# Laplacian-based Semi-supervised Multi-Label Regression

Vivien Kraus
Université Lyon 1
43, Bd du 11 Novembre 1918
Villeurbanne, Cedex 69622, France
vivien.kraus@lizeo-group.com

Khalid Benabdeslem
Université Lyon 1
43, Bd du 11 Novembre 1918
Villeurbanne, Cedex 69622, France
kbenabde@univ-lyon1.fr

Bruno Canitia
Lizeo Group
42 Quai Rambaud, 69002 Lyon
Lyon, 69002, France
bruno.canitia@lizeo-group.com

***Abstract*—In multi-label learning, one has to find a model suitable for predicting multiple values for the same individual, based on the same features. The effectiveness of most multi-label algorithms lies in the fact that it is able to consider the correlations between the related labels. On the other hand, in many applications, a multi-label learning task incurs a high cost for the annotation of a single data point. This leads to a dataset consisting of a few labeled data points, and many more unlabeled data points. In this scenario, semi-supervised methods can take advantage of the unlabeled data points. In this article, we propose a new algorithm for multi-label semi-supervised regression, LSMR, as a multi-label extension of a semi-supervised regression algorithm. We provide experimental results on some publicly-available regression datasets showing the effectiveness of our approach.**

## I. Introduction

One of the key challenges of modern machine learning approaches is to learn from both hand-annotated label data as well as unlabeled data, which is generally considered easy to obtain. Weakly labeled learning [1], and more importantly semi-supervised learning [2], [3] addresses this challenge as it may either use *label propagation* to assist supervised methods [3], [4], or use the label information as *constraints* in unsupervised methods [5], [6]. Many types of semi-supervised methods based on supervised methods have been proposed. For instance, adaptations such as *self-training* [2] or *co-training* [7], [8] have been studied, *transductive* learning which aims at using the information of the input distribution of the test dataset has notably been implemented for the Support vector machine [9], [10], and generative methods have also been adapted [11]. For regression problems, with a numeric prediction, specific applications have been proposed [12], [13], however many applications rely either on input space representation learning [14], or on a graph between data points constructed with an input-based similarity measure [15], with an emphasis on the Laplacian regularization [16], [17].

Moreover, multi-label learning aims at exploiting the relations between labels in order to use more information for the learning task. Some theoretical works have been proposed. In [18], the authors decompose the set of classification labels as independent groups in the context of bayesian networks. Many multi-label methods have been proposed [19], [20]. It is not surprising that many multi-label algorithms are built upon existing methods. Some strategies transform a multi-label problem to one or many single-label problems. The *Classifier Chain* approach is a simple algorithm that can take advantage of the relations between labels in order to provide a stronger learner [21]. Given the initial dataset, a first weak learner is trained to predict one of the multiple labels. Then, a new dataset is constructed, by integrating the predicted label as a new feature, and training a slightly stronger learner over the new dataset to predict one of the other labels. By repeating this procedure until the last label, a multi-label classifier is constructed. With some modifications, this gives a good multi-label learning algorithm [22]. Some of the traditional approaches extend to the multi-label case, such as the Gaussian processes [23].

However, many algorithms that are suitable for multi-label learning are defined as the minimization of a regularized objective function. Among these regularizations, we can set apart two groups: some of these regularizations are designed to be *convex*, while others, more recent approaches, focus on non-convex formulations. Non-convex functions, such as the capped $l_1$ norm [24], or more recently the $l_{2,1-2}$ regularization [25], are found to have good performance, although the convergence of the algorithm only guarantees that the loss converges, or that the model converges (which is very desirable).

In other cases, the non-convex formulation may lead to a convex relaxation. By using a $l_0$ regularization in mono-label learning, one can regularize the number of features that are needed to perform the prediction, even if this regularization is nonconvex. The LASSO algorithm [26] is simply a convex relaxation of this term, and the same can be applied to multi-label feature selection [27], [28]. Likewise, the Clustered Multi-Task Learning framework benefits from a convex relaxation [29]. Other regularizations are naturally convex, such as those used for other label space representation learning [30], [31]. In an effort to bring these regularization frameworks under a same optimization strategy, the MALSAR implementation has been proposed [32].

Simultaneous semi-supervised learning and label-space representation learning with two different graphs for label similarity and data point similarity has already received some attention [33].

In the regression settings, few methods exist. The usual approach is to change the loss function in order to accept real-valued labels [32], [34]. In this paper, we derive a new semi-supervised algorithm for multi-label regression.

## II. Related works

The SSSL algorithm (a Simple algorithm for Semi-Supervised Learning, [14]) is a two-step procedure to train a semi-supervised regressor. In the first step, an unsupervised change of space is performed on the data. Then, a simple linear supervised regressor is trained. While this approach is commonly adopted to real-world applications, the authors in [14] showed that it should be part of the whole data processing operation. More specifically, under some hypothesis about the dataset and the number of taken labeled points, the generalization will be better than any of the models for any supervised learning algorithms. Unfortunately, these hypothesis are impossible or very difficult to check in practice.

The algorithm is composed of two steps: an unsupervised step and a fully supervised step. The former exctract features independently of the label data. The latter only occurs in the space of the extracted features, by considering the labeled data. So this approach clearly transforms the semi-supervised problem into a succession of an unsupervised problem and a supervised one.

This approach could however fail to generalize for some applications in practice. To solve this problem, a regularized adaptation of this algorithm has been developed in [35]. This approach, called LapS3L consists of a generalization of the SSSL algorithm. Thus, proper tuning of the hyperparameters is necessary to handle the cases for which the SSSL performs poorly, and sufficient to increase or keep the same performance. In this approach, the expectation for both learning steps is slightly different: while the original SSSL algorithm separated an unsupervised step and a supervised learning step, LapS3L was designed to regularize the linear regression with a semi-supervised regularization.

The used regularization is Laplacian-based. For semi-supervised problems, it aims to enforce the following assumption: if two data points are connected in a similarity graph, then the distance between their predictions should be minimized. Given a data matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$, a graph of instances is constructed, with instances as nodes and feature-based similarity between instances as edges, with a similarity value normalized between 0 and 1. The graph is encoded as an adjacency matrix $\mathbf{M} \in \mathbb{R}^{n\times n}$, and the degree matrix is the diagonal $\mathbf{D} \in \mathbb{R}^{n\times n}$. With such a symmetric graph, $\mathbf{L} = \mathbf{D} - \mathbf{M}$ is a semidefinite positive matrix. For a single-label model $w \in \mathbb{R}^d$, the convex regularization term is ${}^t w\mathbf{X}\mathbf{L}\mathbf{X}w = \sum_{i,j} \mathbf{M}_{ij}\left[(\mathbf{X}w)_i - (\mathbf{X}w)_j\right]^2$.

The introduction of a graph over the data with the similarity between instances forming the edges, is thus a new semi-supervised step after the unsupervised one. The unsupervised step uses a decomposition of a kernel function which is applied between all input instances, like the Laplacian regularization. The difference lies in the fact that both steps do not use the same data features. Thus, this additional regularization helps taking into account both similarity graphs: the one based on the raw data, and the one based on the transformed data.

This Laplacian regularization can also be employed to cope with multi-label learning [36]. In this case, the idea is that two related labels should give two similar models. The regularization leverages a similarity graph between labels, and penalizes a model $\mathbf{W} \in \mathbb{R}^{d\times m}$ by $\mathrm{tr}(\mathbf{W}\mathbf{L}^t\mathbf{W})$, where $m$ is the number of regression labels. This penalization is designed to have similar models for similar labels.

In this paper, we propose to keep extending the SSSL, so as to keep at least the same performance with proper tuning, but dealing with multi-label tasks.

## III. Proposed approach

In this section, we present the details of our approach, LSMR (for Laplacian-based Semi-supervised Multi-label Regression) as a regularization of the SSSL algorithm. We use a regularized convex objective function in a different feature space, with a similar approach to the LapS3L. We thus propose a strict generalization of the SSSL.

As with SSSL, the objective function uses the features from the eigenvector decomposition of the input kernel. So, given the input data as a matrix $\mathbf{V}$ of dimensions $n \times d$, the kernel matrix $\mathbf{K}$ is formed, such that for all $i, j \in \{1, \dots, n\}$,

$$\mathbf{K}_{i,j} = \phi(\mathbf{V}_{i,.}, \mathbf{V}_{j,.}) \tag{1}$$

In this formulation, $\phi$ is a kernel function. It takes two individuals and return a scalar.

As $\mathbf{K}$ is a real symmetric matrix, it has an eigenvector decomposition with real eigenvalues. Let $\sigma \in \mathbb{R}^s$ the $s$ largest eigenvalues and $\mathbf{U} \in \mathbb{R}^{n\times s}$ the associated eigenvectors. The features of interest, $\mathbf{X} \in \mathbb{R}^{n\times s}$, are simply computed as $\mathbf{KU}$. The learning problem becomes to find a model $\mathbf{W} \in \mathbb{R}^{s\times m}$ minimizing the proposed objective function, (2). It is important to note that each row of $\mathbf{X}$ corresponds to exactly one input point; so in the context of semi-supervised learning where some instances are not labeled, then the corresponding rows in $\mathbf{X}$ do not have a label. This lets us keep the same labels $\mathbf{Y}$ in the new space.

The prediction code for a new batch of $n_t$ instances $\mathbf{V_b} \in \mathbb{R}^{n_t\times d}$ is slightly more complex. Let $\mathbf{K_b} \in \mathbb{R}^{n\times n_t}$ the kernel as computed between the training data and the test batch: for all $i \in \{1, \dots, n\}, j \in \{1, \dots, n_t\}$, $\mathbf{K_b}_{i,j} = \phi(\mathbf{V}_{i,.}, \mathbf{V_b}_{j,.})$. Then, the prediction is generated as: $\hat{\mathbf{Y}}_\mathbf{b} = {}^t\mathbf{\Phi}W$ with $\mathbf{\Phi} = \mathbf{\Sigma}^{-\frac{1}{2}}{}^t\mathbf{U}\mathbf{K_b}$ and $\mathbf{\Sigma} = \mathrm{diag}(\sigma)$.

### A. Objective function

The objective function that we use is composed of three weighted terms: the least squares term, which is minimized for a perfect prediction on the training set, the semi-supervised regularization term which penalizes the differences in predictions on related instances, and the multi-label regularization extension.

$$\mathcal{L}(\mathbf{W}) = \|\mathbf{X_l}\mathbf{W} - \mathbf{Y_l}\|_F^2 + \lambda_s\Omega_s(\mathbf{W}) + \lambda_m\Omega_m(\mathbf{W}) \tag{2}$$

with $\mathbf{X_l} \in \mathbb{R}^{n_l \times s}$ containing only the labeled rows of $\mathbf{X}$, and $\mathbf{Y_l} \in \mathbb{R}^{n_l \times m}$ containing the corresponding rows in $\mathbf{Y}$.

The regularization term for semi-supervised learning $\Omega_s(\mathbf{W})$ is controlled by an hyperparameter $\lambda_s$, and the regularization term for multi-label learning $\Omega_m(\mathbf{W})$ is controlled by another hyperparameter $\lambda_m$.

Both regularization terms are shaped with the help of a graph Laplacian matrix. The graph Laplacian matrices are $\mathbf{L_s} \succcurlyeq \mathbf{0}$ for the semi-supervised regularization and $\mathbf{L_m} \succcurlyeq \mathbf{0}$ for the multi-label regularization.

$$\begin{cases}\Omega_s(\mathbf{W}) = \text{trace}({}^t\mathbf{W}{}^t\mathbf{X}\mathbf{L_s}\mathbf{X}\mathbf{W}) \\ \Omega_m(\mathbf{W}) = \text{trace}(\mathbf{W}\mathbf{L_m}{}^t\mathbf{W})\end{cases} \tag{3}$$

The Laplacian matrix encodes a graph. In the case of the semi-supervised regularization, this graph involves instances as nodes and similarities between them as edges. Let $\mathbf{M_s} \in \mathbb{R}^{n \times n}$ the adjacency matrix of this graph. For any two instances $i$ and $j$, if an edge exists between $i$ and $j$, then $\mathbf{M_s}_{i,j}$ is the weight of this edge. Then,

$$\Omega_s(\mathbf{W}) = \sum_{k=1}^{m} \left[ \sum_{i_1=1, i_2=1}^{n} \mathbf{M}_{s i_1, i_2} \left[ (\mathbf{XW})_{i_1,k} - (\mathbf{XW})_{i_2,k} \right]^2 \right] \tag{4}$$

This regularization term is intuitive: it means that for a label, we penalize the solution according to how much related instances differ in their predictions.

In the case of the multi-label regularization, the graph involves labels as nodes and similarities between these points as edges.

$$\Omega_m(\mathbf{W}) = \sum_{j=1}^{s} \left[ \sum_{k_1=1, k_2=1}^{m} \mathbf{M_m}_{k_1,k_2} \left( \mathbf{W}_{j,k_1} - \mathbf{W}_{j,k_2} \right)^2 \right] \tag{5}$$

The solution is thus penalized such that the solution assigns the same weights to related labels.

### *B. Optimization algorithm*

While an analytical solution to this problem is possible [37], it requires solving $m^2$ mono-label problems. However, more recent works (MALSAR, [32]) have explored similar problems with gradient descent, and especially the Accelerated Gradient Descent as published by Nesterov.

Indeed, the objective function (2) is convex, as a sum of three convex functions. The gradient is given by eq (6).

$$\begin{aligned}\nabla_W \mathcal{L}(\mathbf{W}) = 2 \left[ {}^t\mathbf{X_l}\mathbf{X_l} + \lambda_s {}^t\mathbf{X}\mathbf{L_s}\mathbf{X} \right] \mathbf{W} \\ -2{}^t\mathbf{X_l}\mathbf{Y_l} + 2\lambda_m \mathbf{W}\mathbf{L_m}\end{aligned} \tag{6}$$

The gradient descent algorithm iteratively updates a model $\mathbf{W}$ in the opposite direction of the gradient at $\mathbf{W}$, by a factor $\eta$ called the *learning rate*. While $\eta$ is considered as a hyperparameter in the general case, eq (6) is Lipschitz-continuous: for all couples of models $P, \mathbf{Q} \in \mathbb{R}^{s \times m}$,

$$\left\| \nabla_{\mathbf{W}} \mathcal{L}(\mathbf{P}) - \nabla_{\mathbf{W}} \mathcal{L}(\mathbf{Q}) \right\|_F^2 \leq C \left\| \mathbf{P} - \mathbf{Q} \right\|_F^2 \tag{7}$$

with

$$C = 2 \left( \rho \left( {}^t\mathbf{X_l}\mathbf{X_l} + \lambda_s {}^t\mathbf{X}\mathbf{L_s}\mathbf{X} \right) + \lambda_m \rho \left( \mathbf{L_m} \right) \right) \tag{8}$$

where for all real symmetric matrix $\mathbf{M}$, $\rho(\mathbf{M})$ is the spectral radius of $\mathbf{M}$. By setting $\eta = \frac{1}{C}$, we get a suitable learning rate to ensure convergence of the gradient descent [38]. Thus, the whole mathematical development is summarized in Algorithm 1.

**Algorithm 1** **LSMR**

```
Input: data: $\mathbf{V} \in \mathbb{R}^{n \times d}$, $\mathbf{Y} \in \mathbb{R}^{n \times m}$; hyperparameters:
    $\phi \colon \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, $\lambda_s \geq 0$, $\lambda_m \geq 0$, $s \in \{1, \dots, n\}$
Output: Prediction function
  Construct $\mathbf{K} \succcurlyeq \mathbf{0}$ as eq (1)
  Compute the $s$ largest eigenvalues $\sigma \in \mathbb{R}^s$ and associated eigenvectors $\mathbf{U} \in \mathbb{R}^{n \times s}$
  $\mathbf{X} \leftarrow \mathbf{KU}$, $\mathbf{X_l} \in \mathbb{R}^{n_l \times s}$ and $\mathbf{Y_l} \in \mathbb{R}^{n_l \times m}$ the labeled data points
  Estimate $C$ from eq (8)
  Initialize $\mathbf{W} \in \mathbb{R}^{s \times m}$ with random values
  Apply the Accelerated Gradient Descent on $\mathbf{W}$ with fixed step $\frac{1}{C}$, and with the gradient from eq (6)
  function BATCH PREDICTION($\mathbf{V_b} \in \mathbb{R}^{n_t \times d}$)
      $\forall i \in \{1, \dots, n\}, j \in \{1, \dots, n_t\}, \mathbf{K_b}_{i,j} \leftarrow \phi(\mathbf{V}_{i,.}, \mathbf{V_b}_{j,.})$
      return $\hat{\mathbf{Y}}_\mathbf{b} = \mathbf{K_b}\mathbf{U}\mathbf{W}$
  end function
```

The algorithm respects the two-step procedure, by performing a change of space, then optimizing the regularized objective function. Since the objective function gives a model operating in the changed space, the prediction function cannot be a simple matrix product.

## IV. EXPERIMENTS

In order to show the effectiveness of our proposed approach, we conducted two parts experiments. First, as our porposed method is an extension of existing works, we show experimental data validating the need for that extension. Second, we do some comparisons with different regularizations for multi-label regression.

### *A. Datasets*

We used a subset of the multi-label regression data from the MULAN project [1], and added a sample of 1000 individuals from the SARCOS dataset [39]. In order to save time for the extensive tuning, we did not include the datasets that had more than a thousand individuals. The characteristics of the datasets are summarized in table I.

The datasets are randomly split into a train and test parts. To simulate a semi-supervised setting, we randomly keep the

[1]http://mulan.sourceforge.net/datasets-mtr.html

labels for 30% of the individuals, and consider these as the labeled data. This is more labeled individuals than what is usual, but since some of the datasets only have a few data points, we cannot discard too many labels.

Every feature and every label from the dataset is centered and scale so that its mean is set to 0 and its variance to 1. Feature normalization is needed because each feature needs to be taken into account in the kernel function. We need to normalize the labels for a meaningful computation of the averaged metric over all labels.

### B. Experimental setting

For each dataset, the hyperparameters are selected via a random search [40], instead of the more traditional grid search. In order to prioritize the most important hyperparameters, *random search* samples all hyperparameters. The set of hyperparameters that gives the lowest average RMSE between the expected labels $\mathbf{Y_t} \in \mathbb{R}^{n_t \times m}$ and the predictions $\hat{\mathbf{Y}} \in \mathbb{R}^{n_t \times m}$ averaged over a 10-fold cross-validation on the training set. The average RMSE is defined as:

$$\text{aRMSE} = \frac{1}{m} \sum_{k=1}^{m} \sqrt{\frac{\sum_{i=1}^{n_t} \left( \mathbf{Y_t}_{i,k} - \hat{\mathbf{Y}}_{i,k} \right)^2}{n_t}} \tag{9}$$

This metric indicates a perfect fit if $\text{aRMSE} = 0$, and in the case where all labels are centered and scaled, a naive predictor returning a constant 0 has $\text{aRMSE} = 1$.

Only the labeled dataset is split in cross-validation folds. The unlabeled data are used untouched for all folds.

Our proposed approach has 4 hyperparameters:

- the kernel function $\phi$;
- the number of components $s$;
- the Laplacian regularizer $\lambda_s$;
- the multi-label regularizer $\lambda_m$.

We explored three different kinds of kernel functions:

- the linear kernel;
- the cosine kernel (10);
- the Radial Basis Function kernel (11), with a new hyperparameter $\gamma$ (the hyperparameter $\gamma$ is sometimes written as $\frac{1}{2\sigma^2}$, where $\sigma > 0$).

$$\phi(v, v') = \frac{\langle v, v' \rangle}{\|v\| \, \|v'\|} \tag{10}$$

$$\phi(v, v') = e^{-\gamma \|v - v'\|^2} = e^{-\frac{\|v - v'\|^2}{2\sigma^2}} \tag{11}$$

### C. Results with local tuning

Since our proposed approach is an extension combining two regularizations, we first check that the combination gives a better aRMSE value when sharing the values for the hyperparameters of the tuned SSSL.

In order to check this, we first tune the SSSL algorithm for all datasets, and store the hyperparameter values in table II. Depending on the dataset, we may use different kernels for the first unsupervised step, or select a number of components from the decomposition of the kernel matrix. These values have been obtained by sampling the search space for the kernel type, the kernel bandwidth (only applicable for the RBF kernel), and the number of components. The hyperparameter values are those that lead to the best aRMSE value calculated within a 10-fold cross-validation mean.

Given this set of hyperparameters, we then tune our proposed approach by re-using the applicable hyperparameters, i.e. the kernel type, kernel bandwidth and number of components, but simply introduce our two regularization terms and tune the associated values. These new hyperparameters are sampled on a logarithmic scale, for small values. For each test, we record the aRMSE value for the SSSL alone, i.e. with both regularizers set to exactly 0, and for our proposed approach, i.e. with both regularizers having a non-zero value. Since SSSL is run as many times as the tuning for our instances, the aRMSE for the SSSL over each dataset is averaged, just as if we tested the SSSL on these hyperparameters several times.

For each dataset, we thus get a heat map of the performance of our combined approach relative to the performance of the SSSL. Subsequent graphs are heat maps of the relative performance with respect to the value of both regularizers. Areas with a red color indicate hyperparameter values for which our approach gives better results than SSSL, and blue zones indicate that the performance is degraded.

For the sake of argument, we present a few of these tuning maps in order to show the different cases we might get.

Figure 1 represents this gain for the `scpf` dataset. This is a difficult dataset, as the multi-label regularizer quickly degrades performance, while the optimal tuning region is very restricted. On the other hand, the `osales` dataset 3 seems to perform better with a very low value of the semi-supervised regularizer $\lambda_s$. However, other datasets such as `sf2` (figure 2) generally show that with an appropriate tuning of both regularizers, the performance can be improved. Finally, figure 4 on the `oes97` dataset shows a limit of the approach of starting from the tuned SSSL: if a regularizer is too large, then the predictions from our approach are so different from the baseline that it becomes impossible to compare them with the same kernel and transformed space. This is why we need to tune our hyperparameters all at the same time.

In summary, table III shows the distribution of the relative gain for our approach over all datasets, with the mean improvement, the first and third quartiles, and the best and worst value.

### D. Comparative study

In a second time of experiments, we compared our proposed approach with state-of-the-art multi-label regression methods on the same datasets. All these methods solve a convex regularized regression function with an accelerated gradient descent. Some of the objective functions use a matrix norm. Unfortunately, the definitions are different for [27] and [41]. We adopt these definitions: for $\mathbf{W} \in \mathbb{R}^{s \times m}$,

TABLE I: Used data sets

| Dataset | #Train (#labeled, #unlabeled) | #Test | #Features | #Labels |
|---|---|---|---|---|
| atp1d | 262 (76 + 186) | 165 | 411 | 6 |
| atp7d | 234 (67 + 167) | 147 | 411 | 6 |
| edm | 121 (35 + 86) | 73 | 16 | 2 |
| enb | 601 (173 + 428) | 366 | 8 | 2 |
| jura | 281 (81 + 200) | 173 | 15 | 3 |
| oes10 | 314 (91 + 223) | 198 | 298 | 16 |
| oes97 | 257 (75 + 182) | 163 | 263 | 16 |
| osales | 495 (144 + 351) | 309 | 401 | 12 |
| sarcossub | 779 (225 + 554) | 467 | 21 | 7 |
| scpf | 889 (256 + 633) | 521 | 23 | 3 |
| sf1 | 250 (73 + 177) | 158 | 31 | 3 |
| sf2 | 832 (240 + 592) | 501 | 31 | 3 |
| wq | 827 (238 + 589) | 487 | 16 | 14 |

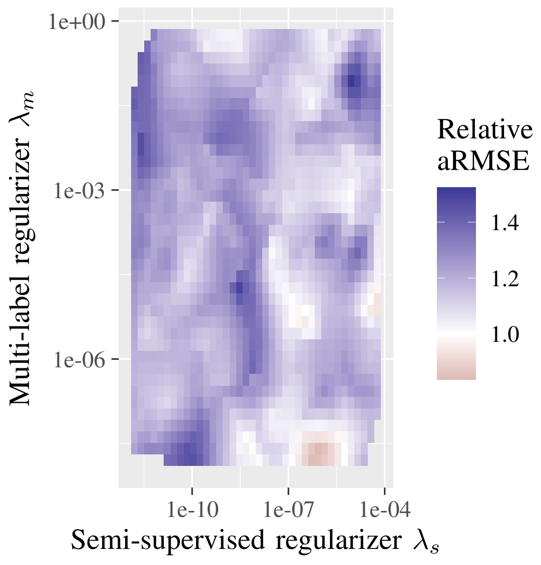

Fig. 1: Relative gain on the scpf dataset

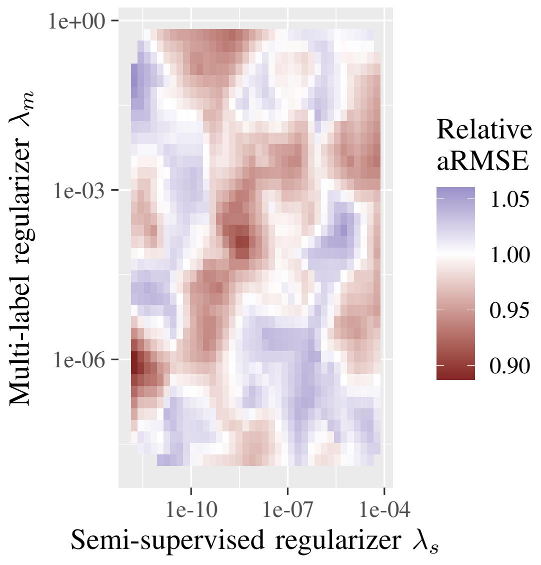

Fig. 3: Relative gain on the osales dataset

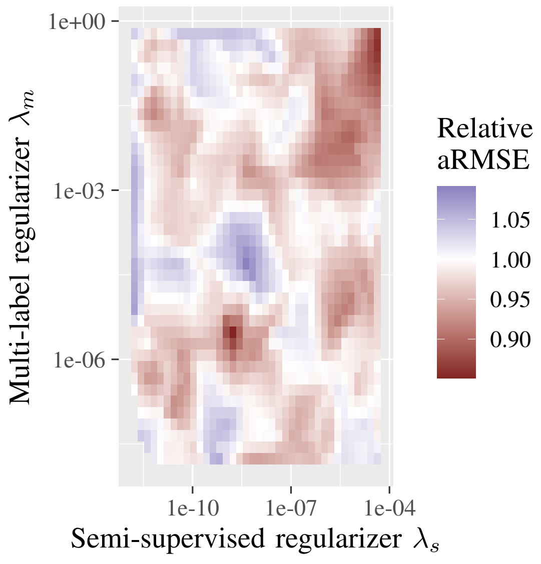

Fig. 2: Relative gain on the sf2 dataset

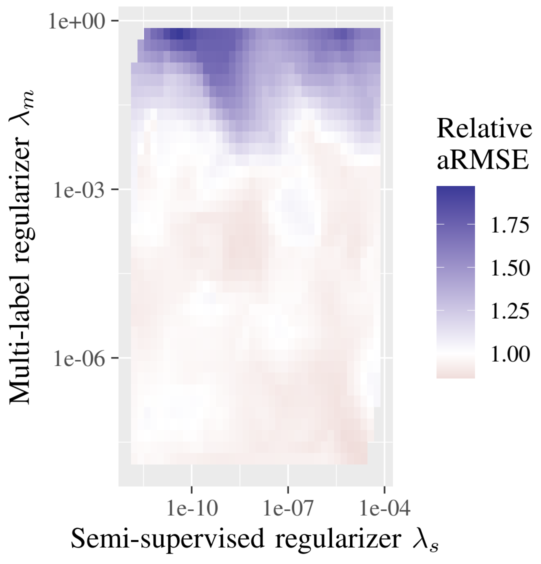

Fig. 4: Relative gain on the oes97 dataset

TABLE II: Tuning of the SSSL algorithm

| **Dataset** | **Kernel** | **Components** |
|---|---|---|
| atp1d | rbf ($\gamma = 3.140 \cdot 10^{-05}$) | 202 |
| atp7d | linear | 5 |
| edm | rbf ($\gamma = 1.703 \cdot 10^{+01}$) | 107 |
| enb | cosine | 10 |
| jura | rbf ($\gamma = 4.272 \cdot 10^{-03}$) | 87 |
| oes10 | linear | 31 |
| oes97 | rbf ($\gamma = 3.785 \cdot 10^{-05}$) | 55 |
| osales | rbf ($\gamma = 1.254 \cdot 10^{-02}$) | 475 |
| scpf | cosine | 2 |
| sf1 | linear | 5 |
| sf2 | cosine | 3 |
| wq | rbf ($\gamma = 1.545 \cdot 10^{+01}$) | 774 |

TABLE III: Relative local improvements on the datasets

| **Dataset** | Mean | Q1 | Q3 | Best | Worst |
|---|---|---|---|---|---|
| sf2 | **0.983** | **0.898** | 1.086 | **0.553** | 1.245 |
| scpf | 1.191 | **0.862** | 1.554 | **0.564** | 1.684 |
| osales | **0.989** | **0.938** | 1.046 | **0.635** | 1.191 |
| oes97 | 1.078 | **0.872** | 1.170 | **0.662** | 2.344 |
| sf1 | **0.994** | **0.922** | 1.062 | **0.682** | 1.206 |
| oes10 | 1.015 | **0.908** | 1.110 | **0.715** | 1.448 |
| jura | 1.014 | **0.954** | 1.065 | **0.799** | 1.485 |
| atp1d | 1.048 | **0.958** | 1.070 | **0.812** | 1.922 |
| edm | **0.998** | **0.964** | 1.037 | **0.821** | 1.190 |
| atp7d | 1.006 | **0.961** | 1.053 | **0.833** | 1.242 |
| enb | **0.998** | **0.964** | 1.027 | **0.864** | 1.124 |
| wq | **0.999** | **0.992** | 1.007 | **0.960** | 1.032 |

$$\begin{cases} \|\mathbf{W}\|_{1,\infty} = \sum_{j=1}^{s} \|\mathbf{W}_{j,.}\|_{\infty} \\ \|\mathbf{W}\|_{1,2} = \sum_{j=1}^{s} \|\mathbf{W}_{j,.}\|_{2} \\ \|\mathbf{W}\|_{*} = \sum_{\sigma \text{ singular value of } \mathbf{W}} \sigma \end{cases} \quad (12)$$

The other methods tested are the baseline unregularized least squares `LSQ`, the Lasso regularization, `MTL` [26], the multi-label Laplacian regularization, `SGR` [36], the group Lasso regularization, `JFS` [27], the Dirty model, `DM` [41], the low-rank regularization, `TNR` [42] and the Clustered Multi-Task Learning `CMTL` [43]. The methods introduce new regularizers.

The results shown in table IV are multiple fold:

- our proposed approach, if tuned globally, performs better than the SSSL. The local tuning is however impaired, as the search space for the regularizers is too far away from zero, as discussed previously;
- our proposed approach performs better than the Laplacian multi-label regularization alone (SGR) on many datasets;
- the global tuning of our algorithm gives almost always better results than if it is tuned from the best SSSL run;
- unlike the original SSSL algorithm, our proposed extension gives competitive results to the state of the art for the class of objective functions we considered.

The significance of these results needs to be questioned. According to [44], there are several points to check in order to choose the methodology to compare these multiple algorithms over multiple datasets.

The first question to address is whether paired t-tests can apply. According to this methodology, we could compare algorithms by pairs, by considering the difference between their performance across all datasets as a normal distribution, and test whether we can conclude over the sign of the mean of the difference using a t-test. While this approach is quite simple to set up, the hypothesis are quite strong.

The most obvious problem is that under such analysis, for two algorithms, the difference is supposed to be a Gaussian sample. We have no means to verify this, and the number of datasets is clearly not enough to accomodate for this : we need at least 30 datasets, and we have only 13.

On the other hand, due to the fact that the selected datasets have a very small number of individuals, there are situations in which the test data may differ from the learning data. To tackle this issue, we run the tests 20 times, each time selecting a different subset for the training set and thus for the test set. However, the final results are averaged. The resulting poor choices of training sets may thus still appear in the averaged results. Since this phenomenon depends on the dataset, the variance of the performance between the algorithms varies according to the dataset. Reasoning about the mean of the difference of the results between two algorithms on each dataset may therefore not be adequate.

We could apply the Wilcoxon signed-ranks test. In this test, for each pair of algorithms, the difference of performance for each dataset is taken and ranked according to their absolute value. Then, the ranks corresponding to the datasets in favor of the first algorithm (respectively the second algorithm) are summed, and this sum is compared. Thus, we do not consider the average of the performance on a dataset, but a ranking. This lets us remove the first problem, but the second still holds: the test gives too much importance on the small datasets. In such situation, the pairwise comparison is not very informative; using a value of $\alpha = 0.1$ (as in figure 5) gives some more instances, from which we can still infer that our generalization gives better results than the SSSL, but it does not say much more.

For a multi-classifier analysis, we are interested in the non-parametric Friedman test. Under such a test, all classifiers are ranked for each dataset. The problems that we noticed in our comparison are accounted for under this test, because we do not compare any difference in performance to any other difference, be it for a difference between algorithms given dataset or for a difference between datasets given algorithm. In this way, the datasets that have a high variance in the performance of the algorithms are treated the same as the more dense datasets, so we do not risk favoriting them too much.

The Friedman test tries to reject the hypothesis that the mean ranks of all datasets are equal. Under a risk of $\alpha = 0.05$, this hypothesis is rejected, and we can proceed to compare the ranks of the classifiers.

If the mean ranks are separated from at least a certain critical distance depending on the number of classifiers and datasets, then we can run the Nemenyi test, which tries to reject the post-hoc hypothesis that these classifiers are equivalent. The diagram in figure 6 displays the average ranks for all classifiers on the datasets, with the critical distance. This diagrams shows that even if our proposed approach is not

TABLE IV: aRMSE Results for the global tuning

| Dataset | LSMR | Local LSMR | SSSL | CMTL | DM | JFS | LSQ | MTL | SGR | TNR |
|---|---|---|---|---|---|---|---|---|---|---|
| atp1d | **0.472** | 1.006 | 0.481 | 0.533 | 0.522 | 0.543 | 0.549 | 0.534 | 0.548 | 0.548 |
| atp7d | 0.888 | **0.713** | 0.779 | 0.727 | 0.782 | 0.764 | 0.787 | 0.787 | 0.787 | 0.787 |
| edm | **0.841** | 1.005 | 0.895 | 0.849 | 0.857 | 0.856 | 1.198 | 0.855 | 1.207 | 0.852 |
| enb | **0.320** | 0.541 | 0.339 | 0.332 | 0.333 | 0.332 | 0.332 | 0.332 | 0.332 | 0.332 |
| jura | 0.661 | 0.847 | 0.727 | 0.597 | 0.593 | **0.591** | 0.609 | 0.593 | 0.611 | 0.598 |
| oes10 | **0.390** | 0.488 | 0.395 | 0.398 | 0.402 | 0.402 | 0.402 | 0.402 | 0.402 | 0.402 |
| oes97 | **0.445** | 0.918 | 0.494 | 0.515 | 0.510 | 0.534 | 0.534 | 0.534 | 0.534 | 0.534 |
| osales | 1.012 | 0.957 | 0.938 | 0.882 | **0.839** | 0.861 | 0.921 | 0.873 | 0.873 | 0.906 |
| sarcossub | 0.363 | 0.816 | 0.428 | 0.357 | 0.357 | **0.354** | 0.357 | 0.354 | 0.357 | 0.357 |
| scpf | 1.111 | 0.926 | 1.253 | **0.621** | 0.636 | 0.635 | 0.635 | 0.635 | 0.635 | 0.635 |
| sf1 | 1.070 | **0.998** | 1.092 | 0.999 | 0.999 | 0.999 | 1.292 | 0.999 | 1.285 | 0.999 |
| sf2 | **0.973** | 1.041 | 1.114 | 1.022 | 1.161 | 1.024 | 1.262 | 1.024 | 1.249 | 1.024 |
| wq | 0.999 | 0.999 | 1.003 | **0.946** | 1.006 | 0.982 | 1.079 | 0.947 | 1.079 | 1.023 |

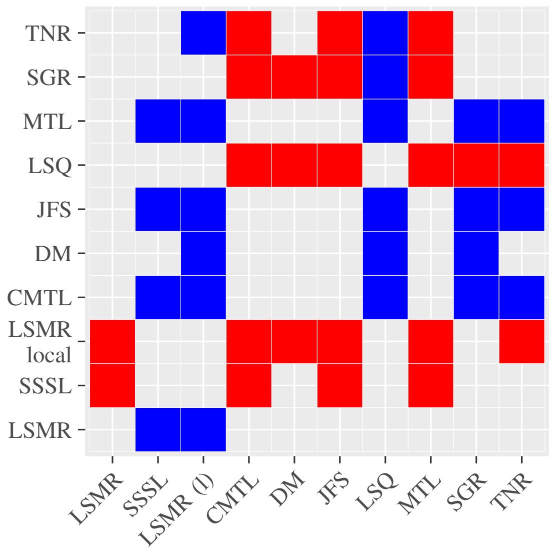

Fig. 5: Comparison of all pairs of algorithms, using the Wilcoxon signed-ranks test. For each algorithm row, blue demonstrate that algorithm in the row outperforms the algorithm in the column, under a risk $\alpha = 0.1$; red shows that the algorithm in the column outperforms the algorithm in the row. Gray cells are inconclusive.

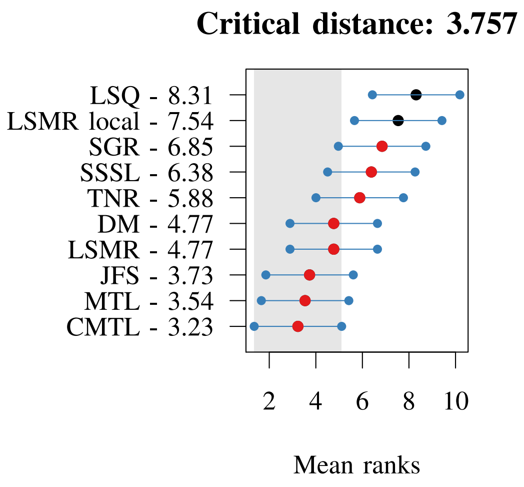

Fig. 6: Plot of the ranks of the algorithm with the Critical Distance

ranked best on average, it is still within the critical distance of the best, contrary to the other methods from which we draw our ideas: the special case SSSL, and the multi-label Laplacian regularization SGR.

## V. Conclusion & Future works

In this paper, we proposed to extend a semi-supervised regression algorithm to a multi-label framework. We experimentally showed that our approach, as a strict generalization of the SSSL, gives better results than the SSSL, even if not tuning the common hyperparameters. By tuning all hyperparameters, our method is competitive with other state-of-the-art algorithms.

The future works will be investigating whether we can adapt our approach to perform multi-label feature selection. Since the proposed approach works on extracted features, this will be a challenge. We will also investigate learning from another label space. Finally, we will also consider non-convex regularizations for our approach.

## References

[1] Y.-F. Li, I. W. Tsang, J. T. Kwok, and Z.-H. Zhou, "Convex and scalable weakly labeled svms," *Journal of Machine Learning Research*, vol. 14, pp. 2151–2188, 2013. [Online]. Available: http://jmlr.org/papers/v14/li13a.html

[2] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning*. MIT Press, 2006.

[3] X. Zhu, "Semi-supervised learning literature survey," *Computer Science, University of Wisconsin-Madison*, vol. 2, no. 3, p. 4, 2006.

[4] M. Zhao, T. W. Chow, Z. Wu, Z. Zhang, and B. Li, "Learning from normalized local and global discriminative information for semi-supervised regression and dimensionality reduction," *Information Sciences*, vol. 324, pp. 286–309, Dec. 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0020025515004533

[5] S. Basu, I. Davidson, and K. Wagstaff, *Constrained clustering: Advances in algorithms, theory, and applications*. CRC Press, 2008.

[6] Z. Zhang, T. W. S. Chow, and M. Zhao, "Trace ratio optimization-based semi-supervised nonlinear dimensionality reduction for marginal manifold visualization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 5, pp. 1148–1161, May 2013.

[7] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the eleventh annual conference on Computational learning theory*. ACM, 1998, pp. 92–100.

[8] Z.-H. Zhou and M. Li, "Semi-supervised regression with co-training," in *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, ser. IJCAI'05. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2005, pp. 908–913. [Online]. Available: http://dl.acm.org/citation.cfm?id=1642293.1642439

[9] T. Joachims, "Transductive inference for text classification using support vector machines," in *ICML*, vol. 99, 1999, pp. 200–209.

[10] K. P. Bennett and A. Demiriz, "Semi-supervised support vector machines," in *Advances in Neural Information processing systems*, 1999, pp. 368–374.

[11] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using em," *Machine learning*, vol. 39, no. 2, pp. 103–134, 2000.

[12] D. Azriel, L. D. Brown, M. Sklar, R. Berk, A. Buja, and L. Zhao, "Semi-supervised linear regression," *arXiv preprint arXiv:1612.02391*, 2016.

[13] K. J. Ryan and M. V. Culp, "On semi-supervised linear regression in covariate shift problems." *Journal of Machine Learning Research*, vol. 16, pp. 3183–3217, 2015.

[14] M. Ji, T. Yang, B. Lin, R. Jin, and J. Han, "A Simple Algorithm for Semi-supervised Learning with Improved Generalization Error Bound," *arXiv:1206.6412 [cs, stat]*, Jun. 2012, arXiv: 1206.6412. [Online]. Available: http://arxiv.org/abs/1206.6412

[15] A. Moscovich, A. Jaffe, and B. Nadler, "Minimax-optimal semi-supervised regression on unknown manifolds," *arXiv preprint arXiv:1611.02221*, 2016.

[16] D. Cai, X. He, and J. Han, "Semi-supervised regression using spectral techniques," 07 2006.

[17] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *Journal of Machine Learning Research*, vol. 7, pp. 2399–2434, 2006.

[18] M. Gasse, A. Aussem, and H. Elghazel, "On the Optimality of Multi-Label Classification under Subset Zero-One Loss for Distributions Satisfying the Composition Property," in *International Conference on Machine Learning*, ser. Journal of Machine Learning Reasearch Proceedings, F. R. Bach and D. M. Blei, Eds., vol. 37, Lille, France, 2015, pp. 2531–2539. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01234346

[19] H. Borchani, G. Varando, C. Bielza, and P. Larrañaga, "A survey on multi-output regression," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 5, no. 5, pp. 216–233, 2015. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1002/widm.1157/full

[20] E. Spyromitros-Xioufis, G. Tsoumakas, W. Groves, and I. Vlahavas, "Multi-Target Regression via Input Space Expansion: Treating Targets as Inputs," *Machine Learning*, vol. 104, no. 1, pp. 55–98, Jul. 2016, arXiv: 1211.6581. [Online]. Available: http://arxiv.org/abs/1211.6581

[21] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2009, pp. 254–269.

[22] W. Liu, I. W. Tsang, and K.-R. Müller, "An easy-to-hard learning paradigm for multiple classes and multiple labels," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 3300–3337, 2017.

[23] K. Yu, V. Tresp, and A. Schwaighofer, "Learning Gaussian processes from multiple tasks," in *Proceedings of the 22nd international conference on Machine learning*. ACM, 2005, pp. 1012–1019.

[24] P. Gong, J. Ye, and C.-s. Zhang, "Multi-stage multi-task feature learning," in *Advances in neural information processing systems*, 2012, pp. 1988–1996.

[25] Y. Shi, J. Miao, Z. Wang, P. Zhang, and L. Niu, "Feature selection with $l_{2,\,1\text{-}2}$ regularization," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 29, no. 10, pp. 4967–4982, 2018. [Online]. Available: https://doi.org/10.1109/TNNLS.2017.2785403

[26] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[27] A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-task feature learning," in *Advances in neural information processing systems*, 2007, pp. 41–48.

[28] L. Jian, J. Li, K. Shu, and H. Liu, "Multi-Label Informed Feature Selection." in *IJCAI*, 2016, pp. 1627–1633.

[29] J. Zhou, J. Chen, and J. Ye, "Clustered multi-task learning via alternating structure optimization," in *Advances in neural information processing systems*, 2011, pp. 702–710.

[30] Y.-n. Chen and H.-t. Lin, "Feature-aware Label Space Dimension Reduction for Multi-label Classification," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1529–1537.

[31] M. Luo, L. Zhang, F. Nie, X. Chang, B. Qian, and Q. Zheng, "Adaptive semi-supervised learning with discriminative least squares regression," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 2017, pp. 2421–2427. [Online]. Available: https://doi.org/10.24963/ijcai.2017/337

[32] J. Zhou, J. Chen, and J. Ye, "Malsar: Multi-task learning via structural regularization," *Arizona State University*, 2011.

[33] G. Chen, Y. Song, F. Wang, and C. Zhang, "Semi-supervised multi-label learning by solving a sylvester equation," in *Proceedings of the 2008 SIAM International Conference on Data Mining*. SIAM, 2008, pp. 410–419.

[34] E. S. Xioufis, W. Groves, G. Tsoumakas, and I. P. Vlahavas, "Multi-label classification methods for multi-target regression," *CoRR*, vol. abs/1211.6581, 2012. [Online]. Available: http://arxiv.org/abs/1211.6581

[35] V. Kraus, S.-E. Benkabou, K. Benabdeslem, and F. Cherqui, "An improved laplacian semi-supervised regression," in *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)*. Volos, Greece: IEEE, Nov. 2018, pp. 564–570.

[36] H. Q. Minh and V. Sindhwani, "Vector-valued manifold regularization." in *ICML*. Citeseer, 2011, pp. 57–64.

[37] T. Evgeniou and M. Pontil, "Regularized multi–task learning," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2004, pp. 109–117.

[38] Y. NESTEROV, "Gradient methods for minimizing composite functions," *Université catholique de Louvain, Center for Operations Research and Econometrics (CORE), CORE Discussion Papers*, vol. 140, 01 2007.

[39] S. Vijayakumar and S. Schaal, "Locally weighted projection regression: An o (n) algorithm for incremental real time learning in high dimensional space," in *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*, vol. 1, 2000, pp. 288–293.

[40] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

[41] A. Jalali, S. Sanghavi, C. Ruan, and P. K. Ravikumar, "A dirty model for multi-task learning," in *Advances in Neural Information Processing Systems 23*, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, Eds. Curran Associates, Inc., 2010, pp. 964–972. [Online]. Available: http://papers.nips.cc/paper/4125-a-dirty-model-for-multi-task-learning.pdf

[42] S. Ji and J. Ye, "An accelerated gradient method for trace norm minimization," in *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009, pp. 457–464.

[43] J. Zhou, J. Chen, and J. Ye, "Clustered multi-task learning via alternating structure optimization," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2011, pp. 702–710. [Online]. Available: http://papers.nips.cc/paper/4292-clustered-multi-task-learning-via-alternating-structure-optimization.pdf

[44] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine learning research*, vol. 7, no. Jan, pp. 1–30, 2006.