Multi-task feature selection

Guillaume Obozinski gobo@stat.berkeley.edu
Ben Taskar taskar@cs.berkeley.edu
Michael Jordan jordan@cs.berkeley.edu

Technical Report
June 2006
Department of Statistics
University of California, Berkeley

Abstract

We address the problem of *joint feature selection* across a group of related classification or regression tasks. We propose a novel type of joint regularization of the model parameters in order to couple feature selection across tasks. Intuitively, we extend the ℓ_1 regularization for single-task estimation to the multi-task setting. By penalizing the sum of ℓ_2-norms of the blocks of coefficients associated with each feature across different tasks, we encourage multiple predictors to have similar parameter sparsity patterns. To fit parameters under this regularization, we propose a blockwise boosting scheme that follows the regularization path. The algorithm introduces and updates simultaneously the coefficients associated with one feature in all tasks. We show empirically that this approach outperforms independent ℓ_1-based feature selection on several datasets.

1 Introduction

We consider the setting of multi-task learning, where the goal is to estimate predictive models for several related tasks. For example, we might need to recognize speech of different speakers, or handwriting of different writers, learn to control a robot for grasping different objects or drive in different landscapes, etc. We assume that the tasks are sufficiently different that learning a specific model for each task results in improved performance, but similar enough that they share some common underlying representation that should make simultaneous learning beneficial. In particular, we focus on the scenario where the different tasks share a subset of relevant features to be selected from a large common space of features.

1.1 Feature selection for a group of tasks

Feature selection has been shown to improve generalization in situations in which many irrelevant features are present. In particular, the regularization by the ℓ_1-norm implemented by the LASSO procedure [12] has been shown to yield useful feature selection. Model fits obtained under the ℓ_1-norm penalty are typically sparse in the sense that only a few of the parameters are non-zero and the resulting models are often more accurate and more easily interpretable. Fu & Knight [7] characterize the asymptotic behavior of the solutions and their sparsity patterns and Donoho [5] shows how for some large linear systems of equations the ℓ_1 regularized solution achieves in a certain sense optimal sparsity. Recent papers [6, 11, 15] have established clearer correspondences between the regularization path of the LASSO and the path obtained with certain boosting schemes.

To learn models for several tasks, the ℓ_1 regularization can obviously be used individually for each task. However, in the multi-task setting we prefer a regularization scheme that encourages solutions with shared pattern of sparsity. For example, when learning personalized models for handwriting recognition for each writer, we expect that even though individual styles vary, there is a common subset of image features (pixels, strokes) shared across writers. If we consider the entire block of coefficients associated with a feature across tasks as a unit, we can encourage sparsity at the block level, leading to estimated parameters where several blocks of coefficients are set to zero.

2 Joint Regularization for Feature Selection

2.1 From single tasks to multiple tasks

Formally, we assume that there are L models to learn and our training set consists of the samples $\{(x_i^l, y_i^l) \in \mathcal{X} \times \mathcal{Y},\ i = 1 \cdots N_l,\ l = 1 \cdots L\}$ where l indexes tasks and i indexes the i.i.d. samples for each task. We assume that the input space $\mathcal{X}$ is $\Re^K$ and the output space $\mathcal{Y}$ is either $\{0, 1\}$ for binary classification tasks or $\Re$ for regression tasks.

Let $w^l \in \Re^K$ be the model parameter vector for each task, and let $J^l(w^l, x^l, y^l)$ be the loss function on example (x^l, y^l) for task each task l. Typical loss functions for linear classification models include log-likelihood, exponential and hinge losses. For regression, squared error and absolute error are commonly used. Learning a model independently through empirical risk minimization with ℓ_1 regularization would yield the optimization problem:

$$\min_{w^l} \frac{1}{N_l} \sum_{i=1}^{N_l} J^l(w^l, x_i^l, y_i^l) + \lambda \|w^l\|_1.$$

Solving each of these problems independently is equivalent[1] to solving the global

[1] Provided the regularization coefficient λ is the same across tasks.

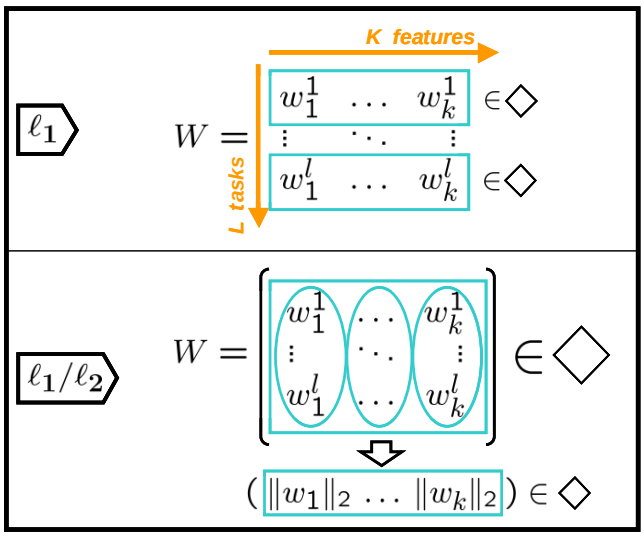

Figure 1: The ℓ_1/ℓ_2 vs the ℓ_1 regularization schemes .

problem obtained by summing the objectives:

$$\min_W \sum_{l=1}^{L} \frac{1}{N_l} \sum_{i=1}^{N_l} J^l(w^l, x_i^l, y_i^l) + \lambda \sum_{l=1}^{L} \|w^l\|_1 \tag{1}$$

where $W = (w_k^l)_{l,k}$ is the matrix with w^l in rows or equivalently with w_k in columns where w_k is the vector of coefficients associated with feature k across tasks. Solving this optimization problem would lead to individual sparsity patterns for each w^l.

Alternatively, in order to select features globally, we would like to encourage several w_k to be zero. We thus propose to solve the problem

希望这个feature对每个task都没用

$$\min_W \sum_{l=1}^{L} \frac{1}{N_l} \sum_{i=1}^{N_l} J^l(w^l, x_i^l, y_i^l) + \lambda \sum_{k=1}^{K} \|w_k\|_2 \tag{2}$$

i.e., to penalize the ℓ_1-norm of the vector of ℓ_2-norms of the feature-specific coefficient vectors (see Figure 1 for a pictorial representation of this regularization). Note that this ℓ_1/ℓ_2 regularization scheme reduces to the ℓ_1 regularization in the single-task case, and can thus be seen an extension of it where instead of summing the absolute values of coefficients associated to features we sum the Euclidean norms of coefficient blocks.

The ℓ_2-norm is just used here as a measure of magnitude and one could also use ℓ_p-norms for $1 \leq p \leq \infty$ and generalize to ℓ_1/ℓ_p-norms. The choice of p should depend on how much a priori feature sharing we assume between the tasks, from none ($p = 1$) to complete ($p = \infty$). Indeed, increasing p corresponds

to allowing better "group discounts" for sharing the same feature, from $p = 1$ where the cost grows linearly with the number of tasks that use a feature to $p = \infty$ where only the most demanding task matters.

2.2 Geometric interpretation

The shape of the unit ball of the ℓ_1/ℓ_2-norm is somewhat difficult to visualize. It clearly has sharp edges (even though not straight) and vertices, that, in a manner analogous to the ℓ_1 norm, tend to produce sparse solutions. One way to appreciate the effect of the ℓ_1/ℓ_2 norm is to consider two tasks with two features and to observe (see Figure 2.2 (Center)) the ball of the norm induced on w^2 when w^1 varies under the constraint that $\|w^1\|_1 = 1$ in an ℓ_1/ℓ_2 ball of size $2\sqrt{2}$ (which is the value of the ℓ_1/ℓ_2 norm if $w_i^1 = w_i^2 = 1$). If a feature k has a non-zero coefficient in w^1 then the induced norm on w^2 is smooth around $w_k^2 = 0$. Otherwise, it has sharp corners, which encourages w_k^2 to be set to zero.

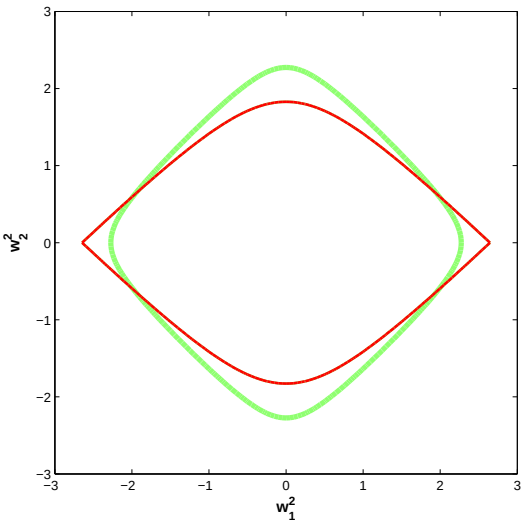

Figure 2: Norm ball induced on the coefficients (w_1^2, w_2^2) for task 2 as feature coefficients for task 1 vary: thin red contour for $(w_1^1, w_2^1) = (0, 1)$ and thick green contour for $(w_1^1, w_2^1) = (0.5, 0.5)$.

3 A Boosting Algorithm

To fit a regression or classification model under the ℓ_1/ℓ_2 regularization, we extend a recently-proposed boosting methodology due to Zhao and Yu [15]. The Zhao and Yu algorithm follows the regularization path (in the limit of

vanishing step size) of the empirical risk associated with any strictly convex smooth function J and penalized with any convex function T (typically a norm). Like forward stagewise boosting, their algorithm is based on cautious coordinate descent. They distinguish two types of steps:[2]

Backward or Correction step:
An ϵ step along the *non-zero* coordinate that most decreases $\Gamma_\lambda(w) = \lambda T(w) + J(w)$. After this step λ stays the same.

Forward or Prediction step:
An ϵ step along the coordinate that most decreases J. Taking a *forward step* corresponds to decreasing λ.

Their algorithm is as follows:

1. Take a forward step from 0 to $w^{(1)}$ and define $\lambda^1 = \frac{J(0)-J(w^{(1)})}{T(w^{(1)})-T(0)}$.
2. Take several backward steps until Γ_{λ^t} doesn't decrease significantly anymore
3. Take a forward step from $w^{(t)}$ to $w^{(t+1)}$. Update λ: $\lambda^{t+1} = \min\left(\lambda^t, \frac{J(w^{(t)}-J(w^{(t+1)}))}{T(w^{(t+1)})-T(w^{(t)})}\right)$ then go to 2 unless λ^{t+1} is very small in which case stop.

Backward steps are extremely fast since their only require to consider non-zero coordinates, whereas forward steps involve finding the best coordinate among a possibly large set of directions.

3.1 Block-coordinate Descent

In our case, J is the empirical loss $J(W) = \frac{1}{nL}\sum_{i,l} J^l(w^l, x_i^l, y_i^l)$ and $T(W) = \sum_k ||w_k||_2$. To respect the block structure, we extend the Zhao and Yu algorithm such that an ϵ step involves the simultaneous update of the block of coefficients associated with one feature. Accordingly, a candidate step for feature k moves in the steepest descent direction in the k^{th} subspace along:

- $-\nabla_{w_k}\Gamma_{\lambda^t}(W^{(t)}) = -\lambda^t \frac{{w_k}^{(t)}}{||{w_k}^{(t)}||_2} - \nabla_{w_k} J(W^{(t)})$ for a backward step
- $-\nabla_{w_k} J(W^{(t)})$ for a forward step.[3]

The feature which is selected for the next step is the one which, to first order, decreases the objective most i.e.,

- $\text{argmax}_k \, ||\nabla_{w_k}\Gamma_{\lambda^t}(W^{(t)})||_2$ for a backward step
- $\text{argmax}_k \, ||\nabla_{w_k} J(W^{(t)})||_2$ for a forward step.

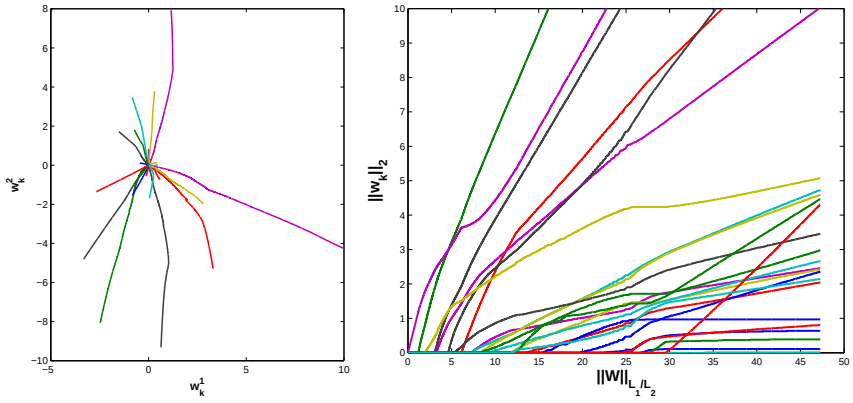

Figure 3: (Left) ℓ_1/ℓ_2 boosting path in the plane of the two tasks, (Right) The ℓ_2-norms of the coefficient vectors w_k associated with each feature as a function of the total ℓ_1/ℓ_2-norm of the solutions obtained on the boosting path

However taking an ϵ step in the direction of the gradient does not necessarily decrease the objective, since the step size may be too large. For backward steps, doing line searches instead of ϵ steps is faster anyway and we can replace the backwards steps by any procedure that minimizes Γ_{λ_t} with the current set of active (or non-zero) features held fixed. For the forward step, the line search should be bounded so as to make a step of at most ϵ to relax the regularization gradually.[4]

4 Extension to subspace selection

Our assumption has been that a set of related tasks share a common small set of features implies that the parameters of the corresponding predictors share similar sparsity patterns and as a consequence that all belong to the low-dimensional subspace spanned by the selected features. An alternative approach considers arbitrary low-dimensional subspaces which are not necessarily aligned with the original axes [1]. Multi-task learning in this setting means finding the common general-position subspace that is shared among tasks.

It turns out that we can also implement such an approach within our framework. Reposing on the Johnson-Lindenstrauss lemma [3], which says that, with very high probability, projections are nearly isometries (up to a rescaling), we

[2]The forward step is identical to the step in forward stagewise boosting.

[3]Note that the backward steps are taken along *non-zero* coordinates so that there are no issues of differentiating the ℓ_2-norm at 0.

[4]Reducing the size of the step obviously slows down the algorithm; on the other hand even with the simpler case of [15] for the L_1 norm, for a fixed size ϵ the boosting algorithm can be stuck far away from the unregularized solution.

can consider generating large sets of candidate features by taking random projections. In particular, if we assume the existence of a common subspace for the parameters of multiple predictors, it should be locally well approximated by random projections. Given randomly-chosen candidate projections, and given a multi-task learning problem, our algorithm should be able to uncover the projections which are most useful across tasks, thus uncovering the common subspace linking the tasks.

5 Experiments and applications

5.1 Writer specific OCR

5.1.1 Setting

We apply our method in the context of handwritten character recognition. Consider, for different writers, the task of learning to differentiate between pairs of letters. The simplest approach is to pool all the letters from all writers and learn global classifiers; this may be justifiable if we dispose of only a few examples of each letter per writer, but of enough different writers. Another naive method is to learn each classifier independently. We propose to compare these with our two proposed approaches involving either *joint feature selection* or *common subspace selection*.

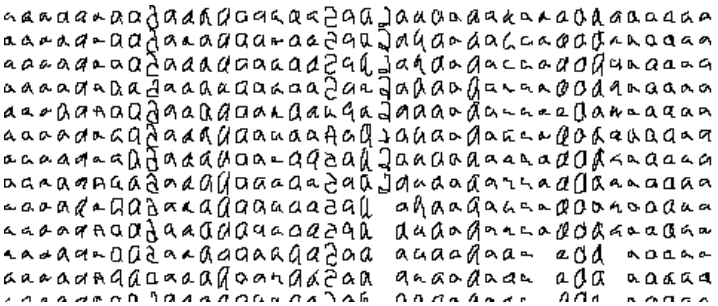

Figure 4: The letter a written by 40 different people.

5.1.2 Data

We used letters from a handwritten words dataset collected by Rob Kassel at the MIT Spoken Language Systems Group. This dataset contains writings from more than 180 different writers. For each writer, however, the number of examples of each letter is rather small: between 4 and 30 depending on the letter. The letters are originally represented as 8×16 binary pixel images.

5.1.3 Experimental setup

We built binary classifiers that discriminate between pairs of letters. Specifically we concentrated on the pairs of letters that are the most difficult to distinguish when written by hand. We compare four discriminative methods, all based on minimizing a log-loss, with different regularization schemes as follows:

1. **Pooled ℓ_1:** The writers are ignored, all the letters of both classes to be discriminated are pooled. We use the Blasso boosting scheme to follow the regularization path of the ℓ_1 logistic regression.

2. **Independent ℓ_1 regularization:** For each writer an independent classifier is learned using Blasso boosting.

3. **ℓ_1/ℓ_1-regularization:** The objective function is (1) with the log-loss i.e. the tasks are only tied by the regularization coefficient. The regularization path is obtained by a joint Blasso boosting scheme where the updates of features used in different tasks are interwoven.

4. **ℓ_1/ℓ_2-regularization:** The objective is (2) with the log-loss and therefore the feature selection processes are tied by the regularization.

These methods are also combined with random projections, used either as an efficient dimensionality reduction technique, or as a way to find common subspaces for the parameters.

5.1.4 Feature engineering

We compare two approaches:

- The features used are the raw pixels. These features are noisy features that are not particularly adapted to the encoding of written letters.

- The features used are based on a model of *strokes* (see 5.1.5). These features should make the classification problem easier.

5.1.5 Model of strokes and feature extraction

Our model of strokes is guided by the fact that people write letters as several strokes, which are smooth pieces of curve building the letter, (i.e. drawn each at slowly varying speed). To match the intuition we use an ad hoc second order Gaussian Markov model where the speed varies slowly to privilege low curvature. Following this model we take a random walk on pixels of the letter, constrained to move to a neighboring pixel in the letter at each time step. We follow walks of lengths 2, 4 and 6 and call these walks "strokes". To take into account the thickness of strokes we then add all the pixels of the letters that are neighbors of the stroke to it. The obtained stroke is finally smoothed by convolution with a small kernel. To construct a relevant set of strokes for the task of discriminating between two letters we extract strokes in the training set

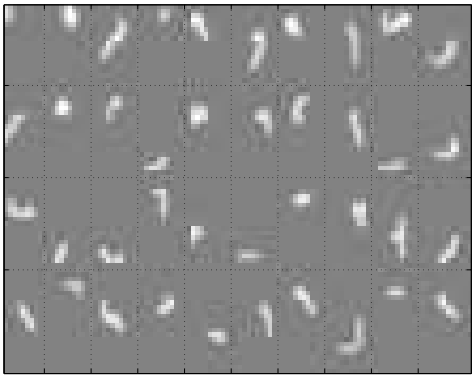

Figure 5: Stroke features extracted from the data

from letters of these two types and a few from other letter types as well. The total number of strokes we generated in each of our experiments is on the order of a thousand.

5.1.6 Results

We learned classifications of 9 pairs of letters for 40 different writers according to the 4 schemes presented in Section 5.1.3. We used the two types of features proposed (pixels and strokes) and conducted experiments with or without random projection as preprocessing. For $\{c, e\}$ and $\{g, s\}$ we generated two different sets of stroke features. For g, y we generated two sets of random projections. The error rates of the classifiers obtained are reported in table 1.

5.1.7 Analysis

In terms of performance, using better features obviously leads to better classifiers. For stroke features, independent ℓ_1 and ℓ_1/ℓ_2 regularizations perform comparably. Pooling doesn't perform very well because the inter-writer variance of the letters is large compared to the inter-class variance. For pixel features, independent ℓ_1 classification performs much worse than ℓ_1/ℓ_2 regularization. The ℓ_1/ℓ_2-regularized classifiers built on pixels don't perform as well as those built on strokes, but are still reasonably close.

The limited information contained in the pixel representation appears to be much harder to extract in a data-limited setting than extracting the information contained in the strokes; the ℓ_1/ℓ_2 seems to allow the appropriate level of exchange of information between tasks to permit the discovery of the relevant features. Moreover, it may be that in the case of the stroke features, much of the inter-task knowledge is already extracted by the stroke construction process.

Table 1: **Average 0-1 loss on the test set for the described combinations** (In each cell the first row contains results for feature selection, the second row uses random projections to obtain a common subspace, Bold: best of ℓ_1/ℓ_2, ℓ_1/ℓ_1, idpt. ℓ_1 or pooled ℓ_1, Boxed : best of cell)

	strokes : error(%)				pixels: error (%)			
Task	ℓ_1/ℓ_2	ℓ_1/ℓ_1	sp.ℓ_1	pool	ℓ_1/ℓ_2	ℓ_1/ℓ_1	sp.ℓ_1	pool
c/e	**2.5**	3.0	3.3	3.0	**4.0**	8.5	9.0	4.5
	2.0	3.5	3.3	2.5	**3.5**	7.8	10.3	4.5
g/y	8.4	11.3	**8.1**	17.8	**11.4**	16.1	17.2	18.6
	10.3	10.3	**9.3**	16.9	11.6	**9.7**	10.9	21.4
g/s	3.3	3.8	**3.0**	10.7	**4.4**	10.0	10.3	6.9
	3.8	4.0	**2.5**	12.0	4.7	6.7	5.0	6.4
m/n	4.4	4.4	**3.6**	4.7	**2.5**	6.3	6.9	4.1
	4.1	5.8	**3.6**	5.3	**1.9**	2.8	4.1	
a/g	**1.4**	2.8	2.2	2.8	**1.3**	3.6	4.1	3.6
	0.8	1.6	1.3	2.5	**0.8**	1.7	1.4	3.9
i/j	**8.9**	9.5	9.5	11.5	12.0	14.0	14.0	**11.3**
	9.2	9.8	11.1	11.3	**10.3**	12.7	13.5	11.5
a/o	**2.0**	2.9	2.3	3.8	**2.8**	4.8	5.2	4.2
	2.7	2.7	**1.9**	4.3	**2.1**	3.1	3.5	4.2
f/t	**4.0**	5.0	6.0	8.1	**5.0**	6.7	6.1	8.2
	5.8	**4.1**	5.5	7.5	**6.4**	11.1	9.6	7.1
h/n	**0.9**	1.6	1.9	3.4	**3.2**	14.3	18.6	5.0
	0.9	0.6	**0.3**	3.7	**1.8**	3.6	5.0	5.0

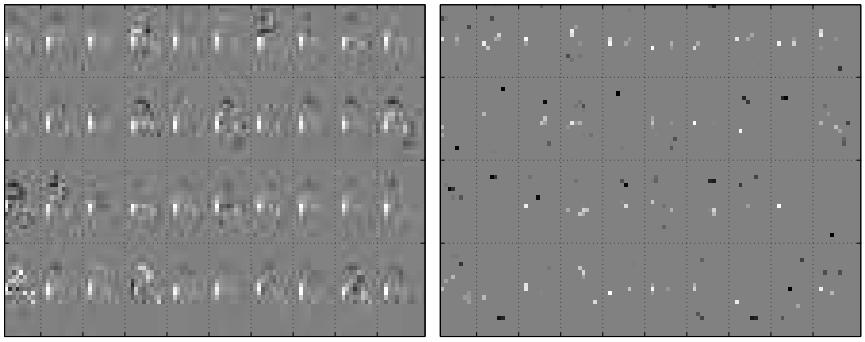

Figure 6: Mask learnt for the classification of a vs g with ℓ_1/ℓ_2 regularization (left) and with independent ℓ_1 regularization.

In terms of size of the resulting predictors, the ℓ_1/ℓ_2-regularization uses a minimal number of features, since it maximizes the overlap of the feature sets. This is particularly noticeable for the stroke feature representation where less than 50 features are typically retained versus 3-5 times as many for the other schemes except pooling. Conversely, the other schemes are forced by the regularization to use only very few features

5.2 Multi-class classification

There are two important learning problems that provide a good fit to our framework but which are not multi-task per se. In particular, in both multivariate (linear) regression and multi-label classification, a notion of "feature transfer" is relevant and the methods presented in this paper are potentially beneficial. In Section 2.1, the loss that we have introduced is a sum of task-specific losses; however if we generalize to the case were the contribution of each task is not necessarily linear, we can treat multi-class logistic regression in our framework too. In the remainder of this section we investigate the use of our regularization on two multi-class classification problems.

5.2.1 "Transfer regime"

The Dermatology dataset [4] of the UCI repository [10] involves classifying a disease in six possible diagnoses based on 33 symptoms with four values each; we code them as 99 binary features. Varying the size of the training set we observe different regimes. Our experiments show that for very small sample sizes as well as in the asymptotic regime, the two regularizations perform equally well, but for moderate sample sizes i.e. around 50 datapoints the ℓ_1/ℓ_2 regularization provides a significant improvement of up to 50% in average over the independent ℓ_1 regularization.

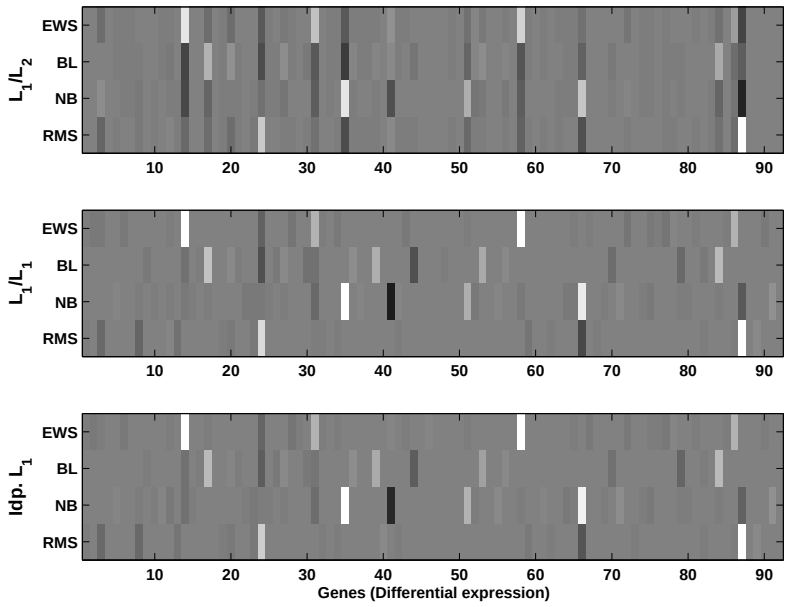

Figure 7: Matrix of feature coefficients obtained from three regularization methods. ℓ_1/ℓ_2, ℓ_1/ℓ_1 and idp. ℓ_1 respectively use 57, 81 and 85 contributing genes to classify four cancer types: EWS, BL, NB, RMS. ℓ_1/ℓ_2 has an interesting "mikado" pattern indicating that a given feature has important opposite effects in the classification of two classes that it discriminates well.

5.2.2 Compact gene signature

The diagnosis of complex diseases such as cancers can today be assisted by genomic information thanks to the development of microarrays; more precisely the latter allow us to identify genes that are differentially expressed in different cell lineages or at different stages of a cancer. This is interesting because the relationship between gene expression patterns and the illness is more direct than general symptoms, but it is also difficult to assess because of the large number of genes and the high levels of noise present in the data. We used the ℓ_1/ℓ_2 regularization scheme to learn to differentiate four types of skin cancers [9] from differential gene expression (DGE). The $\ell_1/\ell 2$, $\ell_1/\ell 1$ and independent ℓ_1 penalized regressions all performed as well as other methods proposed in [9] and [14]; however, the ℓ_1/ℓ_2 regression proposes a smaller set of genes than the others, while still matching the results in [14]. A small gene signature is obviously of biological interest. A striking feature of the sparsity pattern obtained from ℓ_1/ℓ_2 is that several genes are eliminated because if the expression of a gene is indicative of a cancer type, then that feature is encouraged to be also discriminative for the other cancers.

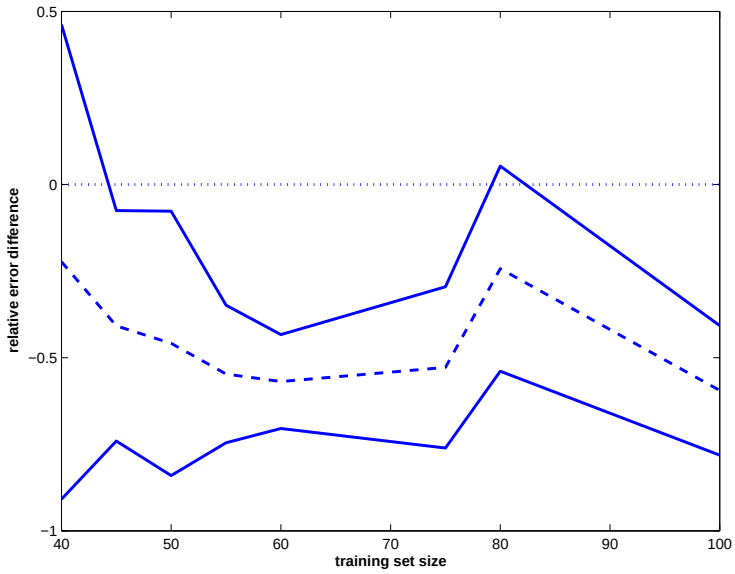

Figure 8: Average relative difference in error of the ℓ_1/ℓ_2 scheme with respect to the error of the independent ℓ_1 regularization scheme on the UCI dermatology dataset as a function of increasing training set size. The error bars are 2 standard deviations away from the mean and show that at each position in the range 45-75 and above 85 the improvement of the ℓ_1/ℓ_2 regularization (up to 50%) is significant at the 95% level.

6 Related Work

There is a relatively small existing literature on selecting features for multiple related tasks. Jebara [8] proposed a general formulation of feature selection which naturally extends to the multi-task case in the framework of Maximum Entropy Discrimination. In the context of multi-class classification, Torralba et al. [13] proposed a joint feature boosting algorithm learn all the one-vs-all binary classifiers where the weak learners of the classical boosting scheme are step functions applied to individual features. The learners are selected greedily according to whether they separate well some bipartition of the set of classes and among them reduce most the average empirical risk on all the classifiers of interest. Their work has the advantage of allowing for non-linear classification, but it also has some shortcomings: the choice of the feature coefficients are tied across tasks, and the restriction to weak classifiers of bipartitions is restrictive. Finally, the ℓ_1/ℓ_2-norm has also been considered in the primal formulation of the Support Kernel Machine [2].

7 Conclusion

We have presented a new regularization scheme for multi-task feature selection where different tasks are learned simultaneously and make an interdependent set of choices of relevant features. We have proposed a boosting scheme appropriate for fitting models under this regularization.

We note that even though we have focused a symmetric learning situation, the same approach can be applied to the asymmetric situation in which first one task is learned and then a second task is learned, by simply "conditioning" on the parameters learned for the earlier task. We also proposed a variant based on random projection, to learn a common parameter subspace.

We showed empirically in three application domains that the proposed regularization methodology can improve overall classification results. Our interpretation of these results is that the method is particularly useful in cases in which features are complex and difficult to induce from limited data, or in cases in which data are available in relatively limited quantity per task. Another advantage of the regularization is that it provide more compact and hence more interpretable models.

References

[1] Rie K. Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *JMLR*, 6:1817–1853, 2005.

[2] Francis R. Bach, Gert R. G. Lanckriet, and Michael I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. Morgan Kaufmann, 2004.

[3] Sanjoy Dasgupta and Anupam Gupta. An elementary proof of the Johnson-Lindenstrauss lemma. Technical Report TR-99-06, International Computer Science Institute, Berkeley, CA, 1999.

[4] G. Demiroz, H. A. Govenir, and N. Ilter. Learning differential diagnosis of eryhemato-squamous diseases using voting feature intervals. *Artificial Intelligence in Medicine*, pages 147–165, 1998.

[5] David Donoho. For most large underdetermined systems of linear equations the minimal ℓ_1-norm solution is also the sparsest solution. Technical report, Statistics Dpt, Stanford University, 2004.

[6] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. LARS. *Ann. Statistics*, 32(2):407–499, 2004.

[7] W. Fu and K. Knight. Asymptotics for lasso-type estimators. *Ann. Statistics*, 28:1356–1378, 2000.

[8] T. Jebara. Multi-task feature and kernel selection for svms. *ICML*, 2004.

[9] Javed Khan, Jun S. Wei, Markus Ringnr, Lao H. Saal, Marc Ladanyi, Frank Westermann, Frank Berthold, Manfred Schwab, Cristina R. Antonescu, Carsten Peterson, and Paul S. Meltzer. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine*, 7:673–679, 2001.

[10] D.J. Newman, S. Hettich, C.L. Blake, and C.J. Merz. UCI repository of machine learning databases, 1998.

[11] Saharon Rosset. *Topics in Regularization and Boosting*. PhD thesis, Statistics Department,Stanford University, 2003.

[12] Robert Tibshirani. Regression shrinkage and selection via the LASSO. *J. Royal. Stat. Soc. B*, 58(1):267–288, 1996.

[13] A. Torralba, K. Murphy, and W. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. *CVPR*, pages 762–768, 2004.

[14] Baolin Wu. Differential gene expression detection and sample classification using penalized linear regression models. *Bioinformatics*, 22(5):472–476, 2005.

[15] Peng Zhao and Bin Yu. Boosted lasso. Technical report, Statistics Department, UC Berkeley, 2004.