decomposition step, i.e., after solving the small QP occurring in SVMLight, which allows very fast convergence [5].

Another efficient optimization approach is by [13], who optimize the completely dualized MKL formulation (similar to Problem 3), that is,

$$\inf_{\boldsymbol{\theta}\in\Theta_p} \sup_{\boldsymbol{\alpha}\in\mathbb{R}^n:\sum_{i=1}^n \alpha_i y_i=0} -\frac{1}{2}\left\| \left(\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j \sum_{m=1}^M \theta_m \tilde{k}_m(x_i,x_j) \right)_{1\le m\le M} \right\|_{p^*} - C\,\mathfrak{L}^*(-\boldsymbol{\alpha}/C)\,.$$

This implementation comes along without a $\boldsymbol{\theta}$-step, but any of the α_i-steps computations of the α_i-steps are more costly as in the case of vanilla (MT-)SVMs.

For an overview of MKL algorithms and their implementations, see, e.g., the survey paper by [14], which is why this topic is not further discussed here.

E Large-scale Algorithm for Specific Kernels

E.1 Solving the $\boldsymbol{\theta}$ Step

In this section, we discuss how to compute the update of the kernel weights $\boldsymbol{\theta}$ as carried out in Line 6 of Algorithm 1. Note that for fixed $\boldsymbol{W}\in\mathcal{H}$ it holds

$$\operatorname*{arginf}_{\boldsymbol{\theta}\in\Theta_p} \ \mathfrak{R}_{\boldsymbol{\theta}}(\boldsymbol{W}) + C\,\mathfrak{L}\big(A(\boldsymbol{W})\big) \ = \ \operatorname*{arginf}_{\boldsymbol{\theta}\in\Theta_p} \ \mathfrak{R}_{\boldsymbol{\theta}}(\boldsymbol{W})\,,$$

where $\mathfrak{R}_{\boldsymbol{\theta}}(\boldsymbol{W}) = \frac{1}{2}\sum_{m=1}^M \frac{\operatorname{tr}(W_m Q_m W_m)}{\theta_m}$. Furthermore, by Lagrangian duality,

$$\begin{aligned}\inf_{\boldsymbol{\theta}\in\Theta_p} \ \frac{1}{2}\sum_{m=1}^M \frac{\operatorname{tr}(W_m Q_m W_m)}{\theta_m} \ &= \ \max_{\lambda\ge 0} \ \inf_{\boldsymbol{\theta}\succeq\mathbf{0}} \ \frac{1}{2}\sum_{m=1}^M \frac{\operatorname{tr}(W_m Q_m W_m)}{\theta_m} + \lambda\sum_{m=1}^M \theta_m^p \\ &= \ \inf_{\boldsymbol{\theta}\succeq\mathbf{0}} \ \underbrace{\frac{1}{2}\sum_{m=1}^M \frac{\operatorname{tr}(W_m Q_m W_m)}{\theta_m} + \lambda^*\sum_{m=1}^M \theta_m^p}_{=:\psi(\boldsymbol{\theta})}\,,\end{aligned}$$

where we denote the optimal λ in the above maximization by $\lambda^\star$. The infimum is either attained at the boundary of the constraints or when $\nabla_{\boldsymbol{\theta}}\psi(\boldsymbol{\theta})=0$, thus the optimal point $\boldsymbol{\theta}^\star$ satisfies $\theta_m^\star = \left(\operatorname{tr}(W_m Q_m W_m)/\lambda^\star\right)^{1/(p+1)}$ for any $m=1,\dots,M$. Because $\boldsymbol{\theta}^\star\in\Theta_p$, i.e., $\|\boldsymbol{\theta}\|_p = 1$, it follows $\lambda^\star = \left(\sum_{m=1}^M \operatorname{tr}(W_m Q_m W_m)^{p/(p+1)}\right)^{(p+1)/p}$, under the minimal assumption that $\boldsymbol{W}\neq\mathbf{0}$. Thus, because $\operatorname{tr}(W_m Q_m W_m) = \sum_{s,t=1}^T q_{mst}\left\langle \boldsymbol{w}_{ms}, \boldsymbol{w}_{mt}\right\rangle$,

$$\forall m=1,\dots,M: \quad \theta_m^\star = \frac{\sqrt[p+1]{\sum_{s,t=1}^T q_{mst}\left\langle \boldsymbol{w}_{ms}, \boldsymbol{w}_{mt}\right\rangle}}{\left(\sum_{m=1}^M \sqrt[p+1]{\sum_{s,t=1}^T q_{mst}\left\langle \boldsymbol{w}_{ms}, \boldsymbol{w}_{mt}\right\rangle}^{\,p}\right)^{1/p}}\,. \tag{E.1}$$

E.2 Solving the $\boldsymbol{W}$ Descend Step

To solve the W step as carried out in Line 4 of Algorithm 1, we consider the kernel weights $\{\theta_m | m=1,\dots,M\}$ as being fixed and optimize solely with respect to $\boldsymbol{W}$. In fact, we perform the $\boldsymbol{W}$ descend step in the dual, i.e., by optimizing the dual objective of Problem 2, i.e., solving

$$\sup_{\boldsymbol{\alpha}\in\mathbb{R}^n} \ -\mathfrak{R}^*_{\boldsymbol{\theta}}(A^*(\boldsymbol{\alpha})) \ - \ C\,\mathfrak{L}^*(-(\boldsymbol{\alpha})/C)\,.$$

Although our framework is also valid for other loss functions, for the presentation of the algorithm, we make a specific choice of a proven loss function, that is, the hinge loss $l(a)=\max(0,1-a)$, so that, by (C.1), the above task becomes

$$\sup_{\boldsymbol{\alpha}\in\mathbb{R}^n:\mathbf{0}\preceq\boldsymbol{\alpha}\preceq\mathbf{C}} \ -\mathfrak{R}^*_{\boldsymbol{\theta}}(A^*(\boldsymbol{\alpha})) + \sum_{i=1}^n \alpha_i\,. \tag{E.2}$$

We discuss extensions to other loss functions, such as the logistic loss, at the end of this section. Our algorithm optimizes (E.2) by dual coordinate ascend, i.e., by optimizing the dual variables α_i

one after another (i.e., only a single dual variable α_i is optimized at a time),

$$\sup_{d\in\mathbb{R}:\, 0\le\alpha_i+d\le C} \; -\mathfrak{R}^*_{\boldsymbol{\theta}}(A^*(\boldsymbol{\alpha}+d\boldsymbol{e}_i)) \;+\; \sum_{i=1}^n \alpha_i + d\,,$$

where we denote the unit vector of ith coordinate in $\mathbb{R}^n$ by $\boldsymbol{e}_i$. The good message here is that, as it will turn out, this task can be performed analytically; however, performed purely in the dual it involves computing a sum over all support vectors which is infeasible for large n. Our proposed algorithm is instead based on the application of the representer theorem carried out in Section F.2: recall from (F.1) that, for all $m = 1, \ldots, M$ and $t = 1, \ldots, T$, it holds

$$\boldsymbol{w}_{mt} = \theta_m \sum_{i=1}^n q^{(-1)}_{m\tau(i)t}\alpha_i y_i \varphi_m(x_i)\,.$$

The core idea of our algorithm is to express the update of the α_i in the coordinate ascend procedure solely in terms of the vectors $\boldsymbol{w}_{mt}$. While optimizing the variables α_i one after another, we keep track of the changes in the vectors $\boldsymbol{w}_{mt}$. This procedure is reminiscent of the *dual coordinate ascent* method, but differs in the way the objective is computed. Of course, this implies that we need to manipulate feature vectors, which explains why our approach relies on efficient infrastructure of storing and computing feature vectors and their inner products. If the infrastructure is adequate so that computing inner products in the feature space is more efficient than computing a row of the kernel matrix, our algorithm will have a substantial gain.

E.3 Expressing the update of a single variable α_i in terms of the vectors $\boldsymbol{w}_{mt}$

As argued above our aim is to express the (analytical) computation of

$$\sup_{d\in\mathbb{R}:\, 0\le\alpha_i+d\le C} \; -\mathfrak{R}^*_{\boldsymbol{\theta}}(A^*(\boldsymbol{\alpha}+d\boldsymbol{e}_i)) \;+\; \sum_{i=1}^n \alpha_i + d\,.$$

solely in terms of the vectors $\boldsymbol{w}_{mt}$. To start the derivation, note that, by (2),

$$\mathfrak{R}^*_{\boldsymbol{\theta}}(A^*(\boldsymbol{\alpha}+d\boldsymbol{e}_i)) \;=\; \frac{1}{2}\sum_{m=1}^M \theta_m \left\|A^*(\boldsymbol{\alpha}+d\boldsymbol{e}_i)\right\|^2_{Q_m^{-1}}$$

with,

$$\left\|A^*(\boldsymbol{\alpha}+d\boldsymbol{e}_i)\right\|^2_{Q_m^{-1}} \;=\; \sum_{j,\tilde{j}=1}^n \alpha_j\alpha_{\tilde{j}}y_j y_{\tilde{j}}\tilde{k}_m(x_j,x_{\tilde{j}}) \;+\; 2dy_i\sum_{j=1}^n \alpha_j y_j \tilde{k}_m(x_i,x_j) \;+\; d^2 k_m(x_i,x_i)\,,$$

where

$$\tilde{k}_m(x_i,x_j) \;=\; q^{(-1)}_{m\tau(i)\tau(j)}k_m(x_i,x_j)$$

is the mth multi-task kernel. Thus,

$$\begin{aligned}
&\operatorname*{argsup}_{d\in\mathbb{R}:\, 0\le\alpha_i+d\le C} \; -\mathfrak{R}^*_{\boldsymbol{\theta}}(A^*(\boldsymbol{\alpha}+d\boldsymbol{e}_i)) \;+\; \sum_{i=1}^n \alpha_i + d \\
=\;& \operatorname*{argsup}_{d\in\mathbb{R}:\, 0\le\alpha_i+d\le C} \; d \;-\; dy_i\sum_{j=1}^n \alpha_j y_j \left(\sum\nolimits_{m=1}^M \theta_m\tilde{k}_m(x_i,x_j)\right) \;-\; \frac{1}{2}d^2\left(\sum\nolimits_{m=1}^M \theta_m\tilde{k}_m(x_i,x_j)\right) \\
=\;& \operatorname*{argsup}_{d\in\mathbb{R}:\, 0\le\alpha_i+d\le C} \; d \;-\; \underbrace{dy_i\left(\sum\nolimits_{m=1}^M \left\langle \boldsymbol{w}_{m\tau(i)}, \varphi_m(x_i)\right\rangle\right) \;-\; \frac{1}{2}d^2\left(\sum\nolimits_{m=1}^M \theta_m\tilde{k}_m(x_i,x_j)\right)}_{=:\psi(d)}\,.
\end{aligned}$$

The optimum of $\psi(d)$ is either attained at the boundaries of the constraint $0 \le \alpha_i + d \le C$ or when $\psi'(d) = 0$. Hence, the optimal $d^\star$ can be expressed analytically as

$$d^\star = \max\left(-\alpha_i\,,\;\min\left(C-\alpha_i\,,\;\frac{1-y_i\sum_{m=1}^M \theta_m\left\langle \boldsymbol{w}_{m\tau(i)},\varphi_m(x_i)\right\rangle}{\sum_{m=1}^M \theta_m\tilde{k}_m(x_i,x_j)}\right)\right)\,. \tag{E.3}$$

Whenever we update an α_i according to

$$\alpha_i^{\text{new}} := \alpha_i^{\text{old}} + d^\star$$

with d computed as in (E.3), we need to also update the vectors $\boldsymbol{w}_{mt}$, $m = 1, \ldots, M$, $t = 1, \ldots, T$, according to

$$\boldsymbol{w}_{mt}^{\text{new}} := \boldsymbol{w}_{mt}^{\text{old}} + d\theta_m q_{m\tau(i)t}^{(-1)} y_i \varphi_m(x_i)\,, \tag{E.4}$$

to be consistent with (F.1). Similar, we need to update the vectors $\boldsymbol{w}_{mt}$ after each θ step according to

$$\boldsymbol{w}_{mt}^{\text{new}} := \left(\theta_m^{\text{new}} / \theta_m^{\text{old}}\right) \boldsymbol{w}_{mt}^{\text{old}}\,. \tag{E.5}$$

To avoid recurrences in the iterates, a $\boldsymbol{\theta}$-step should only be performed, if the primal objective has decreased between subsequent $\boldsymbol{\theta}$-steps. Thus, after each $\boldsymbol{\alpha}$ epoch, the primal objective needs to be computed in terms of $\boldsymbol{W}$. As described above, the algorithm keeps $\boldsymbol{W}$ up to date when $\boldsymbol{\alpha}$ changes, which makes this task particular simple.

E.4 Large-scale Algorithm

Data and the labels are input to the algorithm as well as a sub-procedure for efficient computation of feature maps. Lines 2 and 3 initialize the optimization variables. In Line 4 the inverses of the task similarity matrices are pre-computed. Algorithm E.1 iterates over Lines 7–16 until the stopping criterion falls under a pre-defined accuracy treshold ε. In Lines 7–11 the line search is computed for all dual variables. Lines 14 and 15 update the primal variables and kernel weights to be consistent with the representer theorem, only if the primal objective has decreased since the last $\boldsymbol{\theta}$-step. We stop Algorithm E.1 when the relative change in the objective o is less than ϵ. Notice that we do not optimize the W step to full precision, but instead alternate between one pass over the α_i and a $\boldsymbol{\theta}$ step.

Algorithm E.1 (DUAL-COORDINATE-ASCEND-BASED MTL TRAINING ALGORITHM). Generalization of the LibLinear training algorithm to multiple tasks and multiple linear kernels.

1: **input:** data $x_1, \ldots, x_n \in \mathcal{X}$ and labels $y_1, \ldots, y_n \in \{-1, 1\}$ associated with tasks $\tau(1), \ldots, \tau(n) \in \{1, \ldots, T\}$; efficiently computable feature maps $\varphi_1, \ldots, \varphi_M$; task similarity matrices $Q_1, \ldots, Q_M$; optimization precision ε
2: for all $i \in \{1, \ldots, n\}$ initialize $\alpha_i = 0$
3: for all $m \in \{1, \ldots, M\}$ and $t \in \{1, \ldots, T\}$, initialize $\boldsymbol{w}_{mt}$ according to (F.1)
4: for all $m \in \{1, \ldots, M\}$, compute inverse $Q_m^{-1} = \left(q_{mst}^{(-1)}\right)_{1 \le s,t \le T}$
5: initialize primal objective $o = nC$
6: **while** optimality conditions are not satisfied **do**
7: **for** all $i \in \{1, \ldots, n\}$
8: compute d according to (E.3)
9: update $\alpha_i := \alpha_i + d$
10: for all $m \in \{1, \ldots, M\}$ and $t \in \{1, \ldots, T\}$, update $\boldsymbol{w}_{mt}$ according to (E.4)
11: **end for**
12: store primal objective $o^{\text{old}} = o$ and compute new primal objective o
13: **if** primal objective has decreased, i.e., $o < o^{\text{old}}$
14: for all $m \in \{1, \ldots, M\}$, compute θ_m from $\boldsymbol{w}_{m1}, \ldots, \boldsymbol{w}_{mT}$ according to (E.1)
15: for all $m \in \{1, \ldots, M\}$ and $t \in \{1, \ldots, T\}$, update $\boldsymbol{w}_{mt}$ according to (E.5)
16: **end if**
17: **end while**
18: **output:** ϵ-accurate optimal hypothesis $\boldsymbol{W} = (\boldsymbol{w}_{mt})_{1 \le m \le M, 1 \le t \le T}$ and kernel weights $\boldsymbol{\theta} = (\theta_m)_{1 \le m \le M}$

E.5 Convergence Analysis

In this section, we establishing convergence of Algorithm 1 under mild assumptions. To this end, we build on the established theory of convergence of the block coordinate descent method. Classical results usually assume that the function to be optimized is strictly convex and continuously differentiable. This assertion is frequently violated in machine learning when, e.g., the hinge loss is employed. In contrast, we base our convergence analysis on the work of [15] concerning the convergence of the block coordinate descent method. The following proposition is a direct consequence of Lemma 3.1 and Theorem 4.1 in [15].

Proposition E.1. *Let $f : \mathbb{R}^{d_1 + \cdots + d_R} \to \mathbb{R} \cup \{\infty\}$ be a function. Put $d = d_1 + \cdots + d_R$. Suppose that f can be decomposed into $f(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_r) = f_0(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_r) + \sum_{r=1}^{R} f_r(\boldsymbol{a}_r)$ for some $f_0 : \mathbb{R}^d \to$*

$\mathbb{R} \cup \{\infty\}$ *and* $f_r : \mathbb{R}^{d_r} \to \mathbb{R} \cup \{\infty\}$, $r = 1, \ldots, R$. *Initialize the block coordinate descent method by* $\boldsymbol{a}^0 = (\boldsymbol{a}_1^0, \ldots, \boldsymbol{a}_R^0)$. *Let* $(r_k)_{k\in\mathbb{N}} \subset \{1, \ldots, R\}$ *be a sequence of coordinate blocks. Define the iterates* $\boldsymbol{a}^k = (\boldsymbol{a}_1^k, \ldots, \boldsymbol{a}_R^k)$, $k > 0$, *by*

$$\boldsymbol{a}_{r_k}^{k+1} \in \operatorname*{argmin}_{\mathfrak{A}\in\mathbb{R}^{d_{r_k}}} f\big(\boldsymbol{a}_1^{k+1}, \cdots, \boldsymbol{a}_{r_k-1}^{k+1}, \mathfrak{A}, \boldsymbol{a}_{r_k+1}^{k}, \cdots, \boldsymbol{a}_R^k\big)\,, \quad \boldsymbol{a}_r^{k+1} := \boldsymbol{a}_r^k\,, \quad r \neq r_k\,, \quad k \in \mathbb{N}_0\,. \tag{E.6}$$

Assume that

(A1) f *is convex and proper (i.e.,* $f \not\equiv \infty$*)*

(A2) the sublevel set $\mathcal{A}^0 := \{\boldsymbol{a} \in \mathbb{R}^d : f(\boldsymbol{a}) \leq f(\boldsymbol{a}^0)\}$ *is compact and* f *is continuous on* $\mathcal{A}^0$
*(*ASSURES EXISTENCE OF MINIMIZER IN (E.6)*)*

(A3) $\operatorname{dom}(f_0) := \{a \in \mathbb{R}^d : f_0(\boldsymbol{a}) < \infty\}$ *is open and* f_0 *is Gâteaux differentiable (e.g., continuously differentiable) on* $\operatorname{dom}(f_0)$
*(*YIELDS REGULARITY—I.E., ANY COORDINATE-WISE MINIMUM IS A MINIMUM OF f*)*

(A4) it exists a number $T \in \mathbb{N}$ *so that, for each* $k \in \mathbb{N}$ *and* $r \in \{1, \ldots, R\}$, *there is* $\tilde{k} \in \{k, \ldots, k + T\}$ *with* $r_{\tilde{k}} = r$.
*(*ENSURES THAT EACH COORDINATE BLOCK IS OPTIMIZED "SUFFICIENTLY OFTEN"*)*

Then the minimizer in (E.6) *exists and any cluster point of the sequence* $(\boldsymbol{a}^k)_{k\in\mathbb{N}}$ *minimizes* f *over* $\mathcal{A}$.

Corollary E.2. *Assume that*

(B1) the data is represented by $\phi_m(x_i) \in \mathbb{R}^{e_m}$, $i = 1, \ldots, n$, $e_m < \infty$, $m = 1, \ldots, M$.

(B2) the loss function l *is convex, finite in 0, and continuous on its domain* $\operatorname{dom}(l)$

(B3) the task similarity matrices $Q_1, \ldots, Q_T$ *are positive definite*

(B3) any iterate $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_M)$ *traversed by Algorithm 1 has* $\theta_m > 0$, $m = 1, \ldots, M$

(B4) the exact search specified in Line 5 of Algorithm 1 is performed

Then Algorithm 1 is well-defined and any cluster point of the sequence traversed by the Algorithm 1 is a minimal point of Problem 1.

Proof. The corollary is obtained by applying Proposition E.1 to Problem 1, that is,

$$\inf_{\boldsymbol{W},\,\boldsymbol{\theta}:\,\boldsymbol{\theta}\in\Theta_p} \frac{1}{2}\sum_{m=1}^{M} \frac{\|W_m\|_{Q_m}^2}{\theta_m} + C\sum_{i=1}^{n} l\bigg(y_i \sum_{m=1}^{M} \big\langle \boldsymbol{w}_{m\tau(i)}, \varphi_m(x_i)\big\rangle\bigg)\,, \tag{E.7}$$

where $\Theta_p = \{\boldsymbol{\theta} \in \mathbb{R}^M : \theta_m \geq 0, m = 1, \ldots, M, \|\boldsymbol{\theta}\|_p \leq 1\}$ and, by (B1), $\boldsymbol{W} \in \mathbb{R}^{eT}$, $e = e_1 + \cdots + e_M$. Note that (E.7) can be written unconstrained as

$$\inf_{\boldsymbol{W},\,\boldsymbol{\theta}} f(\boldsymbol{W}, \boldsymbol{\theta})\,, \quad \text{where } f(\boldsymbol{W}, \boldsymbol{\theta}) := f_0(\boldsymbol{W}, \boldsymbol{\theta}) + f_1(\boldsymbol{W}) + f_2(\boldsymbol{\theta})\,, \tag{E.8}$$

by putting

$$f_0(\boldsymbol{W}, \boldsymbol{\theta}) := \frac{1}{2}\sum_{m=1}^{M} \frac{\|W_m\|_{Q_m}^2}{\theta_m} + I_{\{\boldsymbol{\theta}\succ\mathbf{0}\}}(\boldsymbol{\theta})$$

as well as

$$f_1(\boldsymbol{W}) := C\sum_{i=1}^{n} l\bigg(y_i \sum_{m=1}^{M} \big\langle \boldsymbol{w}_{m\tau(i)}, \varphi_m(x_i)\big\rangle\bigg)\,, \quad f_2(\boldsymbol{\theta}) := I_{\{\|\boldsymbol{\theta}\|_p\leq 1\}}(\boldsymbol{\theta})\,, \tag{E.9}$$

where I is the indicator function, $I_S(s) = 0$ if $s \in S$ and $I_S(s) = \infty$ elsewise. Note that we use the shorthand $\boldsymbol{\theta} \succ \mathbf{0}$ for $\theta_m > 0$, $m = 1, \ldots, M$.

Assumption (B4) ensures that applying the block coordinate descent method to (E.7) and (E.8) problems yields precisely the sequence of iterates. Thus, in order to prove the corollary, it suffices to validate that (E.8) fulfills Assumptions (A1)–(A4) in Proposition E.1.

VALIDITY OF (A1) Recall that Algorithm 1 is initialized with $\boldsymbol{W}^0 = \mathbf{0}$ and $\theta_m^0 = \sqrt[p]{1/M}$, $m = 1, \ldots, M$, so it holds

$$f(\boldsymbol{W}^0, \boldsymbol{\theta}^0) = \underbrace{f_0(\boldsymbol{W}^0, \boldsymbol{\theta}^0)}_{=0} + \underbrace{f_1(\boldsymbol{W}^0)}_{=Cn\,l(0)} + \underbrace{f_2(\boldsymbol{\theta}^0)}_{=0} = Cn\,l(0) < \infty\,, \tag{E.10}$$

hence $f \not\equiv \infty$, so f is proper. Furthermore, $\mathrm{dom}(f_0) = \{(\boldsymbol{W}, \boldsymbol{\theta}) : \boldsymbol{\theta} \succ \mathbf{0}\}$ is convex, and f_0 is convex on $\mathrm{dom}(f_0)$, so f_0 is a convex function. By (B2), the loss function l is convex, so f_1 is a convex function. The domain $\mathrm{dom}(f_2) = \{\boldsymbol{\theta} : \|\boldsymbol{\theta}\|_p \leq 1\}$ is convex, and $f_2 \equiv 0$ on its domain, so f_2 is a convex function. Thus the sum $f = f_0 + f_1 + f_2$ is a convex function, which shows (A1).

VALIDITY OF (A2) Let $(\widetilde{\boldsymbol{W}}, \widetilde{\boldsymbol{\theta}}) \in \mathcal{A}^0 := \{(\boldsymbol{W}, \boldsymbol{\theta}) : f(\boldsymbol{W}, \boldsymbol{\theta}) \leq f(\boldsymbol{W}^0, \boldsymbol{\theta}^0)\}$. We have $f_0, f_1, f_2 \geq 0$, so, for all $m = 1, \ldots, M$,

$$\begin{aligned} \frac{\|\widetilde{W}_m\|_{Q_m}}{2\theta_m} &\leq f_0(\widetilde{\boldsymbol{W}}, \widetilde{\boldsymbol{\theta}}) \leq f_0(\widetilde{\boldsymbol{W}}, \widetilde{\boldsymbol{\theta}}) + \underbrace{f_1(\widetilde{\boldsymbol{W}})}_{\geq 0} + \underbrace{f_2(\widetilde{\boldsymbol{\theta}})}_{\geq 0} \overset{\text{by (E.8)}}{=} f(\widetilde{\boldsymbol{W}}, \widetilde{\boldsymbol{\theta}}) \\ &\leq f(\boldsymbol{W}^0, \boldsymbol{\theta}^0) \overset{\text{by (E.10)}}{\leq} Cn\,l(0)\,, \end{aligned} \tag{E.11}$$

which implies $\|\widetilde{W}_m\|_{Q_m}^2 \leq 2\theta_m Cn\,l(0)$. Similar, because $f_0 \geq 0$, we have $f_2(\widetilde{\boldsymbol{W}}, \widetilde{\boldsymbol{\theta}}) \leq Cn\,l(0) < \infty$, which, by (E.9), implies $\|\widetilde{\boldsymbol{\theta}}\|_p \leq 1$ and thus $\widetilde{\theta}_m \leq 1$, $m = 1, \ldots, M$. Hence, by (E.11), $\|\widetilde{W}_m\|_{Q_m}^2 \leq 2Cn\,l(0)$, $m = 1, \ldots, M$. Because $Q_1, \ldots, Q_M$ are positive definite, $\nu := \min_{m=1,\ldots,M} \mathrm{tr}(Q_m) > 0$. Thus, for any $m = 1, \ldots, M$,

$$\begin{aligned} \|\widetilde{W}_m\|^2 &= \mathrm{tr}(\widetilde{W}_m^* \widetilde{W}_m) = \mathrm{tr}(\widetilde{W}_m^* \widetilde{W}_m)\,\mathrm{tr}(Q_m)/\,\mathrm{tr}(Q_m) \leq \mathrm{tr}(\widetilde{W}_m^* W_m Q_m)/\,\mathrm{tr}(Q_m) \\ &\leq \nu^{-1}\,\mathrm{tr}(\widetilde{W}_m^* \widetilde{W}_m Q_m) = \nu^{-1}\,\mathrm{tr}(\widetilde{W}_m Q_m \widetilde{W}_m^*) = \nu^{-1}\,\|\widetilde{W}_m\|_{Q_m}^2 \leq 2\nu^{-1} Cn\,l(0)\,. \end{aligned}$$

Thus

$$\|(\widetilde{\boldsymbol{W}}, \widetilde{\boldsymbol{\theta}})\|^2 = \|\widetilde{\boldsymbol{W}}\|^2 + \|\widetilde{\boldsymbol{\theta}}\|^2 \leq 2\nu^{-1} CMn\,l(0) + M < \infty\,.$$

Thus $\sup_{(\widetilde{W}, \widetilde{\boldsymbol{\theta}}) \in \mathcal{A}^0} \|(\widetilde{W}, \widetilde{\boldsymbol{\theta}})\| < \infty$, which shows that $\mathcal{A}^0$ is bounded. Furthermore, $\mathcal{A}^0 \subset \mathrm{dom}(f) = \mathrm{dom}(f_0) \cap \mathrm{dom}(f_1) \cap \mathrm{dom}(f_2)$ and f_0, f_1, f_2 are continuous on their respective domains. Thus f is continuous on $\mathrm{dom}(f)$ and thus also on its subset $\mathcal{A}^0$. It holds $\mathcal{A}^0 = f^{-1}(]-\infty, f(\boldsymbol{W}^0, \boldsymbol{\theta}^0)])$, i.e., $\mathcal{A}^0$ is the preimage of closed set under a continuous function; thus $\mathcal{A}^0$ is closed. Any closed and bounded subset of $\mathbb{R}^d$ is compact. Thus $\mathcal{A}^0$ is compact, which was to show.

VALIDITY OF (A3) AND (A4) Clearly, $\mathrm{dom}(f_0) = \{(\boldsymbol{W}, \boldsymbol{\theta}) : \boldsymbol{\theta} \succ \mathbf{0}\}$ is open and f_0 is continuously differentiable on $\mathrm{dom}(f_0)$. Thus it is Gâteaux differentiable on $\mathrm{dom}(f_0)$. Finally, assumption (A4) is trivially fulfilled as Algorithm 1 employs a simple alternating rule for traversing the blocks of coordinates.

In summary, Proposition E.1 can thus be applied to Problem 1, which yields the claim of the corollary. □

Remark E.3. *In this paper, we experiment on finite-dimensional string kernels, so Assumption (B1) is naturally fulfilled. Note that, more generally, $\phi(x_i) \in \mathbb{R}^{d_m}$ for all $i = 1, \ldots, n$, $m = 1, \ldots, M$, can be enforced also for infinite-dimensional kernels, as, for any finite sample $x_1, \ldots, x_n$, there exists a n-dimensional feature representation of the sample that can be explicitly computed in terms of the empirical kernel map [16].*

Remark E.4. *Note that Algorithms 1 and E.1 differ in the way the $\boldsymbol{\alpha}$-step is performed. While the former performs an exact search for an $\boldsymbol{\alpha}$ that is optimal given the current $\boldsymbol{\theta}$ iterate, the latter performs an $\boldsymbol{\alpha}$ descent step. It is indicative that this also yields a convergent procedure and our empirical evidence points strongly in this direction. The theoretical investigation of the convergence of such an* approximate block coordinate descent methods *is, however, beyond the scope of this paper. A good starting point for such an analysis could be the works of [15] and [17].*

F Fenchel Duality in Hilbert Spaces

In this section, we review Fenchel duality theory for convex functions over real Hilbert spaces. The results presented in this appendix are taken from Chapters 15 and 19 in [18]. For complementary reading, we refer to the excellent introduction of [19]. Fenchel duality for machine learning has also been discussed in [20] assuming Euclidean spaces. We start the presentation with the definition of the convex conjugate function.

Definition F.1 (Convex conjugate)**.** *Let $\mathcal{H}$ be a real Hilbert space and let $g : \mathcal{H} \to \mathbb{R} \cup \{\infty\}$ be a convex function. We assume in the whole section that g is proper, that is, $\{\boldsymbol{w} \in \mathcal{H} \mid g(\boldsymbol{w}) \in \mathbb{R}\} \neq \emptyset$. Then the convex conjugate $g^* : \mathcal{H} \to \mathbb{R} \cup \{\infty\}$ is defined by $g^*(\boldsymbol{w}) = \sup_{\boldsymbol{v} \in \mathcal{H}} \langle \boldsymbol{v}, \boldsymbol{w} \rangle - g^*(\boldsymbol{v})$.*

As the convex conjugate is a supremum over affine functions, it is convex and lower semi-continuous. We have the beautiful duality

$$g = g^{**} \Leftrightarrow \begin{cases} g \text{ is convex and} \\ \text{lower semi-continuous} \,. \end{cases}$$

This indicates that the "right domain" to study conjugate functions is the set of convex, lower semi-continuous, and proper ("ccp") functions. In order to present the main result of this appendix, we need the following standard result from operator theory.

Proposition F.2 (Definition and uniqueness of the adjoint map)**.** *Let $\mathcal{H}$ be a real Hilbert space and let $A : \mathcal{H} \to \widetilde{\mathcal{H}}$ be a continuous linear map. Then there exists a unique continuous linear map $A^* : \widetilde{\mathcal{H}} \to \mathcal{H}$ with $\langle A(\boldsymbol{w}), \boldsymbol{\alpha} \rangle = \langle \boldsymbol{w}, A^* \boldsymbol{\alpha} \rangle$, which called* adjoint map *of A.*

For example, in the Euclidean case, we have $\mathcal{H} = \mathbb{R}^m$, $\widetilde{\mathcal{H}} = \mathbb{R}^n$, and $A \in \mathbb{R}^{m \times n}$ so that simply the transpose $A^* = A^\top \in \mathbb{R}^{n \times m}$. We now present the main result of this appendix, which is known as *Fenchel's duality theorem*:

Theorem F.3 (Fenchel's duality theorem)**.** *Let $\mathcal{H}, \widetilde{\mathcal{H}}$ be real Hilbert spaces and let $g : \mathcal{H} \to \mathbb{R} \cup \{\infty\}$ and $h : \widetilde{\mathcal{H}} \to \mathbb{R} \cup \{\infty\}$ be ccp. Let $A : \mathcal{H} \to \widetilde{\mathcal{H}}$ be a continuous linear map. Then the primal and dual problems,*

$$\begin{aligned} p^* &= \inf_{\boldsymbol{w} \in \mathcal{H}} \; g(\boldsymbol{w}) + h(A(\boldsymbol{w})) \\ d^* &= \sup_{\boldsymbol{\alpha} \in \widetilde{\mathcal{H}}} \; -g^*(A^*(\boldsymbol{\alpha})) - h^*(-\boldsymbol{\alpha}) \,, \end{aligned}$$

satisfy weak duality *(i.e., $d^* \leq p^*$). Assume, furthermore, that $A($dom$(g)) \cap$ cont$(h) \neq \emptyset$, where* $\mathrm{dom}(f) := \{\boldsymbol{w} \in \mathcal{H} : g(\boldsymbol{w}) < \infty\}$ *and* cont(h) $:= \{\boldsymbol{\alpha} \in \widetilde{\mathcal{H}} : h$ *continuous in* $\boldsymbol{\alpha}\}$*. Then we even have* strong duality *(i.e., $d^* = p^*$) and any optimal solution $(\boldsymbol{w}^\star, \boldsymbol{\alpha}^\star)$ satisfies*

$$\boldsymbol{w}^\star = \nabla g^*(A^*(\boldsymbol{\alpha}^\star)) \,,$$

if $g^* \circ A^*$ is (Gâteaux) differentiable in $\boldsymbol{\alpha}^\star$.

When applying Fenchel duality theory, we frequently need to compute the convex conjugates of certain functions. To this end, the following computation rules are helpful.

Proposition F.4. *The following computation rules hold for the convex conjugate:*

1. *Let $g : \mathcal{H} \to \mathbb{R} \cup \{\infty\}$ be a proper convex function on a real Hilbert space $\mathcal{H}$. Then, for any $\lambda > 0$ and $\boldsymbol{w} \in \mathcal{H}$, we have $(\lambda g)^*(\boldsymbol{w}) = \lambda h^*(\boldsymbol{w}/\lambda)$.*
2. *Furthermore, assume that $\mathcal{H} = \mathcal{H}_1 \bigoplus \mathcal{H}_2$ and $g(\boldsymbol{w}) = g_1(\boldsymbol{w}_1) + g_2(\boldsymbol{w}_2)$, where $g_1 : \mathcal{H}_1 \to \mathbb{R} \cup \{\infty\}$ and $g_2 : \mathcal{H}_2 \to \mathbb{R} \cup \{\infty\}$ are proper convex functions on Hilbert spaces $\mathcal{H}_1$ and $\mathcal{H}_2$, respectively. Then, for any $\boldsymbol{w} = (\boldsymbol{w}_1, \boldsymbol{w}_2) \in \mathcal{H}_1 \bigoplus \mathcal{H}_2$, we have $g^*(\boldsymbol{w}) = g_1^*(\boldsymbol{w}_1) + g_2^*(\boldsymbol{w}_2)$.*

F.1 Conjugate of the Logistic Loss

The following lemma gives the convex conjugate of the logistic loss.

Lemma F.5 (Conjugate of Logistic Loss)**.** *The conjugate of the logistic loss, defined as $l(a) = \log(1 + \exp(-a))$, is given by*

$$l^*(a) = -t \log(-a) + (1 + a) \log(1 + a) \,.$$

Proof. By definition of the conjugate,

$$l^*(a) \;=\; \sup_{b\in\mathbb{R}} \underbrace{ab \;-\; \log(1+\exp(-b))}_{=:\psi(b)} .$$

The supremum is attained when $\psi'(b) = 0$, which translates into $b = -\log(-a/(1+a))$ and $1+\exp(-b) = 1/(1+a)$. Thus

$$l^*(a) \;=\; -a\log(-a/(1+a)) - \log(1/(1+a)) \;=\; -a\log(-a) + (1+a)\log(1+a)\,,$$

which was to show □

F.2 Representer Theorem

Fenchel's duality theorem (Theorem F.3 in Appendix F) yields a useful optimality condition, that is,

$$(\boldsymbol{W}^\star, \boldsymbol{\alpha}^\star) \text{ optimal } \;\Leftrightarrow\; \boldsymbol{W}^\star = \nabla g^*(A^*(\boldsymbol{\alpha}^\star))\,,$$

under the minimal assumption that $g \circ A^*$ is differentiable in $\boldsymbol{\alpha}^\star$. The above requirement can be thought of as an analogue to the KKT condition *stationarity* in Lagrangian duality. Note that can rewrite the above equation by inserting the definitions of g and A from the previous subsection; this gives, for any $m = 1,\ldots,M$,

$$\forall m = 1,\ldots,M: \quad W_m^\star = \theta_m Q_m^{-1}\left(\sum\nolimits_{i\in I_t} \alpha_i^\star y_i \varphi_m(x_i)\right)_{1\le t\le T}\,,$$

which we may rewrite as

$$\forall\, m = 1,\ldots,M,\, t = 1,\ldots,T: \quad \boldsymbol{w}_{mt}^\star = \theta_m \sum_{i=1}^{n} q^{(-1)}_{m\tau(i)t}\alpha_i^\star y_i \varphi_m(x_i)\,. \tag{F.1}$$

The above equation gives us a representer theorem [21] for the optimal $\boldsymbol{W}^\star$, which we will exploit later in this paper for deriving an efficient optimization algorithm to solve Problem 1.