Efficient Training of Graph-Regularized Multitask SVMs

Christian Widmer[1,2], Marius Kloft[3], Nico Görnitz[3], and Gunnar Rätsch[1,2]

[1] Memorial Sloan-Kettering Cancer Center, New York, USA
[2] FML, Max-Planck Society, Tübingen, Germany
[3] Machine Learning Laboratory, TU Berlin, Germany

Abstract. We present an optimization framework for graph-regularized multi-task SVMs based on the *primal* formulation of the problem. Previous approaches employ a so-called multi-task kernel (MTK) and thus are inapplicable when the numbers of training examples n is large (typically $n < 20,000$, even for just a few tasks). In this paper, we present a primal optimization criterion, allowing for general loss functions, and derive its dual representation. Building on the work of Hsieh et al. [1,2], we derive an algorithm for optimizing the large-margin objective and prove its convergence. Our computational experiments show a speedup of up to *three orders of magnitude* over LibSVM and SVMLight for several standard benchmarks as well as challenging data sets from the application domain of computational biology. Combining our optimization methodology with the COFFIN large-scale learning framework [3], we are able to train a multi-task SVM using over 1,000,000 training points stemming from 4 different tasks. An efficient C++ implementation of our algorithm is being made publicly available as a part of the SHOGUN machine learning toolbox [4].

1 Introduction

The main aim of multi-task learning [5] is to leverage the information of multiple, mutually related learning tasks to make more accurate predictions for the individual tasks. For example in computational biology, multiple organisms share a part of their evolutionary history and thus contain related information that can be exploited to mutually increase the quality of predictions (see, e.g., [6,7]). Further examples of successful application domains for multi-task learning include natural language processing [8] (each speaker giving rise to a task) or computer vision [9,10], where multiple visual object classes may share some of the relevant features [11].

Recently, there has been much research revolving around *regularization-based* multi-task learning machines, which, given training points $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset \mathbb{R}^d$, each associated with a task $t(i) \in \{1. \ldots, T\}$, and labels $Y = \{y_1, \ldots, y_n\} \subset \{-1, 1\}$, for each task $t \in \{1, \ldots, T\}$ learn a linear hypothesis $\boldsymbol{x} \mapsto \langle \boldsymbol{w}_t, \boldsymbol{x} \rangle$ by solving the following mathematical optimization problem:

$$\min_{w=(\boldsymbol{w}_1;\ldots;\boldsymbol{w}_T)\in\mathbb{R}^{nT}} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + J(\boldsymbol{w}) + C\sum_{i=1}^{n} l\left(y_i \boldsymbol{w}_{t(i)}^\top \boldsymbol{x}_i\right) , \tag{1}$$

where $l : \mathbb{R} \to \mathbb{R}_{+,0}$ is a convex loss function and $J(\boldsymbol{w}_1, ..., \boldsymbol{w}_M)$ denotes an additional regularization term that promotes similarities of the hypotheses associated to the tasks [5,12,13].

One of the most popular approaches to multi-task learning is by [14], who have introduced a *graph-based* regularization framework; in this setting, each task is represented by a node in a graph and the similarities between the tasks are encoded via an adjacency matrix A, which can be used to promote couplings between tasks in (1) by putting:

$$J(\boldsymbol{w}_1, ..., \boldsymbol{w}_M) = \frac{1}{2}\sum_i \sum_j \|\boldsymbol{w}_i - \boldsymbol{w}_j\|^2 A_{i,j} \, . \tag{2}$$

Evgeniou, Micchelli, and Pontil [14] show that the dual of this formulation boils down to training a standard support vector machine [15,16] using a so-called *multi-task kernel*

$$K_{\mathrm{MTL}}((\boldsymbol{x}, s), (\tilde{\boldsymbol{x}}, t)) = S_{\mathrm{T}}(s, t) \cdot \langle \boldsymbol{x}, \tilde{\boldsymbol{x}} \rangle \, , \tag{3}$$

where $S_{\mathrm{T}}(s,t)$ is a similarity measure induced by the adjacency matrix A.

In the past, this optimization of this formulation has been addressed by decomposition-based SVM solvers such as SVMLight [17] or LibLinear [2,1] in conjunction with the "kernel" defined in (3). However, this strategy is subject to serious limitations, namely large memory requirements that come from storing the kernel matrix. These limitations allow the efficient use of multi-task learning only for a relative small number of training examples (typically $n < 20,000$, even for a small number of tasks). For larger sample sizes, strategies such as on-the-fly computation of kernel products must be used, which, however, can substantially increase the execution time.

Such large-scale learning problems are frequently encountered nowadays: for example in sequence biology, millions of examples are available from the genomes of multiple organisms and the biological interactions to be learned are typically very *complex*, so that many training examples are needed to obtain a good fit (the lack of sufficient training data is often the main bottleneck in computational biology and multi-task learning). In this paper, we address these limitations by proposing a new optimization framework and giving a high-performance implementation, which is capable of dealing with *millions* of training points at the same time.

In a nutshell, the contributions of this paper can be summarized as follows:

- We present a unifying framework for graph-regularized multi-task learning allowing for arbitrary loss functions and containing, e.g., the works of [12,14] as a special case.

- We give a general dual representation and use the so-obtained primal-dual relations to derive an efficient, provable convergent optimization algorithm for the corresponding large-margin formulation that is based on dual coordinate descent.
- A variety of computational experiments on synthetic data and proven real-world benchmark data sets as well as challenging learning problems from computational genomics show that our algorithms outperform the state-of-the-art by up to three orders of magnitude.
- By including the recent COFFIN framework [3] into our new methodology, we are, for the first time, able to perform graph-based MTL training on very large splice data set consisting of millions examples from 4 organisms.

2 A Novel View of Graph-Regularized Multi-Task Learning

All methods developed in this paper are cast into the established framework of graph-regularized multi-task learning (GB-MTL) outlined in the introduction. Note that Eq (2) may be expressed as

$$\text{Eq. (2)} = \frac{1}{2}\sum_i \sum_j \|w_i - w_j\|^2 A_{i,j} = \sum_i \sum_j w_i^T w_j L_{i,j} \,, \tag{4}$$

where $L = D - A$ denotes the graph Laplacian corresponding to a given similarity matrix A and $D_{i,j} := \delta_{i,j} \sum_k A_{i,k}$. The matrix A is of crucial importance here as it encodes the similarity of the tasks. Note that the number k of zero eigenvalues of the graph Laplacian corresponds to the number of connected components. For the scenario that we are interested in, this will be 1, always.

2.1 Primal Formulation

Using (4), we can thus re-write our base problem (1) as follows:

Generalized *Primal* MTL Problem. *Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^m$ be training data points, each denoted by a task $t(i) \in \{1, \ldots, T\}$, and let $l : \mathbb{R} \to \mathbb{R}$ be a convex loss function. Then the primal MTL optimization problem is given by*

$$\min_{\boldsymbol{w}_1,\ldots,\boldsymbol{w}_T \in \mathbb{R}^m} \quad \frac{1}{2}\sum_{t=1}^{T} \|\boldsymbol{w}_t\|_2^2 + \frac{1}{2}\sum_{s=1}^{T}\sum_{t=1}^{T} L_{st}\boldsymbol{w}_s^\top \boldsymbol{w}_t + C\sum_{i=1}^{n} l\left(y_i \boldsymbol{w}_{t(i)}^\top \boldsymbol{x}_i\right) \,. \tag{5}$$

A first problem we face is that, when applying the standard Lagrangian formalism and invoking the KKT conditions, there are couplings in between the $\boldsymbol{w}_s$ and $\boldsymbol{w}_t$. Unfortunately, this hinders expressing the $\boldsymbol{w}_t$ solely in terms of the coordinate-wise gradient of the dual objective, which is the core idea behind recently proposed optimization strategies in SVM research that we wish to exploit [1]. As a remedy, in this paper, we propose an alternative approach that is based on the following two improvements:

- First, we deploy a new dualization technique that based on the combination of Lagrangian duality with Fenchel-Legendre conjugate functions, extending the work of [18]. The so-obtained synergy allows us to derive the dual in a cleaner way than it would have been using Lagrangian duality alone.
- Second, we use the "block vector view", which—in combination with the above improvement—allows us to formulate a representer theorem that can be resolved for $\boldsymbol{w}$.

As it turns out, the combination of the above two ingredients allows us to express the weights $\boldsymbol{w}_t$ in terms of the gradients of the dual objective in a very simple way.

2.2 "Block-Vector/Matrix" View

We define $\boldsymbol{w} = (\boldsymbol{w}_1^\top, \ldots, \boldsymbol{w}_T^\top)^\top$ and $\psi : \mathbb{R}^m \mapsto \mathbb{R}^{mT}$ is the canonical injective mapping that maps a data point $\boldsymbol{x}_i \in \mathbb{R}^m$ to a vector in $\mathbb{R}^{mT}$ that is zero everywhere except at the task(i)-th block, i.e., $\psi(\boldsymbol{x}_i)$ looks like as follows:

$$\psi(\boldsymbol{x}_i) := \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \boldsymbol{x}_i \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad \leftarrow \textit{t(i)-th block} \tag{6}$$

For example, if $\boldsymbol{x}_i$ belongs to the first task, i.e. $t(i) = 1$, then we have $\psi(\boldsymbol{x}_i) = (\boldsymbol{x}_i, 0, \ldots, 0)^\top$, while, if $\boldsymbol{x}_i$ belongs to the last task, i.e. $t(i) = T$, then $\psi(\boldsymbol{x}_i)$ is is of the form: $\psi(\boldsymbol{x}_i) = (0, \ldots, 0, \boldsymbol{x}_i)^\top$.

Similarly, for a matrix $B \in \mathbb{R}^{T \times T}$, we define

$$\text{block}(B) := \begin{pmatrix} \text{diag}(b_{11}) & \cdots & \text{diag}(b_{1T}) \\ \vdots & & \vdots \\ \text{diag}(b_{T1}) & \cdots & \text{diag}(b_{TT}) \end{pmatrix} , \tag{7}$$

where $\text{diag}(b_{st})$ is a diagonal matrix in $\mathbb{R}^{m \times m}$ with entries b_{st} at the diagonal and zeros everywhere else, i.e., the resulting matrix block(B) is an element of $\mathbb{R}^{mT \times mT}$.

We can thus very elegantly write our primal problem (5) in terms of the block notation as follows:

Generalized Primal MTL Problem *(Block View).*

$$\min_{w} \quad \frac{1}{2} \boldsymbol{w}^\top \textit{block}(I + L) \boldsymbol{w} + C \sum_i l\left(y_i \boldsymbol{w}^\top \psi(\boldsymbol{x}_i)\right), \tag{8}$$

where I is the identity matrix in $\mathbb{R}^{T \times T}$.

Table 1. Loss functions and regularizers used in this paper and corresponding conjugate functions

	loss $l(t)$ / regularizer $g(\boldsymbol{w})$	dual loss $l^*(t)$ / conjugate regularizer $g^*(\boldsymbol{w})$
hinge loss	$\max(0, 1-t)$	t if $-1 \le t \le 0$ and ∞ else
ℓ_p-norm	$\frac{1}{2}\left\|\boldsymbol{w}\right\|_p^2$	$\frac{1}{2}\left\|\boldsymbol{w}\right\|_{p^*}^2$ where $p^* = \frac{p}{p-1}$
quadratic form	$\frac{1}{2}\boldsymbol{w}^\top B\boldsymbol{w}$	$\frac{1}{2}\boldsymbol{w}^\top B^{-1}\boldsymbol{w}$

2.3 Dualization

Now, the above (block-view-) form of the MTL primal allows to derive the Fenchel dual as follows:

$$\begin{aligned}
\text{Eq. (8)} = \quad & \min_{\boldsymbol{w},\boldsymbol{t}} \quad [\frac{1}{2}\boldsymbol{w}^\top \text{block}(I+L)\boldsymbol{w} + C\sum_i l\left(t_i\right)] \\
& \text{s.t.} \quad t_i = y_i \boldsymbol{w}^\top \psi(\boldsymbol{x}_i) \\
\overset{\text{Lagrange}}{=} & \max_{\boldsymbol{\alpha}} \quad \min_{\boldsymbol{w},\boldsymbol{t}} \quad [\frac{1}{2}\boldsymbol{w}^\top \text{block}(I+L)\boldsymbol{w} \\
& \qquad + C\sum_i l\left(t_i\right) + \sum_i \alpha_i\left(t_i - y_i\boldsymbol{w}^\top\psi(\boldsymbol{x}_i)\right)] \qquad (9) \\
= \quad & \max_{\boldsymbol{\alpha}} \quad [-C\sum_i \max_{t_i}\left(-\frac{\alpha_i t_i}{C} - l\left(t_i\right)\right) \\
& - \max_{\boldsymbol{w}}\left(\sum_i \alpha_i y_i \boldsymbol{w}^\top\psi(\boldsymbol{x}_i) - \frac{1}{2}\boldsymbol{w}^\top\text{block}(I+L)\boldsymbol{w}\right)] .
\end{aligned}$$

We now make use of the notion of the Fenchel conjugate of a function f, that is $f^*(\boldsymbol{x}) := \sup_{\boldsymbol{y}} \boldsymbol{x}^\top\boldsymbol{y} - f(\boldsymbol{y})$ to derive a general dual form. Note that the Fenchel conjugates of many functions are known from the literature (see Table 1 for conjugates relevant for this paper; cf. [18] for further reading). For example, the conjugate of the function $f(\boldsymbol{x}) = \frac{1}{2}\left\|\boldsymbol{x}\right\|_B^2 := \frac{1}{2}\boldsymbol{x}^\top B\boldsymbol{x}$ is $f^*(\boldsymbol{x}) = \frac{1}{2}\left\|\boldsymbol{x}\right\|_{B^{-1}}^2 = \frac{1}{2}\boldsymbol{x}^\top B^{-1}\boldsymbol{x}$ and the conjugate of the hinge loss $l(t) = \max(0, 1-t)$ is $l^*(t) = t$ if $-1 \le t \le 0$ and ∞ else.

We are now ready to proceed with the derivation:

$$\begin{aligned}
\text{Eq.(8)} = \quad & \max_{\boldsymbol{\alpha}} \quad [\underbrace{-\max_{\boldsymbol{w}}\left(\sum_i \alpha_i y_i \boldsymbol{w}^\top\psi(\boldsymbol{x}_i) - \frac{1}{2}\left\|\boldsymbol{w}\right\|_{\text{block}(I+L)}^2\right)}_{=\frac{1}{2}\left\|\sum_i \alpha_i y_i \psi(\boldsymbol{x}_i)\right\|_{(\text{block}(I+L))^{-1}}^2} \\
& - C\sum_i \underbrace{\max_{t_i}\left(-\frac{\alpha_i t_i}{C} - l\left(t_i\right)\right)}_{=l^*(-\frac{\alpha_i}{C})}] \\
= \quad & \max_{\boldsymbol{\alpha}} \quad [-C\sum_i l^*\Big(-\frac{\alpha_i}{C}\Big) - \frac{1}{2}\left\|\sum_i \alpha_i y_i \psi(\boldsymbol{x}_i)\right\|_{\text{block}((I+L)^{-1})}^2 \quad]
\end{aligned}$$

where we used the definition of the Fenchel conjugate and the fact that, clearly, for any matrix B it holds

$$(\text{block}(B))^{-1} = \text{block}(B^{-1})\,.$$

We thus obtain the following MTL dual optimization problem:

General *Dual* MTL Problem. *The dual MTL problem is given by:*

$$\max_{\boldsymbol{\alpha}} \quad -C\sum_i l^*\Big(-\frac{\alpha_i}{C}\Big) - \frac{1}{2}\left\|\sum_i \alpha_i y_i \psi(\boldsymbol{x}_i)\right\|^2_{block(M)} \tag{10}$$

where

$$M := (I+L)^{-1} \tag{11}$$

2.4 Special Case: Large-Margin Learning

We can now employ specific loss functions in the primal (5) and obtain a corresponding dual representations right away by plugging the Fenchel conjugate into (10). For example, for the hinge loss, from Table 1 we obtain the conjugate of $l(t) = \max(0, 1-t)$ is $l^*(t) = t$, if $-1 \leq t \leq 0$ and ∞ else. Clearly, the minimum in (12) will never be attained for the objective being ∞ (take, e.g., $\boldsymbol{w} = \mathbf{0}$ in (5) to obtain a finite upper bound on the optimal objective) so that the left-hand term $\sum_i l^*\big(-\frac{\alpha_i}{C}\big)$ translates into the hard constraints

$$\forall i: \quad 0 \leq \alpha_i \leq C\,.$$

Moreover, by (7), we have

$$\frac{1}{2}\left\|\sum_i \alpha_i y_i \psi(\boldsymbol{x}_i)\right\|^2_{\text{block}(M)} = \frac{1}{2}\sum_{s,t=1}^{T} m_{st}\boldsymbol{w}_s^\top \boldsymbol{w}_t,$$

where $M = (m_{st})_{1\leq s,t\leq T}$, so that we obtain the following dual problem for the hinge loss:

Dual MTL-*<u>*SVM*</u> *Problem. Denote by $M := (I+L)^{-1}$. Then the dual MTL-SVM problem is given by:

$$\max_{\mathbf{0}\leq\boldsymbol{\alpha}\leq\mathbf{C}} \quad \mathbf{1}^\top\boldsymbol{\alpha} - \frac{1}{2}\left\|\sum_i \alpha_i y_i \psi(\boldsymbol{x}_i)\right\|^2_{\text{block}(M)} \tag{12}$$

2.5 A Representer Theorem

By the KKT condition *Stationarity*, it follows from (9) that

$$\nabla_{\boldsymbol{w}}\left(\sum_i \alpha_i y_i \boldsymbol{w}^\top \psi(\boldsymbol{x}_i) - \frac{1}{2}\boldsymbol{w}^\top \text{block}(I+L)\boldsymbol{w}\right) = 0\,,$$

which, by (11), translates to

$$\boldsymbol{w} = \sum_i \alpha_i y_i \boldsymbol{w}^\top M \psi(\boldsymbol{x}_i) \tag{13}$$

and (recalling the definitions (6) and (7)) can be equivalently written as

$$\boldsymbol{w}_t = \sum_{i=1}^{n} m_{t,t(i)} \alpha_i y_i \boldsymbol{x}_i . \tag{14}$$

3 Optimization Algorithms

In order to solve the optimization problem (12), we define:

$$\forall t = 1, \ldots, T : \quad \boldsymbol{v}_t = \sum_{i \in I_t} \alpha_i y_i \boldsymbol{x}_i , \tag{15}$$

where $I_t \subset \{1, \ldots, n\}$ denotes the indices of the data points of task t. We thus associate each task t with a "virtual weight vector" $\boldsymbol{v}$ that can be expressed solely terms of the support vectors corresponding to the respective task. Importantly, all the information we need to compute $\boldsymbol{w}$ is contained in $\boldsymbol{v} := (\boldsymbol{v}_1^\top, \ldots, \boldsymbol{v}_T^\top)^\top$, since by (14) holds

$$\forall 1, \ldots, T : \quad \boldsymbol{w}_t = \sum_{s=1}^{T} m_{s,t} \boldsymbol{v}_s . \tag{16}$$

If there is just a single task, as for standard SVM, i.e., $T = 1$ and $M = I$ (because $L = \mathbf{0}$), then the above definition is simply

$$\boldsymbol{w} = \boldsymbol{v} = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i ,$$

which is precisely the representation exploited by [1].

3.1 Derivation of the Optimization Algorithm

The basic idea of our *dual coordinate descent* strategy is to optimize one example weight α_i per iteration, project it onto its feasible set and then update the corresponding parameter vector $\boldsymbol{v}_t$ accordingly. In particular, we can perform *dual coordinate descent* as follows: for each $i \in \{1, \ldots, T\}$ we solve

$$\begin{aligned}
&\underset{d:0\leq\alpha_i+d\leq C}{\operatorname{argmax}} \; d + \mathbf{1}^\top \boldsymbol{\alpha} \\
&- \frac{1}{2}\sum_{s,t=1}^{T} m_{st}\left(\boldsymbol{v}_s + dy_i\boldsymbol{x}_i\mathbb{1}_{t(i)=s}\right)^\top\left(\boldsymbol{v}_t + dy_i\boldsymbol{x}_i\mathbb{1}_{t(i)=t}\right) \\
=& \underset{d:0\leq\alpha_i+d\leq C}{\operatorname{argmax}} \; d - \frac{1}{2}\Bigg(m_{t(i),t(i)}\left\|\boldsymbol{v}_{t(i)} + dy_i\boldsymbol{x}_i\right\|^2 \\
&\quad + 2\sum_{s:s\neq t(i)} m_{s,t(i)}\boldsymbol{v}_s^\top\left(\boldsymbol{v}_{t(i)} + dy_i\boldsymbol{x}_i\right)\Bigg) \\
=& \underset{d:0\leq\alpha_i+d\leq C}{\operatorname{argmax}} \; d - \Bigg(m_{t(i),t(i)}\left(dy_i\boldsymbol{v}_{t(i)}^\top\boldsymbol{x}_i + \frac{1}{2}d^2\boldsymbol{x}_i^\top\boldsymbol{x}_i\right) \\
&\quad + \sum_{s:s\neq t(i)} m_{s,t(i)}y_i\boldsymbol{v}_s^\top\boldsymbol{x}_i d\Bigg) \\
=& \underset{d:0\leq\alpha_i+d\leq C}{\operatorname{argmax}} \; d - \frac{1}{2}d^2\boldsymbol{x}_i^\top\boldsymbol{x}_i - \sum_{s=1}^{T} m_{s,t(i)}y_i\boldsymbol{v}_s^\top\boldsymbol{x}_i d
\end{aligned}$$

We thus observe that for the gradient it holds

$$\frac{\partial f(\boldsymbol{\alpha} + d\boldsymbol{e}_i)}{\partial d} = 1 - d\boldsymbol{x}_i^\top\boldsymbol{x}_i - \sum_{s=1}^{T} m_{s,t(i)}y_i\boldsymbol{v}_s^\top\boldsymbol{x}_i = 0$$

which is equivalent to

$$d = \frac{1 - \sum_{s=1}^{T} m_{s,t(i)}y_i\boldsymbol{v}_s^\top\boldsymbol{x}_i}{\boldsymbol{x}_i^\top\boldsymbol{x}_i}\,. \tag{17}$$

Therefore, taking the needed projections onto the constraints into account, we have the following update rule in each coordinate descent step:

$$\alpha_i = \max\Big(0, \min\Big(C, \alpha_i + d\Big)\Big)\,. \tag{18}$$

Note that, if there is only a single task, then $L = 0$ and thus $M = I$, where I is the identity matrix, and we hence obtain the usual LibLinear standard update (denoting $\boldsymbol{w} = \boldsymbol{v} = \boldsymbol{v}_1$):

$$d = \frac{1 - y_i\boldsymbol{w}^\top\boldsymbol{x}_i}{\boldsymbol{x}_i^\top\boldsymbol{x}_i}\,.$$

The resulting training algorithm is shown in Algorithm (1).

3.2 Convergence Analysis

To prove convergence of our algorithms, we phrase the following useful result about convergence of the (block-) coordinate descent method:

Algorithm 1. (Multi-task LibLinear training algorithm). Generalization of the LibLinear training algorithm to multiple tasks.

```
1: input: $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n \in \mathbb{R}^m$, $t(1),\ldots,t(n) \in \{1,\ldots,T\}$, $y_1,\ldots,y_n \in \{-1,1\}$
2: for all $i \in \{1,\ldots,n\}$ initialize $\alpha_i = 0$
3: for all $t \in \{1,\ldots,T\}$ put $\boldsymbol{v}_t = \sum_{i\in I_t} \alpha_i y_i \boldsymbol{x}_i$
4: while optimality conditions are not satisfied do
5:     for all $i \in \{1,\ldots,n\}$
6:         compute $d$ according to (17)
7:         store $\hat{\alpha}_i := \alpha_i$
8:         put $\alpha_i := \max(0, \min(C, \hat{\alpha}_i + d))$
9:         update $v_{t(i)} := v_{t(i)} + (\alpha_i - \hat{\alpha}_i) y_i \boldsymbol{x}_i$
10:    end for
11: end while
12: for all $t \in \{1,\ldots,T\}$ compute $\boldsymbol{w}_t$ from $\boldsymbol{v}_1,\ldots,\boldsymbol{v}_T$ according to (16)
13: output: $\boldsymbol{w}_1,\ldots,\boldsymbol{w}_T$
```

Proposition 1 (Bertsekas, 1999, Prop. 2.7.1). *Let $\mathcal{X} = \bigotimes_{m=1}^{M} \mathcal{X}_m$ be the Cartesian product of closed convex sets $\mathcal{X}_m \subset \mathbb{R}^{d_m}$, be $f : \mathcal{X} \to \mathbb{R}$ a continuously differentiable function. Define the nonlinear block Gauss-Seidel method recursively by letting $\boldsymbol{x}^0 \in \mathcal{X}$ be any feasible point, and be*

$$\boldsymbol{x}_m^{k+1} = \underset{\boldsymbol{\xi}\in\mathcal{X}_m}{\operatorname{argmin}} f\left(\boldsymbol{x}_1^{k+1}, \cdots, \boldsymbol{x}_{m-1}^{k+1}, \boldsymbol{\xi}, \boldsymbol{x}_{m+1}^{k}, \cdots, \boldsymbol{x}_M^{k}\right), \tag{19}$$

for all $m = 1,\ldots,M$. Suppose that for each m and $\boldsymbol{x} \in \mathcal{X}$, the minimum

$$\min_{\boldsymbol{\xi}\in\mathcal{X}_m} f\left(\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{m-1}, \boldsymbol{\xi}, \boldsymbol{x}_{m+1}, \cdots, \boldsymbol{x}_M\right) \tag{20}$$

is uniquely attained. Then every limit point of the sequence $\{\boldsymbol{x}^k\}_{k\in\mathbb{N}}$ is a stationary point.

The proof can be found in [19], p. 268-269. We can conclude the following corollary, which establishes convergence of the proposed MTL training algorithm.

Theorem 1. *Let l be the hinge loss. Then every limit point of Algorithm 1 is a globally optimal point of* (12).

Proof. First, note that the objective function in (12) is continuously differentiable and convex. Second, we can without loss of generality replace the constraints $0 \leq \alpha_i$ by $0 \leq \alpha_i \leq \alpha_i^*$ for all i, where $\boldsymbol{\alpha}^*$ denotes the optimal solution of (12). Thus, in order to show that the constraints form a closed set, it suffices to show that $\alpha_i^* < \infty$ for all i. To this end, we note that setting $\boldsymbol{w} = \boldsymbol{0}$, which is a feasible point in the primal (5), lets us conclude that the optimal primal objective is less than or equal to $o := C\sum_{i=1}^{n} l(0) = C\sum_{i=1}^{n} \max(0, 1-0) = Cn < \infty$. Hence, denoting by $\boldsymbol{w}^*$ the primal-optimal point, we obtain $\frac{1}{2}\|\boldsymbol{w}^*\| \leq o$ and thus, by (14), it holds $\frac{1}{2}\left\|\sum_i \alpha_i^* y_i \psi(\boldsymbol{x}_i)\right\|_{\text{block}(M)}^2 \leq o$, so that we can conclude that the dual objective in (12) in smaller than or equal to $2o < \infty$. From the latter, we can conclude $\alpha{*}_i \leq 2o < \infty$ for all i, which was sufficient to show.

4 Computational Experiments

In this section, we evaluate the runtime of our proposed dual coordinate descent (DCD) algorithm (described in Algorithm Table 1), which we have implemented[1] (along with a LibLinear-style shrinking strategy) in C++ as a part of the SHOGUN machine learning toolbox [4]. We compare our solver with the state-of-the-art, that is, SVMLight (as integrated into the SHOGUN toolbox) using the multi-task kernel (MTK) as defined in (3).[2]

We experiment on the following five data sets, whose data statistics are summarized in Table 2:

- *Gauss2D.* A controlled, synthetic data set consisting of a balanced sample from two isotropic Gaussian distributions.
- *Breast Cancer.* A classic benchmark data set consisting of a genetic signature of 60 genes used to predict the response to chemotherapy.
- *MNIST-MTL.* A multi-task data set derived from the well-known MNIST data[3] by considering the three separate tasks "1 vs. 0", "7 vs. 9", and "2 vs. 8".
- *Landmine.* A classic multi-task data set, where the different tasks correspond to detecting land mines under various conditions [20].
- *Splicing.* This is the most challenging data set: a huge-scale, multiple-genomes, biological data set, where the goal is to detect splice sites in various organisms, each organism corresponding to a task. The features are derived from raw DNA strings by means of a weighted-degree string kernel [21].

The above data sets are taken from various application domains including computer vision, biomedicine, and computational genomics, and cover many different settings such as small and large dimensionality, various numbers of examples and tasks. Our corpus includes controlled synthetic data as well as established real-world benchmark data and challenging multiple-genomes splice data. The first four data sets contain real valued data, for which we used linear kernels and corresponding standard scalar products.

To compare our implementation with SVMLight using the multi-task kernel (MTK), we measure the *function difference*

$$\Delta := \left|\mathrm{obj}^* - \widehat{\mathrm{obj}}\right| ,$$

where obj^* the true optimal objective and $\widehat{\mathrm{obj}}$ the actual objective achieved by the solver (for DCD and MTK these are primal and dual objectives, respectively). The true objective obj^* is computed up to a duality gap of $< 10^{-10}$. All experiments are performed on a 4GB AMD64 machine using a single core.

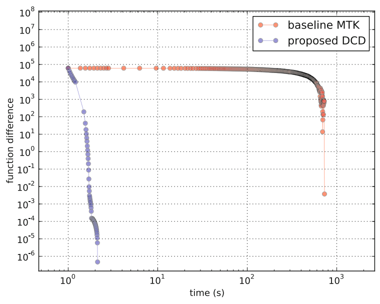
[1] For implementation details, see: `http://bioweb.me/mtl-dcd-solver`

[2] We expect very similar run times by using LIBSVM instead of SVMLight. The runtime measurement was easier to implement in SVMLight than in LIBSVM, which is why we chose the former in our experiments. The SVMLight timing code is specific to our experiments and is therefore located in the ecml2012 git branch of SHOGUN, which is available at: `http://bioweb.me/mtl-dcd`

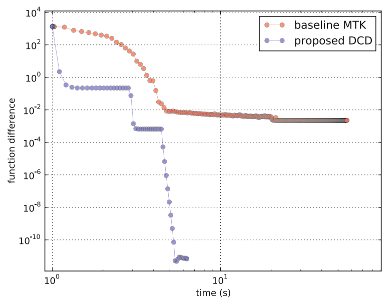
[3] `http://yann.lecun.com/exdb/mnist/`

Table 2. Statistics of the data sets used in this paper

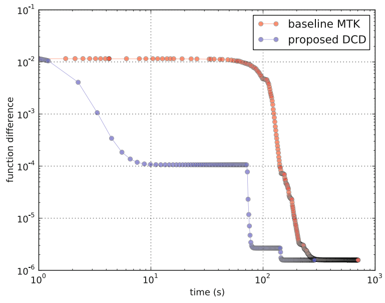
	dim	#examples	#tasks
Gauss2D	2	$1 \cdot 10^5$	2
Breast Cancer	44	474	3
MNIST-MTL	784	$9.0 \cdot 10^3$	3
Land Mine	9	$1.5 \cdot 10^4$	29
Splicing	$6 \cdot 10^6$	$6.4 \cdot 10^6$	4



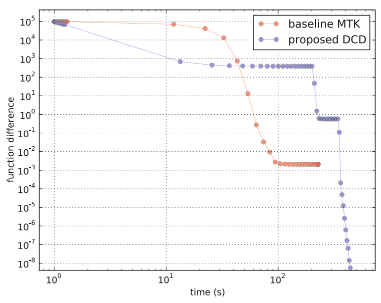
(a) Gauss2D



(b) Breast cancer



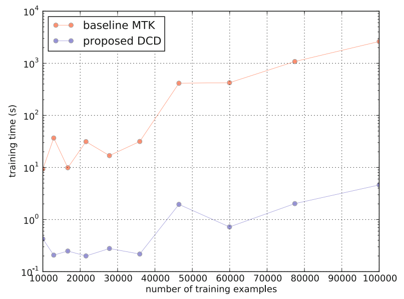
(c) MNIST-MTL



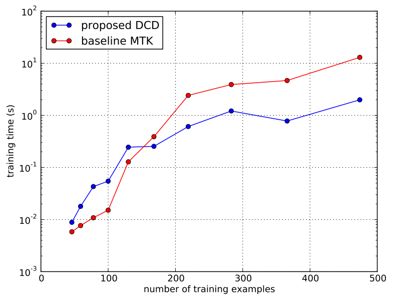
(d) Land Mine

Fig. 1. Results of the runtime experiment in terms of the function difference as a function of the execution time

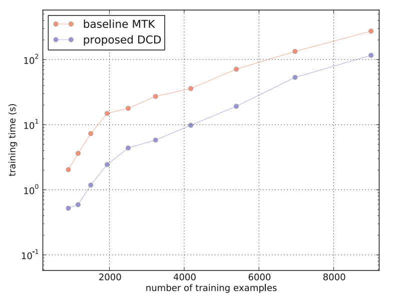
The results are shown in Figure 1, where the function difference of the four real-valued data sets is shown as a function of the execution time. First of all, we observe that in all four cases the two solvers suffer from an initialization phase, in which the function value improves only slowly. For *Gauss2D* the convergence properties of the two methods (e.g., steepness of the decrease in function difference) are very similar, but our proposed DCD solver being up to three magnitudes faster. Furthermore, we observe that, for two out the four data sets, the MTK baseline fails to decrease the function difference beyond a threshold ranging from 10^{-2} to 10^{-4}, while the proposed DCD algorithm nicely converges to



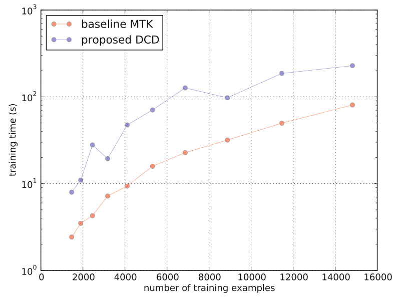
(a) Gauss2D



(b) Breast cancer



(c) MNIST-MTL



(d) Land Mine

Fig. 2. Results of the second runtime experiment: required time to train a multi-task SVM to a relative precision of 10^{-4} for various sample sizes n

a precision of 10^{-7} to 10^{-10} (cf. Figure 1 (b) – (d)). Finally, we can observe that if we stop both algorithms at some arbitrary time point, our method tends to output a solution that is more precise than the MTK baseline by usually several orders of magnitudes (up to ten orders for, e.g., *Gauss2D, and Breast Cancer*).

In a second experiment, we measure the training time a solver needs to reach a given precision (we chose 10^{-4}) as a function of the training set size. The results of this experiment are shown in Figure 2. We observe that for 3 out of 4 data sets, the proposed DCD methods requires less computation time than the MTK solver. For the synthetic data set the difference is the most drastic, being of the order of up to 2.5 magnitudes. Our method is outperformed by the MTK algorithm on the landmine data set (see Subfigure 2(d)), which indicates that our strategy is in disadvantage if the number of tasks is large relative the number of training examples, due to the update rule given by Equation 17. We expect the curves to cross if there are more training examples per task.

Finally, we study a very large splice data set, where the goal is to detect splice sites in various organisms, each organism corresponding to one task. For the MTK solver, the features are derived from raw DNA strings by means of a weighted-degree string kernel [21] of degree 8; for the DCD solver, we combine

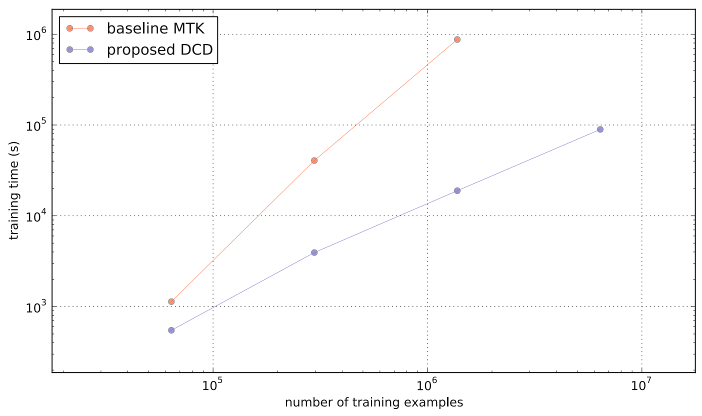


Fig. 3. Results of the large-scale splice site detection experiment

the proposed algorithmic methodology with the COFFIN framework [3] (efficient feature hashing for high-dimensional but sparse feature spaces) as implemented in SHOGUN [4].

The results of this experiment are shown in Figure 3. We observe that the proposed DCD solver is capable of dealing with millions of training points, while the MTK baseline is limited to rather moderate training set sizes of up to hundreds of thousands training points. This experiment demonstrates that we are now able to train on very large genomic sequences in reasonable time, finally allowing for truly large-scale multi-task learning.

5 Conclusion

We have introduced a dual coordinate descent method for graph-regularized multi-task learning. Unlike previous approaches, our optimization methodology is based on the *primal* formulation of the problem. Viewing the latter in terms of block vectors and subsequently deploying Fenchel-Legendre conjugate functions, we derived a general dual criterion allowing us to plug in arbitrary convex loss functions. We presented an efficient optimization algorithm based on dual coordinate descent and prove its convergence. Empirically, we show that our method outperforms existing optimization approaches by up to three orders of magnitude.

By including the recently developed COFFIN framework [3]—which devises feature hashing techniques for extremely high-dimensional feature spaces—into our methodology, we are able, to train a multi-task support vector machine on a splice data set consisting of over $1,000,000$ training examples and 4 tasks. An efficient C++ implementation of our algorithm is being made publicly available as a part of the SHOGUN machine learning toolbox [4].

Our new implementation opens the door to various new applications of multi-task learning in sequence biology and beyond, as it now becomes feasible to combine very large data sets from *multiple* organisms [22]. Our methodology may also serve as technological blueprint for developing further large-scale learning techniques in general: the block vector view gives insights into *structured* learning problems beyond the ones studied in the present paper and, combined with our novel dualization technique, we are able to also extend our optimization approach to various other structured learning machines such as, e.g., structured output prediction as proposed by [7] and block ℓ_p-norm regularized risk minimizers (e.g., [23]).

Acknowledgements. We would like to thank Alexander Zien, who contributed to the *cancer* data set and Jose Leiva for helpful discussions. This work was supported by the German National Science Foundation (DFG) under MU 987/6-1 and RA 1894/1-1 as well as by the European Communitys 7th Framework Programme under the PASCAL2 Network of Excellence (ICT-216886).

References

1. Hsieh, C., Chang, K., Lin, C., Keerthi, S., Sundararajan, S.: A dual coordinate descent method for large-scale linear SVM. In: Proceedings of the 25th International Conference on Machine Learning, pp. 408–415 (2008)
2. Fan, R., Chang, K., Hsieh, C., Wang, X., Lin, C.: LIBLINEAR: A library for large linear classification. Journal of Machine Learning Research 9, 1871–1874 (2008)
3. Sonnenburg, S., Franc, V.: Coffin: A computational framework for linear SVMs. In: Fürnkranz, J., Joachims, T. (eds.) ICML, pp. 999–1006. Omnipress (2010)
4. Sonnenburg, S., Rätsch, G., Henschel, S., Widmer, C., Behr, J., Zien, A., de Bona, F., Binder, A., Gehl, C., Franc, V.: The SHOGUN Machine Learning Toolbox. Journal of Machine Learning Research 11, 1799–1802 (2010)
5. Pan, S., Yang, Q.: A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering, 1345–1359 (2009)
6. Schweikert, G., Widmer, C., Schölkopf, B., Rätsch, G.: An Empirical Analysis of Domain Adaptation Algorithms for Genomic Sequence Analysis. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems 21, pp. 1433–1440 (2008)
7. Görnitz, N., Widmer, C., Zeller, G., Kahles, A., Sonnenburg, S., Rätsch, G.: Hierarchical Multitask Structured Output Learning for Large-scale Sequence Segmentation. In: Advances in Neural Information Processing Systems 24 (2011)
8. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: Cohen, W.W., McCallum, A., Roweis, S.T. (eds.) ICML. ACM International Conference Proceeding Series, vol. 307, pp. 160–167. ACM (2008)
9. Jiang, Y.G., Wang, J., Chang, S.F., Ngo, C.W.: Domain adaptive semantic diffusion for large scale context-based video annotation. In: ICCV, pp. 1420–1427. IEEE (2009)
10. Samek, W., Binder, A., Kawanabe, M.: Multi-task Learning via Non-sparse Multiple Kernel Learning. In: Real, P., Diaz-Pernil, D., Molina-Abril, H., Berciano, A., Kropatsch, W. (eds.) CAIP 2011, Part I. LNCS, vol. 6854, pp. 335–342. Springer, Heidelberg (2011)

11. Torralba, A., Murphy, K.P., Freeman, W.T.: Sharing visual features for multiclass and multiview object detection. IEEE Trans. Pattern Anal. Mach. Intell. 29, 854–869 (2007)
12. Evgeniou, T., Pontil, M.: Regularized multi-task learning. In: International Conference on Knowledge Discovery and Data Mining, pp. 109–117 (2004)
13. Agarwal, A., Daumé III, H., Gerber, S.: Learning Multiple Tasks using Manifold Regularization. In: Advances in Neural Information Processing Systems 23 (2010)
14. Evgeniou, T., Micchelli, C., Pontil, M.: Learning multiple tasks with kernel methods. Journal of Machine Learning Research 6(1), 615–637 (2005)
15. Cortes, C., Vapnik, V.: Support vector networks. Machine Learning 20, 273–297 (1995)
16. Müller, K.R., Mika, S., Rätsch, G., Tsuda, K., Schölkopf, B.: An introduction to kernel-based learning algorithms. IEEE Neural Networks 12(2), 181–201 (2001)
17. Joachims, T.: Making large–scale SVM learning practical. In: Schölkopf, B., Burges, C., Smola, A. (eds.) Advances in Kernel Methods — Support Vector Learning, pp. 169–184. MIT Press, Cambridge (1999)
18. Rifkin, R.M., Lippert, R.A.: Value regularization and Fenchel duality. J. Mach. Learn. Res. 8, 441–479 (2007)
19. Bertsekas, D.: Nonlinear Programming, 2nd edn. Athena Scientific, Belmont (1999)
20. Xue, Y., Liao, X., Carin, L., Krishnapuram, B.: Multi-task learning for classification with dirichlet process priors. J. Mach. Learn. Res. 8, 35–63 (2007)
21. Sonnenburg, S., Rätsch, G., Rieck, K.: Large scale learning with string kernels. In: Bottou, L., Chapelle, O., DeCoste, D., Weston, J. (eds.) Large Scale Kernel Machines, pp. 73–103. MIT Press, Cambridge (2007)
22. Consortium, T.W.T.C.C.: Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. Nature 447(7145), 661–678 (2007)
23. Kloft, M., Brefeld, U., Sonnenburg, S., Zien, A.: Lp-norm multiple kernel learning. Journal of Machine Learning Research 12, 953–997 (2011)