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 $\ell_p$-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 $\ell_p$-norm Multiple Kernel Learning \cite{1,2}, an extension of regular multiple kernel learning \cite{3,4,5}. In particular, this puts our previous work \cite{6} on solid grounds. We also develop an efficient optimization solver. The C++ implementation of which we freely disseminate within the SHOGUN toolbox \cite{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:
Problem 1 (Primal problem). Solve
\[
\inf_{\theta \in \Theta_p, W \in \mathcal{H}} \mathcal{R}_\theta(W) + C \mathcal{L}(A(W)),
\]
where
\[
\mathcal{R}_\theta(W) := \frac{1}{2} \sum_{m=1}^{M} \frac{||W_m||^2_{Q_m}^2}{\theta_m}, \quad ||W_m||^2_{Q_m} := \text{tr}(W_m Q_m W_m)
\]

\[
\mathcal{L}(A(W)) := \sum_{i=1}^{n} l_i(A(W)), \quad A(W) := (A_i(W))_{1 \leq i \leq n}, \quad A_i(W) := y_i \sum_{m=1}^{M} \langle w_{m\tau(i)}, \varphi_m(x_i) \rangle.
\]

The central idea of the above formulation is to define Multitask regularizer $\mathcal{R}_\theta$ based on $M$ task similarity measures $Q_m$. These will subsequently be weighted by tuning the corresponding $\Theta_m$ parameters using MKL. We formulate this idea in terms of general loss functions in $\mathcal{L}(A(W))$ 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 $W$ only (i.e., considering $\theta$ as being fixed) and one that completely removes the dependency on $\theta$.

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
\[
\inf_{\theta \in \Theta_p} \sup_{\alpha \in \mathbb{R}^n} - \mathcal{R}_\theta^*(A^*(\alpha)) - C \mathcal{L}^*(-\alpha/C),
\]
where
\[
\mathcal{R}_\theta^*(A^*(\alpha)) = \frac{1}{2} \sum_{m=1}^{M} \theta_m \frac{||A_m^*(\alpha)||^2_{Q_m}^2}{\alpha_m}, \quad \mathcal{L}^*(\alpha) = \sum_{i=1}^{n} l^*(\alpha_i),
\]
\[
A^*(\alpha) := (A^*_m(\alpha))_{1 \leq m \leq M}, \quad A^*_m(\alpha) = \left( \sum_{t \in T_i} \alpha_i y_i \varphi_m(x_t) \right)_{1 \leq t \leq T}.
\]

The above problem involves minimization with respect to (the primal variable) $\theta$ and maximization with respect to (the dual variable) $\alpha$. The optimization algorithm presented later in this paper will optimize is based on this minimax formulation. However, we may completely remove the dependency on $\theta$, 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 $\theta$, we first note that Problem 2 is convex (even affine) in $\theta$ and concave in $\alpha$ and thus, by Sion’s minimax theorem, we may exchange the order of minimization and maximization:

\[
\text{Eq. (1)} = \inf_{\theta \in \Theta_p} \sup_{\alpha \in \mathbb{R}^n} - \frac{1}{2} \sum_{m=1}^{M} \theta_m \frac{||A_m^*(\alpha)||^2_{Q_m}^2}{\alpha_m} - C \mathcal{L}^*(-\alpha/C)
\]
\[
= \sup_{\alpha \in \mathbb{R}^n} - \sup_{\theta \in \Theta_p} \frac{1}{2} \sum_{m=1}^{M} \theta_m \frac{||A_m^*(\alpha)||^2_{Q_m}^2}{\alpha_m} + C \mathcal{L}^*(-\alpha/C)
\]
\[
= \sup_{\alpha \in \mathbb{R}^n} - \frac{1}{2} \left( \frac{||A_m^*(\alpha)||^2_{Q_m}^2}{\alpha_m} \right)_{1 \leq m \leq M} + C \mathcal{L}^*(-\alpha/C)
\]
where the last step is by the definition of the dual norm, i.e., $\sup_{\theta \in \Omega_p} \langle \theta, \tilde{\theta} \rangle = \|\tilde{\theta}\|_p$, 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

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

where

$$
L^*(\alpha) = \sum_{i=1}^n I^*(\alpha_i), \quad A_m^*(\alpha) = \left( \sum_{t \in T} \alpha_t y_t \varphi_m(x_t) \right)_{1 \leq t \leq T}.
$$

### 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 $\theta$ step, where the kernel weights are improved
2. the $W$ step, where the remaining primal variables are improved.

**Algorithm 1** (Blueprint of the large-scale optimization algorithm). The MKL module ($\theta$ step) is wrapped around the MTL module ($W$ step).

1: **input:** data $x_1, \ldots, x_n \in \mathcal{X}$ and labels $y_1, \ldots, y_n \in \{-1, 1\}$ associated with tasks $\tau(1), \ldots, \tau(n) \in \{1, \ldots, T\}$; feature vectors $\phi_1(x_i), \ldots, \phi_M(x_i)$; task similarity matrices $Q_1, \ldots, Q_M$; optimization precision $\varepsilon$

2: initialize $\theta_m := \sqrt{1/M}$ for all $m = 1, \ldots, M$; initialize $W = 0$

3: **while** optimality conditions are not satisfied within tolerance $\varepsilon$ **do**

4: $W$ descend step: compute a new $W$ such that the objective $\mathcal{L}_W(W) + C \mathcal{L}(W)$ decreases

5: e.g., $W := \arg\min_{W} \mathcal{L}_W(W) + C \mathcal{L}(W)$

6: $\theta$ step: compute the minimizer $\theta := \arg\min_{\theta \in \Omega_p} \mathcal{L}_\theta(W) + C \mathcal{L}(W)$ according to \ref{E.1}

7: **end while**

8: **output:** $\varepsilon$-accurate optimal hypothesis $W$ and kernel weights $\theta$

These steps are illustrated in Algorithm Table \ref{Table1}. We observe from the table that the variables are split into the two sets $\{\theta_m | m = 1, \ldots, M\}$ and $\{w_{mt} | m = 1, \ldots, M, t = 1, \ldots, T\}$. 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 \ref{E.1} and \ref{E.2}). The resulting large-scale algorithm is summarized in Algorithm Table \ref{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 $\mu_{\text{pos}}$ and $\mu_{\text{neg}}$ is captured by a difference vector $\mu_d$. We set $\mu_{\text{pos}} = 0.5\mu_d$ and $\mu_{\text{neg}} = -0.5\mu_d$. 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 $\mu_d$, 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 $\mu_d$ at the inner nodes and work down the tree until each leaf carries its own $\mu_d$. We sample 10 training points and 1000 test points for each class and for each of the 32 tasks. The similarity between the $\mu_d$ 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 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 $\ell_p$-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


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 $t$th 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 $f_t : \mathcal{X} \rightarrow \mathbb{R}$. In this paper, we consider composite functions of the form $f_t : x \mapsto \sum_{m=1}^M \langle w_{mt}, \varphi_m(x) \rangle$, $1 \leq t \leq T$, where $\varphi_m : \mathcal{X} \rightarrow \mathcal{H}_m$, $1 \leq m \leq M$, 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 $W := (w_{mt})_{1 \leq m \leq M, 1 \leq t \leq T}$, $w_{mt} \in \mathcal{H}_m$ 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 $L(W) := \sum_{i=1}^n \ell(y_i f_{\tau(i)}(x_i))$, where $l : \mathbb{R} \rightarrow \mathbb{R} \cup \{\infty\}$ 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 $Q_1, \ldots, Q_M \in \text{GL}_n(\mathbb{R})$ with $Q_m = (q_{mnt})_{1 \leq s, t \leq T}$, $Q_m^{-1} = (q_{mnt}^{-1})_{1 \leq s, t \leq T}$, and consider the regularizer $\mathcal{R}(W) = \frac{1}{2} \sum_{m=1}^M ||W_m||_{Q_m}^2$ (setting 1/0 := $\infty$, 0/0 := 0) with $W_m := \text{tr}(W_m Q_m W_m^*) = \sqrt{\sum_{s,t=1}^T q_{mnt} \langle w_{ms}, w_{mt} \rangle}$, where $W_m = (w_{m1}, \ldots, w_{mT}) \in \bigoplus_{t=1}^T \mathcal{H}_m =: \mathcal{H}_m^T$, with adjoint $W_m^*$ and $\text{tr}(\cdot)$ denotes the trace class operator of the tensor Hilbert space $\mathcal{H}_m \otimes \mathcal{H}_m$. Note that also the direct sum $\mathcal{H} := \bigoplus_{m=1}^M \mathcal{H}_m$ is a Hilbert space, which will allow us to view $W \in \mathcal{H}$ as an element in a Hilbert space. The parameters $\theta = (\theta_m)_{1 \leq m \leq M} \in \Theta_p$, $\Theta_p := \{\theta \in \mathbb{R}^M : \theta_m \geq 0, 1 \leq m \leq M, ||\theta||_p \leq 1\}$, are adaptive weights of the views, where $||\theta||_p = \sqrt{\sum_{m=1}^M |\theta_m|^p}$ denotes the $l_p$-norm. Here $\theta \geq 0$ 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 $A : \mathcal{H} \rightarrow \mathbb{R}^n$, $A(W) = (A_i(W))_{1 \leq i \leq n}$, as well as the convex conjugates of $\mathcal{R}$ and $L$. 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. $\ref{prop:basicidentities}$, we have that

$$
(C \mathcal{L}(\alpha))^* = C \mathcal{L}^*(c/C/C) = C \left( \sum_{i=1}^n l(\alpha_i/C) \right)^* = C \sum_{i=1}^n l^* (\alpha_i/C).
$$

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

$$
\langle W, A^*(\alpha) \rangle = \left\langle \left( w_{mt} \right)_{1 \leq m \leq M, 1 \leq t \leq T}, \left( \sum_{i \in I_t} \alpha_i y_i \varphi_m(x_i) \right)_{1 \leq m \leq M, 1 \leq t \leq T} \right\rangle
$$

$$
= \sum_{m=1}^M \sum_{t=1}^T \left( \sum_{i \in I_t} \alpha_i y_i \right) \langle w_{mt}, \varphi_m(x_i) \rangle
$$

$$
= \sum_{m=1}^M \sum_{t=1}^T \alpha_i y_i \left( \sum_{i \in I_t} \langle w_{mt}, \varphi_m(x_i) \rangle \right)
$$

$$
= \langle A(W), \alpha \rangle.
$$

Thus, $A^*$ as defined above is indeed the adjoint map. Finally, we compute the conjugate of $\mathcal{R}$ with respect to $W$, where we consider $\theta$ as a constant. We write $r_m(W_m) := \frac{1}{2} ||W_m||_{Q_m}$ and note that,
by Prop. F.4

\[ R_{\theta}(W) = \left( \sum_{m=1}^{M} \theta_m^{-1} r_m(W_m) \right)^{*} = \sum_{m=1}^{M} \theta_m^{-1} r_m^*(\theta_m W_m). \]

Furthermore,

\[ r_m^*(W_m) = \sup_{V_m \in \mathcal{H}^T_m} \langle V_m, W_m \rangle - \frac{1}{2} \text{tr}(V_m Q_m V_m) =: \psi(V_m) \quad \text{(B.1)} \]

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

\[ r_m^*(W_m) = \frac{1}{2} \| W_m \|^2_{Q_m^{-1}} =: \psi(W_m). \]

\[ R_{\theta}(W) = \frac{1}{2} \sum_{m=1}^{M} \theta_m \| W_m \|^2_{Q_m^{-1}}. \]

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

\[ L(A(W)) = \sum_{i=1}^{n} \max \left( 0, 1 - y_i \sum_{m=1}^{M} \langle w_m \tau(i), \varphi_m(x_i) \rangle \right). \]

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

\[ -C L^*(-\alpha/C) = -C \sum_{i=1}^{n} l^*(-\alpha_i/C) = \sum_{i=1}^{n} \alpha_i \]

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

\[ \inf_{\theta \in \Theta} \sup_{0 \leq \alpha \leq C} -R_{\theta}(A(\alpha)) + \sum_{i=1}^{n} \alpha_i. \]

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

\[ \inf_{\theta \in \Theta} \sup_{\alpha \in \mathbb{R}^n} -R_{\theta}(A(\alpha)) + \sum_{i=1}^{n} \alpha_i. \]

which is exactly the optimization problem of \( \ell_p \)-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 \( \ell_p \)-norm MKL. Most of the \( \ell_p \)-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 module [12] of the SHOGUN toolbox such that the \( \theta \)-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,

\[
\inf_{\theta \in \Theta_p} \sup_{\alpha \in \mathbb{R}^n : \sum_{i=1}^n \alpha_i y_i = 0} - \frac{1}{2} \left( \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \sum_{m=1}^M \theta_m k_m(x_i, x_j) \right)_{1 \leq m \leq M} + p^*.
\]

This implementation comes along without a \( \theta \)-step, but any of the \( \alpha \)-steps computations of the \( \alpha \)-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 \( \theta \) Step

In this section, we discuss how to compute the update of the kernel weights \( \theta \) as carried out in Line 4 of Algorithm 1. Note that for fixed \( W \in \mathcal{H} \) it holds

\[
\arg \inf_{\theta \in \Theta_p} \mathcal{R}_\theta(W) + C \mathcal{L}(A(W)) = \arg \inf_{\theta \in \Theta_p} \mathcal{R}_\theta(W),
\]

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

\[
\inf_{\theta \geq 0} \frac{1}{2} \sum_{m=1}^M \frac{\text{tr}(W_m Q_m W_m)}{\theta_m} = \max_{\lambda \geq 0} \inf_{\theta \geq 0} \frac{1}{2} \sum_{m=1}^M \frac{\text{tr}(W_m Q_m W_m)}{\theta_m} + \lambda \sum_{m=1}^M \theta_m^p,
\]

where we denote the optimal \( \lambda \) in the above maximization by \( \lambda^* \). The infimum is either attained at the boundary of the constraints or when \( \nabla_{\theta} \psi(\theta) = 0 \), thus the optimal point \( \theta^*_m = (\text{tr}(W_m Q_m W_m)/\lambda^*)^{1/(p+1)} \) for any \( m = 1, \ldots, M \). Because \( \theta^* \in \Theta_p \), i.e., \( \| \theta \|_p = 1 \), it follows

\[
\lambda^* = \left( \frac{\sum_{m=1}^M \text{tr}(W_m Q_m W_m)^p}{p+1} \right)^{1/p},
\]

under the minimal assumption that \( W \neq 0 \). Thus, because \( \text{tr}(W_m Q_m W_m) = \sum_{s,t=1}^T q_{mst} (w_{ms} w_{mt}) \),

\[
\forall m = 1, \ldots, M : \quad \theta_m^* = \left( \frac{p+1}{\sum_{s,t=1}^T q_{mst} (w_{ms} w_{mt})} \right)^{1/p} \left( \frac{\sum_{m=1}^M p+1}{\sum_{s,t=1}^T q_{mst} (w_{ms} w_{mt})} \right)^{1/(p+1)} \left( \frac{\sum_{m=1}^M p+1}{\sum_{s,t=1}^T q_{mst} (w_{ms} w_{mt})} \right)^{1/(p+1)}.
\]

E.2 Solving the \( W \) Descend Step

To solve the \( W \) step as carried out in Line 4 of Algorithm 1, we consider the kernel weights \( \{\theta_m | m = 1, \ldots, M\} \) as being fixed and optimize solely with respect to \( W \). In fact, we perform the \( W \) descend step in the dual, i.e., by optimizing the dual objective of Problem 3, i.e., solving

\[
\sup_{\alpha \in \mathbb{R}^n} - \mathcal{R}_\theta^*(A^*(\alpha)) - C \mathcal{L}^*(-\alpha/C).
\]

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

\[
\sup_{\alpha \in \mathbb{R}^n : 0 \leq \alpha \leq C} - \mathcal{R}_\theta^*(A^*(\alpha)) + \sum_{i=1}^n \alpha_i.
\]

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

\[
\sup_{d \in \mathbb{R}: 0 \leq \alpha_i + d \leq C} - \mathcal{R}_\theta(A^*(\alpha + de_i)) + \sum_{i=1}^{n} \alpha_i + d,
\]

where we denote the unit vector of \(i\)th coordinate in \(\mathbb{R}^n\) by \(e_i\). 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

\[
w_{mt} = \theta_m \sum_{i=1}^{n} q_{mT(i)}^{(i)} \alpha_i y_i \varphi_m(x_i).
\]

The core idea of our algorithm is to express the update of the \(\alpha_i\) in the coordinate ascend procedure solely in terms of the vectors \(w_{mt}\). While optimizing the variables \(\alpha_i\) one after another, we keep track of the changes in the vectors \(w_{mt}\). 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 \(\alpha_i\) in terms of the vectors \(w_{mt}\)

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

\[
\sup_{d \in \mathbb{R}: 0 \leq \alpha_i + d \leq C} - \mathcal{R}_\theta(A^*(\alpha + de_i)) + \sum_{i=1}^{n} \alpha_i + d.
\]

soley in terms of the vectors \(w_{mt}\). To start the derivation, note that, by [2],

\[
\mathcal{R}_\theta(A^*(\alpha + de_i)) = \frac{1}{2} \sum_{m=1}^{M} \theta_m \|A^*(\alpha + de_i)\|^2_{Q_m^{-1}}
\]

with,

\[
\|A^*(\alpha + de_i)\|^2_{Q_m^{-1}} = \sum_{j=1}^{n} \alpha_j \alpha_j y_j y_j \tilde{k}_m(x_j, x_j) + 2dy_i \sum_{j=1}^{n} \alpha_j y_j \tilde{k}_m(x_i, x_j) + d^2 k_m(x_i, x_i),
\]

where

\[
\tilde{k}_m(x_i, x_j) = q_{mT(i)}^{(i)} k_m(x_i, x_j)
\]

is the \(m\)th multi-task kernel. Thus,

\[
\arg\sup_{d \in \mathbb{R}: 0 \leq \alpha_i + d \leq C} - \mathcal{R}_\theta(A^*(\alpha + de_i)) + \sum_{i=1}^{n} \alpha_i + d
\]

\[
= \arg\sup_{d \in \mathbb{R}: 0 \leq \alpha_i + d \leq C} d - dy_i \sum_{j=1}^{n} \alpha_j y_j \left( \sum_{m=1}^{M} \theta_m \tilde{k}_m(x_i, x_j) \right) - \frac{1}{2} d^2 \left( \sum_{m=1}^{M} \theta_m \tilde{k}_m(x_i, x_j) \right)
\]

\[
= \arg\sup_{d \in \mathbb{R}: 0 \leq \alpha_i + d \leq C} d - dy_i \left( \sum_{m=1}^{M} \theta_m \tilde{k}_m(x_i, x_j) \right) - \frac{1}{2} d^2 \left( \sum_{m=1}^{M} \theta_m \tilde{k}_m(x_i, x_j) \right)
\]

where

\[
\psi(d) = \min \left\{ C - \alpha_i, \frac{1 - y_i \sum_{m=1}^{M} \theta_m \tilde{k}_m(x_i, x_j)}{\sum_{m=1}^{M} \theta_m \tilde{k}_m(x_i, x_j)} \right\}.
\]

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

\[
d^* = \max \left\{ -\alpha_i, \min \left\{ C - \alpha_i, \frac{1 - y_i \sum_{m=1}^{M} \theta_m \tilde{k}_m(x_i, x_j)}{\sum_{m=1}^{M} \theta_m \tilde{k}_m(x_i, x_j)} \right\} \right\}.
\]

Whenever we update an \(\alpha_i\) according to

\[
\alpha_i^{\text{new}} := \alpha_i^{\text{old}} + d^*
\]
with $d$ computed as in (E.3), we need to also update the vectors $w_{mt}$, $m = 1, \ldots, M$, $t = 1, \ldots, T$, according to
\[
    w_{mt}^{new} := w_{mt}^{old} + d\theta m q_m^{(-1)} \varphi_m(x_t),
\]
(E.4)
to be consistent with (F.1). Similar, we need to update the vectors $w_{mt}$ after each $\theta$ step according to
\[
    w_{mt}^{new} := \left( \frac{\nu_{mt}^{new}}{\nu_{mt}^{old}} \right) w_{mt}^{old}.
\]
(E.5)
To avoid recurrences in the iterates, a $\theta$-step should only be performed, if the primal objective has decreased between subsequent $\theta$-steps. Thus, after each $\alpha$ epoch, the primal objective needs to be computed in terms of $W$. As described above, the algorithm keeps $W$ up to date when $\alpha$ 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 data and the labels are input to the algorithm as well as a sub-procedure for efficient computation which makes this task particular simple.

**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 $\varepsilon$
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 $w_{mt}$ according to (E.1)
4: for all $m \in \{1, \ldots, M\}$, compute inverse $Q_m^{-1} = (q_{mst}^{(-1)})_{1 \leq s,t \leq T}$
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 $w_{mt}$ according to (E.4)
11: end for
12: store primal objective $o^{old} = o$ and compute new primal objective $o$
13: if primal objective has decreased, i.e., $o < o^{old}$
14:     for all $m \in \{1, \ldots, M\}$, compute $\theta_m$ from $w_{m1}, \ldots, w_{mT}$ according to (E.1)
15:     for all $m \in \{1, \ldots, M\}$ and $t \in \{1, \ldots, T\}$, update $w_{mt}$ according to (E.5)
16: end if
17: end while
18: output: $\varepsilon$-accurate optimal hypothesis $W = (w_{mt})_{1 \leq m \leq M, 1 \leq t \leq T}$ and kernel weights $\theta = (\theta_m)_{1 \leq m \leq M}$

### 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 $f : \mathbb{R}^{d_1 + \cdots + d_R} \rightarrow \mathbb{R} \cup \{\infty\}$ be a function. Put $d = d_1 + \cdots + d_R$. Suppose that $f$ can be decomposed into $f(a_1, \ldots, a_r) = f_0(a_1, \ldots, a_r) + \sum_{r=1}^{R} f_r(a_r)$ for some $f_0 : \mathbb{R}^d \rightarrow \mathbb{R}$.
\[ \mathbb{R} \cup \{\infty\} \text{ and } f_r : \mathbb{R}^d \to \mathbb{R} \cup \{\infty\}, \ r = 1, \ldots, R. \] Initialize the block coordinate descent method by \( a^0 = (a_{1}^0, \ldots, a_{R}^0). \) Let \((r_k)_{k \in \mathbb{N}} \subset \{1, \ldots, R\}\) be a sequence of coordinate blocks. Define the iterates \( a^k = (a_{1}^k, \ldots, a_{R}^k), \ k > 0, \) by

\[
a_{r_k}^{k+1} \in \arg\min_{a \in \mathbb{R}^{d_{r_k}}} f\left(a_1^{k+1}, \ldots, a_{r_k-1}^{k+1}, a, a_{r_k+1}^{k+1}, \ldots, a_{R}^{k}\right), \quad a_{r_k}^{k+1} := a_{r_k}^k, \ r \neq r_k, \ k \in \mathbb{N}_0. \tag{E.6}
\]

Assume that

\begin{enumerate}[(A1)]
  \item \( f \) is convex and proper (i.e., \( f \neq \infty \))
  \item the sublevel set \( A^0 := \{ a \in \mathbb{R}^d : f(a) \leq f(a^0) \} \) is compact and \( f \) is continuous on \( A^0 \) (assures existence of minimizer in (E.6))
  \item \( \text{dom}(f_0) := \{ a \in \mathbb{R}^d : f_0(a) < \infty \} \) is open and \( f_0 \) is Gâteaux differentiable (e.g., continuously differentiable) on \( \text{dom}(f_0) \) (yields regularity—I.e., any coordinate-wise minimum is a minimum of \( f \))
  \item it exists a number \( T \in \mathbb{N} \) so that, for each \( k \in \mathbb{N} \) and \( r \in \{1, \ldots, R\} \), there is \( \hat{k} \in \{\hat{k}, \ldots, k + T\} \) with \( r_{\hat{k}} = r \). (ensures that each coordinate block is optimized “sufficiently often”)
\end{enumerate}

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

**Corollary E.2.** Assume that

\begin{enumerate}[(B1)]
  \item the data is represented by \( \phi_m(x_i) \in \mathbb{R}^{e_m}, \ i = 1, \ldots, n, \ e_m < \infty, \ m = 1, \ldots, M. \)
  \item the loss function \( l \) is convex, finite in 0, and continuous on its domain \( \text{dom}(l) \)
  \item the task similarity matrices \( Q_1, \ldots, Q_T \) are positive definite
  \item the exact search specified in Line 5 of Algorithm 1 is performed
\end{enumerate}

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,

\[
\inf_{W, \theta : \theta \in \Theta_p} \frac{1}{2} \sum_{m=1}^{M} \frac{||W_m||_{Q_m}}{\theta_m} + C \sum_{i=1}^{n} \left(y_i \sum_{m=1}^{M} \langle w_{m \tau(i)}, \varphi_m(x_i) \rangle \right), \tag{E.7}
\]

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

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

by putting

\[
f_0(W, \theta) := \frac{1}{2} \sum_{m=1}^{M} ||W_m||_{Q_m}^{2} \theta_m + I_{\{\theta > 0\}}(\theta)
\]

as well as

\[
f_1(W) := C \sum_{i=1}^{n} \left(y_i \sum_{m=1}^{M} \langle w_{m \tau(i)}, \varphi_m(x_i) \rangle \right), \quad f_2(\theta) := I_{\{||\theta||_{p} \leq 1\}}(\theta), \tag{E.9}
\]

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 \( \theta > 0 \) 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 [I] is initialized with \( W^0 = 0 \) and \( \theta_m^0 = \sqrt{1/M}, \) \( m = 1, \ldots, M, \) so it holds
\[
\begin{align*}
f(W^0, \theta^0) = f_0(W^0, \theta^0) + f_1(W^0) + f_2(\theta^0) = Cn \ell(0) < \infty,
\end{align*}
\] hence \( f \neq \infty, \) so \( f \) is proper. Furthermore, \( \text{dom}(f_0) = \{ (W, \theta) : \theta > 0 \} \) is convex, and \( f_0 \) is convex on \( \text{dom}(f_0), \) so \( f_0 \) is a convex function. By (B2), the loss function \( l \) is convex, so \( f_1 \) is a convex function. The domain \( \text{dom}(f_2) = \{ \theta : ||\theta||_p \leq 1 \} \) 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 \( (\bar{W}, \bar{\theta}) \in \mathcal{A}^0 := \{ (W, \theta) : f(W, \theta) \leq f(W^0, \theta^0) \}. \) We have \( f_0, f_1, f_2 \geq 0, \) so, for all \( m = 1, \ldots, M, \)
\[
\frac{||\bar{W}_m||^{2}_{Q_m}}{2\theta_m} \leq f_0(\bar{W}, \bar{\theta}) \leq f_0(\bar{W}, \bar{\theta}) + f_1(\bar{W}) + f_2(\bar{\theta}) \leq f_0(\bar{W}, \bar{\theta}) \leq Cn \ell(0),
\] which implies \( ||\bar{W}_m||^{2}_{Q_m} \leq 2\theta_m Cn \ell(0). \) Similar, because \( f_0 \geq 0, \) we have \( f_2(\bar{W}, \bar{\theta}) \leq Cn \ell(0), \) which, by (E.9), implies \( ||\bar{\theta}||_p \leq 1 \) and thus \( \bar{\theta}_m \leq 1, \) \( m = 1, \ldots, M. \) Hence, by (E.11), \( ||\bar{W}_m||^{2}_{Q_m} \leq 2Cn \ell(0), \) \( m = 1, \ldots, M. \) Because \( Q_1, \ldots, Q_M \) are positive definite, \( \nu := \min_{m=1,\ldots,M} \text{tr}(Q_m) > 0. \) Thus, for any \( m = 1, \ldots, M, \)
\[
||\bar{W}_m||^2 = \text{tr}(\bar{W}^*_m \bar{W}_m) = \text{tr}(\bar{W}^*_m \bar{W}_m) \text{tr}(Q_m) / \text{tr}(Q_m) \leq \text{tr}(\bar{W}^*_m W_m Q_m) / \text{tr}(Q_m)
\]
\[
\leq \nu^{-1} \text{tr}(\bar{W}^*_m \bar{W}_m) \nu^{-1} \text{tr}(W_m Q_m \bar{W}_m) = \nu^{-1} ||\bar{W}_m||^2_{Q_m} \leq 2\nu^{-1} Cn \ell(0).
\]
Thus
\[
||\bar{W}||^2 = ||\bar{W}||^2 + ||\bar{\theta}||^2 \leq 2\nu^{-1} CMn \ell(0) + M < \infty.
\]
Thus \( \sup_{(\bar{W}, \bar{\theta}) \in \mathcal{A}^0} ||(\bar{W}, \bar{\theta})|| < \infty, \) which shows that \( \mathcal{A}^0 \) is bounded. Furthermore, \( \mathcal{A}^0 \subset \text{dom}(f) = \text{dom}(f_0) \cap \text{dom}(f_1) \cap \text{dom}(f_2) \) and \( f_0, f_1, f_2 \) are continuous on their respective domains. Thus \( f \) is continuous on \( \text{dom}(f) \) and thus also on its subset \( \mathcal{A}^0. \) It holds \( \mathcal{A}^0 = f^{-1}([ -\infty, f(W^0, \theta^0) ]), i.e., \mathcal{A}^0 \) 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, \( \text{dom}(f_0) = \{ (W, \theta) : \theta > 0 \} \) is open and \( f_0 \) is continuously differentiable on \( \text{dom}(f_0). \) Thus it is Gâteaux differentiable on \( \text{dom}(f_0). \) Finally, assumption (A4) is trivially fulfilled as Algorithm [I] employs a simple alternating rule for traversing the blocks of coordinates.

In summary, Proposition [E.1] can thus be applied to Problem [I] which yields the claim of the corollary. \( \square \)

Remark E.3. In this paper, we experiment on finite-dimensional string kernels, so Assumption (B1) is naturally fulfilled. Note that, more generally, \( \phi(x_i) \in \mathbb{R}^{d_{\text{inf}}} \) 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 [18].

Remark E.4. Note that Algorithms [I] and [E.1] differ in the way the \( \alpha \)-step is performed. While the former performs an exact search for an \( \alpha \) that is optimal given the current \( \theta \) iterate, the latter performs an \( \alpha \) 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, \( \{ w \in \mathcal{H} \mid g(w) \in \mathbb{R} \} \neq \emptyset \).

Then the convex conjugate \( g^* : \mathcal{H} \to \mathbb{R} \cup \{ \infty \} \) is defined by \( g^*(w) = \sup_{v \in \mathcal{H}} \langle v, w \rangle - g^*(v) \).

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

\[
g = g^{**} \iff \begin{cases} 
\text{g is convex and} \\
\text{lower semi-continuous.}
\end{cases}
\]

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 following the 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 \( A : \mathcal{H} \to \mathcal{H} \) be a continuous linear map. Then there exists a unique continuous linear map \( A^* : \mathcal{H} \to \mathcal{H} \) with \( (A(w), \alpha) = \langle w, A^* \alpha \rangle \), which called adjoint map of \( A \).

For example, in the Euclidean case, we have \( \mathcal{H} = \mathbb{R}^m \), \( \mathcal{H}^* = \mathbb{R}^n \), 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 \( \mathcal{H}, \mathcal{H}^* \) be real Hilbert spaces and let \( g : \mathcal{H} \to \mathbb{R} \cup \{ \infty \} \) and \( h : \mathcal{H}^* \to \mathbb{R} \cup \{ \infty \} \) be ccp. Let \( A : \mathcal{H} \to \mathcal{H}^* \) be a continuous linear map. Then the primal and dual problems,

\[
p^* = \inf_{w \in \mathcal{H}} g(w) + h(A(w)) \\
d^* = \sup_{\alpha \in \mathcal{H}^*} -g^*(A^*(\alpha)) - h^*(-\alpha),
\]

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

\[
w^* = \nabla g^*(A^*(\alpha^*)),
\]

if \( g^* \circ A^* \) is (Gâteaux) differentiable in \( \alpha^* \).

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 \( w \in \mathcal{H} \), we have \( (\lambda g)^*(w) = \lambda h^*(w/\lambda) \).

2. Furthermore, assume that \( \mathcal{H} = \mathcal{H}_1 \oplus \mathcal{H}_2 \) and \( g(w) = g_1(w_1) + g_2(w_2) \), 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 \( w = (w_1, w_2) \in \mathcal{H}_1 \oplus \mathcal{H}_2 \), we have \( g^*(w) = g_1^*(w_1) + g_2^*(w_2) \).

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,

\[ l^*(a) = \sup_{b \in \mathbb{R}} ab - \log(1 + \exp(-b)) \quad (= \psi(b)). \]

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 \( \square \)

F.2 Representer Theorem

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

\( (W^*, \alpha^*) \) optimal \( \iff W^* = \nabla g^\ast(A^\ast(\alpha^*)), \)

under the minimal assumption that \( g \circ A^\ast \) is differentiable in \( \alpha^* \). 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 \( m = 1, \ldots, M \),

\[ \forall m = 1, \ldots, M : \quad W^*_m = \theta_m Q_m^{-1} \left( \sum_{i \in I_t} \alpha_i^* y_i \varphi_m(x_i) \right)_{1 \leq t \leq T}, \]

which we may rewrite as

\[ \forall m = 1, \ldots, M, \ t = 1, \ldots, T : \quad w^*_m = \theta_m \sum_{i=1}^{n} q^{(-1)}_{m} \alpha_i^* y_i \varphi_m(x_i). \quad \text{(F.1)} \]

The above equation gives us a representer theorem [21] for the optimal \( W^* \), which we will exploit later in this paper for deriving an efficient optimization algorithm to solve Problem [I]