Learning linear classifiers with ternary weights from Metagenomic Data

Yann Chevaleyre[1], Frederic Koriche[2], Jean-Daniel Zucker[3,4]

[1] LIPN, Université Paris 13, XX ,
93000, Villetaneuse, France
chevaleyre@lipn.univ-paris13.fr

[2] LIRMM, Université Montpellier II. 161 rue Ada, 34392 Montpellier Cedex 5, France
frederic.koriche@lirmm.fr

[3] INSERM U872 Equipe 7 CRC, Université Pierre et Marie Curie(UPMC),15 rue des Ecole de Medecine, 75005 Paris, France
[4] IRD, UMI 209, UMMISCO, IRD France Nord, F-93143, Bondy, France
jean-daniel.zucker@ird.fr

Abstract : Motivated by recent researches in metagenomic classification tasks, this paper investigates the problem of finding interpretable concepts from training data in which the number of features is larger than the number of samples. In this setting, the classification problem is modeled as a combinatorial optimization problem, in which the aim of the learner is to find a $\{-1, 0, +1\}$-weighted linear threshold function that minimizes the hinge loss induced from the training data. Two methods are analyzed and experimented: *convex optimization* that applies randomized rounding to the optimum in the convex hull of the concept class, and *supermodular minimization* that performs a local search in the concept class, with a guarantee on the approximate solution for the subclass of $\{0, 1\}$-weighted threshold functions.

Mots-clés : High-dimensional data, linear boolean concepts, combinatorial optimization, convex relaxation, submodular optimization, randomized rounding

1. Introduction

1.1. Motivation

Classification tasks in which the number of features n is much larger than the number of samples m has gained interest in the past decade in Machine Learning (ML). Such classification tasks, often written $n \gg m$ (Hastie *et al.*, 2005), are in sharp contrast with the Data Mining paradigm in which m may be several orders of magnitude larger than n. Applications of classification tasks from high dimensional data have grown significantly in *genomics post-genomics* and more generally in computational biology. From theoretical point of view, the major issues arising from this setting are high variance and over-fitting. For these reasons, the analysis of $n \gg m$ tasks requires new procedures which depart from the usual $m > n$ setting.

Gene Expression data that are nowadays well known by many ML practicians are typically constituted of hundreds of samples ($m \simeq 100$) and ten thousands of features ($n \simeq 10000$). A density $^m/_n$ of 1% is very typical and a large literature has explored the pitfalls related to directly applying standard approaches in such cases (Hanczar *et al.*, 2007). In the past five years, thanks to the development of metagenomic approaches, the capacity to extensively explore microbiota composition and its variations with environmental challenges has considerably changed (Riesenfeld *et al.*, 2004). In fact, our view of the microbial world and its impact on human health is changing radically with the ability to sequence uncultured or unculturable microbes sampled directly from their habitats, as made possible by faster and cheaper emerging NGS (Next Generation Sequencing technologies). It represents a paradigmatic shift in the analysis of habitat biodiversity (human, soil or ocean microbiome). Such recent developments provide researchers with metagenomic data as counts for millions of different batteries. It corresponds to a decrease by two orders of magnitude compared with traditional microarray data of the density mentioned above which becomes as low as $0,01\%$. However, in such settings, the issues of over-fitting and *noise-discovery* (Ioannidis, 2005) has never been so high. In a nutshell, one of the key problem in metagenomic data is to find accurate, stable and interpretable models of molecular signatures.

1.2. Application context

Next-generation sequencing (NGS) technologies has already allowed the publication of a first human gut microbial gene catalogue based on the analysis of 124 Europeans gut microbiota (Qin *et al.*, 2010). In the context of the FP7 EU program MetaHIT (Metagenomics of the human Intestinal Tract) lead by Prof. Dusko Ehrlich (Ehrlich & Consortium, 2010; Arumugam *et al.*, 2011) 111 lean individuals (BMI<25 kg/m2) and 154 obese individuals (BMI>30 kg/m2) were included in the studied cohort. Common clinical and biological phenotypes have been collected as well as an analysis of their gut microbiota where counts for several millions bacterial genes where reported thanks to NGS technology.

In this paper, we present the models and algorithms we designed with these goals in mind. However, the dataset we used to run our experiments only contains a small fraction of all features we expect to have in the near future: This dataset has 38 individuals and 69 features. Each feature represents the activity of a bacterial gene and has been carefully selected by biologists.

The biologists with whom we have worked were eager to have interpretable models as their ultimate objective is to understand the etiology of various morbid phenotypes.

1.3. The Model

In this paper, the model we have decided to explore consist in $\{-1, 0, +1\}$-weighted linear threshold functions, which are conceptually simpler than the standard real-weighted linear threshold functions, also known as "perceptrons". Our discrete model has a simple and direct interpretation as a presence of inhibiting and activating genes. Furthermore, this simplicity decreases the structural risk of over-fitting. In this setting, the problem $n > m$ is viewed as a combinatorial optimization problem. Specifically, the task is to find a discrete n-dimensional linear model that minimizes the cumulated loss over a training set of m examples.

Despite the crucial importance of finding simple and interpretable linear models, there has been relatively few investigations in the study of integer-weighted linear threshold functions. A notable exception is the work by Golea & Marchand (1993a,b) who investigate the learnability issue of perceptrons with binary weights. In particular, they demonstrate that the problem of finding a $\{0, 1\}$-weighted linear threshold function that minimizes the cumulative zero-one loss (i.e. the number of classification errors) in a training set is NP-hard, even in the "realizable" case where the data can be separated by a function in this concept class.

In order to circumvent this computational barrier, the error measure examined in this study is the *hinge loss*, a well-known surrogate of the zero-one loss endowed with remarkable geometric properties. Namely, the hinge loss is *convex* and, for certain subclasses of discrete linear concepts, *supermodular*. Based on these properties, we propose two methods for minimizing the cumulative hinge loss of a training set. The first method operates in the convex relaxation of the concept class and uses randomized rounding, while the second method directly search an approximation of the optimum in the concept class by exploiting the supermodularity of the loss function.

2. Problem Setting

A *ternary-weighted linear threshold concept* is a pair $(\mathbf{c}, r)$ where $\mathbf{c}$ is a vector in $\{-1, 0, +1\}^n$ and r is a nonnegative real capturing the threshold or offset. In gene expression data, $c_i = 0$ indicates that the ith gene is inexpressive, $c_i = +1$ indicates that the ith gene is expressive with activating role, and $c_i = -1$ indicates that the ith gene is expressive with inhibiting role. Given an instance $\mathbf{x}$ in $\mathbb{R}^n$, the label assigned by $(\mathbf{c}, r)$ to $\mathbf{x}_t$ is defined by $\operatorname{sgn}(\langle \mathbf{c}, \mathbf{x} \rangle - r)$.

Two natural classes of concepts arise from this setting. The first class is the family $\mathcal{C}_r[0, 1]$ of r-threshold boolean functions. When the threshold r is clear from the context, each concept in this class is simply represented as a boolean vector $\mathbf{c}$ in $\{0, 1\}^n$. The second class is the family $\mathcal{C}_0[-1, 0, +1]$ of zero-threshold ternary functions. Here, a concept is represented as a vector $\mathbf{c} \in \{-1, 0, +1\}$. A concept $\mathbf{c}$ is *k-sparse* if $\|\mathbf{c}\|_1 \leq k$, that is, the number of non-zero weights in $\mathbf{c}$ is less than or equal to k.

An *example* is a pair $(\mathbf{x}, y)$ where $\mathbf{x}$ is an instance in $\mathbb{R}^n_+$ and y is a label in $\{-1, +1\}$. A *training set* is a collection $\{(\mathbf{x}_t, y_t)\}_{t=1}^m$ of examples.

A *loss function* is a map $\ell : \{-1, 0, +1\}^n \times \mathbb{R}^n \times \{-1, +1\}$ such that $\ell(\mathbf{c}; \mathbf{x}, y)$ measures the discrepancy between the predicted value $\langle \mathbf{c}, \mathbf{x} \rangle$ and the true label y. For a concept $\mathbf{c}$ and a training set $S = \{(\mathbf{x}_t, y_t)\}_{t=1}^m$, the *cumulative loss* of $\mathbf{c}$ with respect to S is given by $L(\mathbf{c}; S) = \sum_{t=1}^m \ell(\mathbf{c}; \mathbf{x}_t, y_t)$. Based on this performance measure, the combinatorial optimization problem investigated in this study is described as follows.

Loss Minimization over $\{-1, 0, +1\}^n$

Given

- a target concept class $\mathcal{C} \subseteq \{-1, 0, +1\}^n$,
- a training set $S = \{(\mathbf{x}_t, y_t)\}_{t=1}^m$,
- a loss function ℓ,

Find a concept $\mathbf{c} \in \mathcal{C}$ that minimizes $L(\mathbf{c}, S)$.

Recall that the *zero-one loss* is the loss function defined by $\ell_{01}(\mathbf{c}; \mathbf{x}, y) = 1$ if $\operatorname{sgn}(\langle \mathbf{c}, \mathbf{x} \rangle - r) \neq y$ and $\ell_{01}(\mathbf{c}; \mathbf{x}, y) = 0$. For this loss function the minimization problem over binary-weighted or ternary-weighted linear threshold concepts is known to be NP-hard, even in the ideal case where the training set is separable by the concept class (Golea & Marchand, 1993a). For this reason, we shall concentrate on the *hinge loss* a well-known surrogate of the zero-one loss defined as follows:

$$\ell_\gamma(\mathbf{c}; \mathbf{x}, y) = \frac{1}{\gamma} \max[0, \gamma - y(\langle \mathbf{w}, \mathbf{x} \rangle - r)] \text{ where } \gamma \in \mathbb{R}_+$$

Intuitively, the hinge loss penalizes a concept for any margin less than γ. By a direct reduction from the Binary Integer Programming problem, the minimization problem with the hinge loss is still NP-hard, even when the data is separable. However, the crucial difference with the zero-one loss lies in the fact that the hinge loss is convex and, for some concept classes, also supermodular. Such useful properties will be exploited in the remaining sections.

Reduction to Binary Weights. Instead of coping with ternary weights, we can use a simple trick to reduce our minimization problem to a binary minimization problem. The idea is to duplicate attributes in the following way: to each instance $\mathbf{x} \in \mathbb{R}^n$ we associate an instance $\mathbf{x}' \in \mathbb{R}^d$ where $d = 2n$ and $\mathbf{x}' = (x_1, -x_1, x_2, -x_2, \cdots, x_n, -x_n)$. Given a binary-valued concept $\mathbf{c}' \in \{0,1\}^d$, we can recover the ternary-valued concept $\mathbf{c} \in \{-1, 0, +1\}^n$ by setting $c_i = c'_{2i-1} - c'_{2i}$ for each $i \in [n]$. Based on this transformation, it is easy to see that $\ell(\mathbf{c}'; \mathbf{x}', y) = \ell(\mathbf{c}; \mathbf{x}, y)$, and hence, $L(\mathbf{c}'; \{(\mathbf{x}'_t, y_t)\}) = L(\mathbf{c}; \{(\mathbf{x}_t, y_t)\})$. Thus, if $\mathbf{c}'$ minimizes the cumulative loss on the training set $S' = \{(\mathbf{x}'_t, y_t)\}$, then $\mathbf{c}$ minimizes the cumulative loss on $S = \{(\mathbf{x}_t, y_t)\}$. If, in addition, $\mathbf{c}'$ is k-sparse, then $\mathbf{c}$ is k-sparse.

In the following section, we will present algorithms able to learn concepts from the class $\mathcal{C}_0[0,1]$. In fact, with the help of this reduction, the algorithms of the next section can also learn concepts from $\mathcal{C}_0[-1,0,+1]$.

3. Convex Formulation

In this section, we shall tackle the problem of learning concepts in $\mathcal{C}_0[0,1]$ by exploiting the property that the hinge loss is *convex*. The overall idea is to first relax the combinatorial optimization problem into a convex optimization problem, which will give us a vector of real valued weights. Then, to obtain weights in $\{0,1\}^d$, we shall apply to this real valued vector a randomized rounding procedure. We will begin by tackling the non sparse learning problem.

Figure 1: **Convex Rounding**

```
Input: A convex loss function $L$ and a training set $S$
(1) solve $\hat{\mathbf{w}} = \mathrm{argmin}\{L_\gamma(\mathbf{w}; S) : \mathbf{w} \in \mathbb{R}_+^d, \|\mathbf{w}\|_\infty \leq 1\}$
(2) Build the concept $\mathbf{c} \in \{0,1\}^d$ as follows: for each $i \in [d]$, $c_i$ is set to 1 with probability $\hat{\mathbf{w}}_i$
(3) Return $\mathbf{c}$
```

3.1. Non sparse algorithm

For a class of concept $\mathcal{C} \subseteq \{0,1\}^d$, we denote by $\mathrm{conv}(\mathcal{C})$ the convex hull of $\mathcal{C}$. In particular, if $\mathcal{C} = \{0,1\}^d$, then $\mathrm{conv}(\mathcal{C}) = [0,1]^d$. To convexify our learning problem, we shall replace the discrete constraint $\mathbf{c} \in \{0,1\}^d$ by a continuous constraint $\hat{\mathbf{w}} \in [0,1]^d$, and since the hinge loss ℓ_γ is convex, then for any training set S of size m the task of finding a vector $\hat{\mathbf{w}} \in [0,1]^d$ that minimizes $L_\gamma(\mathbf{w}; S)$ can be solved in time polynomial in d and m. In order to get $\{0,1\}$-valued concept $\mathbf{c}$, we need a final step called the *randomized rounding procedure*. Since $\hat{\mathbf{w}}$ is a point in the convex hull of $\{0,1\}^d$, we draw the random concept $\mathbf{c}$ in $\{0,1\}^d$ from the following distribution: each value c_i is chosen independently to be 1 with probability $\hat{w}_i$. Therefor, we know that $\hat{\mathbf{w}} = \mathbb{E}[\mathbf{c}]$ where $\mathbf{c}$ is the random concept.

The resulting algorithm called *convex rounding* is specified in Figure 1. In our implementation of this algorithm, we used a linear programming software (CVXOPT) to solve the optimization problem. Note that this optimization problem can easily be solved online with projection gradient techniques for example.

We note in passing that our optimization problem in the polytope $[0,1]^d$ can be viewed as an optimization problem over $\mathbb{R}_+^d$ under the l_∞-norm constraint. Also note that this algorithm has not been designed to generate k-sparse solutions.

When does the convex relaxation directly yield binary weights ? At the end of the convex optimization step, the learning algorithm generates a weight vector $\hat{\mathbf{w}}$. The accuracy of our algorithm depends heavily on this vector. If it is the case that most weights of $\hat{\mathbf{w}}$ are already in $\{0,1\}$, then these weights will remain unchanged by the randomized rounding procedure, and $\mathbf{c}$ will be quite close to $\hat{\mathbf{w}}$. Thus, the randomized rounding will not degrade the accuracy too

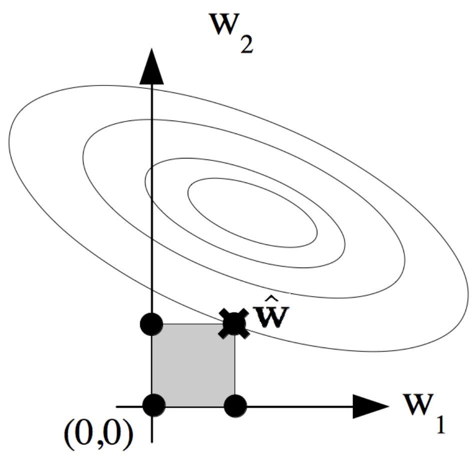

Figure 2: The solution to the convex relaxation coincindes with the solution to the original problem

much. Figure 2 illustrates this by representing a case where $\hat{\mathbf{w}}$ and $\mathbf{c}$ coincide. On his figure, the objective function is represented by ellipses. The four dots at the corner of the squares are the vectors $\{0, 1\}^2$. The vector $\hat{\mathbf{w}}$ positioned on the cross is the optimal solution both to the relaxed problem and to the discrete problem.

Increasing the margin parameter γ also increases the likelihood that weights become binary. In a similar way, this phenomenon appears in the Lasso feature selection procedure, where the weight vector are more likely to fall on a vertex of the ℓ_1 ball as the margin increases. To illustrate this, consider figure 3, where the number of non-zero weights is plotted as a function of γ. For sufficiently big γ, note that all weights are binary.

Also, consider the regularization path on figure 4. In this figure, the value of the first 20 weights have been plotted as functions of γ. For low values of the margin (left of the graph), most weights oscillate between 0 and 1, and for larger margins, nearly all weights become binary: their value is either 0 or 1.

In conclusion, depending on the margin, we may or may not need a randomized rounding pass. Also, when the margin is low, weights of $\hat{\mathbf{w}}$ are not binary and the randomized rounding procedure might by harmful.

In the experimental section, figure 7 on which we can see the empirical error rate before and after rounding confirms this fact: when the margin is low, the randomized rounding adds a huge amount of error.

To strengthen our analysis, the following proposition bounds the differ-

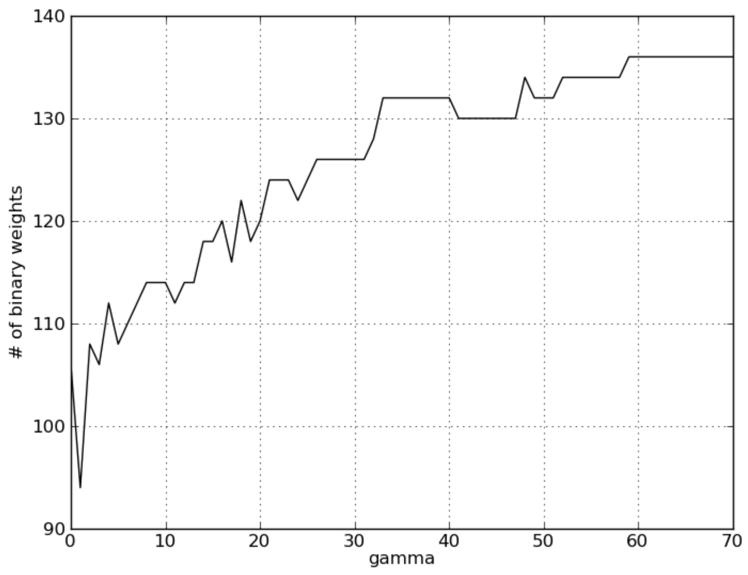

Figure 3: Number of weights whose value is exactly 0 or 1. As the margin γ grows, nealy all the 138 weights become binary.

ence between the loss of randomized rounded vector $\mathbf{c}$ and the optimal binary weight vector. As expected, this gap between the randomized rounding solution and the optimal one is proportional to $\frac{1}{\gamma}$. Interestingly, the *average* loss (not the cumulative loss) does not depend either on d nor on m. A crucial parameter is thus the ℓ_1 norm of the examples.

Proposition 1 *For any training set $S = \{(x_t, y_t)\}_{t=1}^m$, let $X_1 = \max_t \|\mathbf{x}_t\|_1$, and let $\mathbf{c}^* \in \{0,1\}^d$ be an optimal solution of the combinatorial loss minimization problem over S using the hinge loss. If $\mathbf{c}$ is the solution returned by the algorithm in Figure 1, then the following holds:*

$$\mathbb{E}[L(\mathbf{c}, S)] \leq L(\mathbf{c}^*, S) + m\frac{X_1}{2\gamma}$$

Proof (The proof is inspired by lemma A.1 and A.2 in Shalev-Shwartz *et al.*

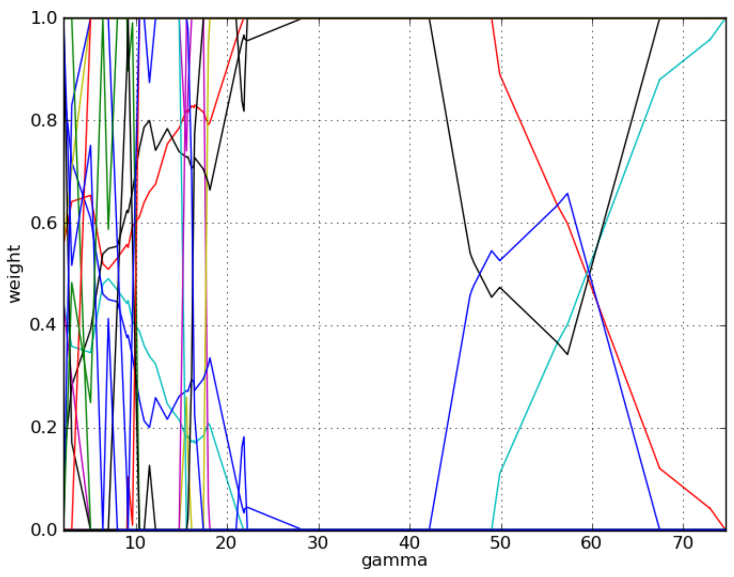

Figure 4: Regularization path showing the variation of the 20 first weights

(2010)))

$$\begin{aligned}
\mathbb{E}\left[L(\mathbf{c};S)\right]-L(\mathbf{c}^{*};S) &\leq \mathbb{E}\left[L(\mathbf{c};S)-L(\hat{\mathbf{w}};S)\right] \\
&\leq \sum_{t=1}^{m}\mathbb{E}\left[\ell(\mathbf{c};\mathbf{x}_t,y_t)-\ell(\hat{\mathbf{w}};\mathbf{x}_t,y_t)\right] && (1)\\
&\leq \frac{1}{\gamma}\sum_{t=1}^{m}\mathbb{E}\left[\left|\langle\mathbf{c},\mathbf{x}_t\rangle-\langle\hat{\mathbf{w}},\mathbf{x}_t\rangle\right|\right] && (2)\\
&\leq \frac{1}{\gamma}\sum_{t=1}^{m}\sqrt{\mathbb{E}\left[\left(\langle\mathbf{c},\mathbf{x}_t\rangle-\langle\hat{\mathbf{w}},\mathbf{x}_t\rangle\right)^2\right]} && (3)\\
&= \frac{1}{\gamma}\sum_{t=1}^{m}\sqrt{\mathbb{E}\left[\left(\langle\mathbf{c},\mathbf{x}_t\rangle-\mathbb{E}\langle\mathbf{c},\mathbf{x}_t\rangle\right)^2\right]} \\
&\leq \frac{1}{\gamma}\sum_{t=1}^{m}\sqrt{Var\left(\langle\mathbf{c},\mathbf{x}_t\rangle\right)} && (4)\\
&= \frac{1}{\gamma}\sum_{t=1}^{m}\sqrt{\sum_{i=1}^{d}\mathbf{x}_{t,i}^2 Var(\mathbf{c}_i)} \\
&\leq \frac{1}{\gamma}\sum_{t=1}^{m}\sqrt{\sum_{i=1}^{d}\frac{\mathbf{x}_{t,i}^2}{4}} && (5)\\
&\leq \frac{1}{2\gamma}\sum_{t=1}^{m}\sqrt{\left(\sum_{i=1}^{d}|\mathbf{x}_{t,i}|\right)^2} && (6)\\
&\leq \frac{mX_1}{2\gamma} && (7)
\end{aligned}$$

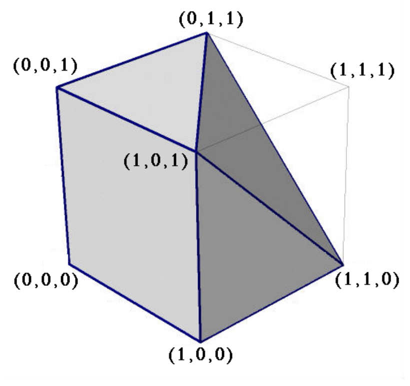

Figure 5: Intersection of the l_1 ball of radius 2, of the l_∞ ball of radius 1 for non negative coordinates

Inequation 2 comes from the fact that the hinge loss is $\frac{1}{\gamma}$-Lipschitz. Inequation 3 follows from Jensen's inequality. In inequation 4, we recognize the formula of the variance (note Var). Because $Var(aX) = a^2 Var(X)$ and because the variance of a bernouilli random variable is bounded by $\frac{1}{4}$, inequation 5 follows.

3.2. Sparsity

The *Convex Rounding* algorithm produces binary weights, but does not encourage sparsity. To generate k-sparse concepts, we have to find the convex hull of the set of all constraints including the k-sparsity. The class of concepts we wish to learn is $\{\mathbf{c} \in \{0,1\}^d : \|\mathbf{c}\|_1 \leq k\}$. The convex hull of this set is the intersection of $\mathbb{R}_+^d$, the l_∞ ball of radius 1, and the l_1 ball of radius k. Figure 5 represents this convex hull for $d = 3$ and $k = 2$. On this figure, the vertices are all points with binary coordinates except point $(1,1,1)$, which has more than k ones.

We will call *Sparse Convex Rounding* the convex rounding algorihm in which the sparsity constraints have been added. Note that the optimization problem (step 1 of the algorithm) can still be formulated as a linear optimization problem.

This new algorithm yields two nice guarantees: First, the bound of proposition 1 still holds, and the proof does not require any modification. Also, the only guarantee we get w.r.t. sparsity is the following:

Proposition 2 *Let $\mathbf{c}$ be the output of the Sparse Convex Rouding algorithm ran with parameter k. Then, $\mathbb{E}[\|\mathbf{c}\|_1] = k$*

The proof is straightforward, and follows from linearity of expectation.

4. Supermodular Minimization

In this section, we shall concentrate on the class $\mathcal{C}_r[0,1]$ of r-threshold boolean functions. As specified previously, any concept in this class can be encoded as a vector $\mathbf{c}$ in $\{0,1\}^n$. For any instance $\mathbf{x}_t$ in $\mathbb{R}^n_+$, the label predicted by $\mathbf{c}$ on $\mathbf{x}_t$ is given by $\mathrm{sgn}(\langle \mathbf{c}, \mathbf{x}_t \rangle - r)$.

Given two concepts $\mathbf{c}$ and $\mathbf{c}'$ in $\{0,1\}^n$, we write $\mathbf{c} \leq \mathbf{c}'$ if $c_i \leq c'_i$ for all $i \in [n]$. It is well-known that the boolean hypercube $\{0,1\}^n$ equipped with the partial ordering $\leq$ is a complete and distributive lattice, in which the meet operation is the boolean conjunction $\wedge$ and the join operation is the boolean disjunction $\vee$. Based on these considerations, a pseudo-boolean function $f : \{0,1\}^n \rightarrow \mathbb{R}$ is called *supermodular* if:

$$f(\mathbf{c} \vee \mathbf{c}') + f(\mathbf{c} \wedge \mathbf{c}') \geq f(\mathbf{c}) + f(\mathbf{c}') \text{ for all } \mathbf{c}, \mathbf{c}' \in \{0,1\}^n$$

In a dual way, f is called *submodular* if $-f$ is supermodular. We say that f is *monotone increasing* (resp. *monotone decreasing*) if, for all $\mathbf{c}, \mathbf{c}' \in \{0,1\}^n$, $\mathbf{c} \leq \mathbf{c}'$ implies $f(\mathbf{c}) \leq f(\mathbf{c}')$ (resp. $f(\mathbf{c}) \geq f(\mathbf{c}')$). In the setting suggested by our optimization problem, the interest of supermodular functions arises from the following property.

Proposition 3 *Let $S = \{\mathbf{x}_t, y_t\}_{t=1}^m$ be a training set and $\gamma > 0$ a margin parameter. Then, the function $f : \{0,1\}^n \rightarrow \mathbb{R}_+$ given by:*

$$f(\mathbf{c}) = \sum_{t=1}^{m} \max[0, \gamma - y_t(\langle \mathbf{c}, \mathbf{x}_t \rangle - r)]$$

is supermodular.

Proof Let g_t and h_t be two supermodular functions such that $g_t - h_t$ is either monotone increasing or monotone decreasing. Then, by (Lovász, 1983, Proposition 2.2.), the function $\max(g_t, h_t)$ is also supermodular. Now, suppose that g_t is the constant function 0 and h_t is the linear function given by

Figure 6: Supermodular Local Search

```
Input: A class $\mathcal{C}_r$ of $r$-threshold boolean concepts; a supermodular function $f$
Initialization: Set $\mathbf{c} \leftarrow \mathbf{0}$

(1) compute $\mathbf{c}^* = \text{argmin}(f(\mathbf{c}') : \|\mathbf{c} - \mathbf{c}'\|_1 = 1)$
(2) if $f(\mathbf{c}^*) < (1 - \epsilon/n^2) f(\mathbf{c})$ set $\mathbf{c} \leftarrow \mathbf{c}^*$ and goto 1
(4) return $\mathbf{c}$
```

$h_t(\mathbf{c}) = \gamma - y_t(\langle \mathbf{c}, \mathbf{x}_t \rangle - r)$. If $y_t = +1$ then $0 - h_t(\mathbf{c})$ is monotone increasing. Dually, if $y_t = -1$ then $0 - h_t(\mathbf{c})$ is monotone decreasing. Therefore, $\max(0, h_t(\mathbf{c}))$ is supermodular. Since supermodularity is preserved under summation, the result follows.

In general, minimizing a supermodular function f over a $\{0,1\}^n$ is NP-hard. However if f is supermodular and monotone decreasing then the minimization problem can be approximated within a ratio of $1 - 1/e$ using a simple greedy algorithm suggested by Nemhauser *et al.* (1978). Many efforts have been focused on designing heuristics for arbitrary supermodular functions. Perhaps one of the most noteworthy algorithms is the local search technique proposed by Vondrák (2007) which yields a deterministic $1/3$-approximation.

In Figure 6 is presented a variant of Vondrák's algorithm, adapted for supermodular minimization. In step (1), the algorithm considers all binary weight vectors at a hamming distance of 1 of the current vector, and selects the best one. In step (2), we have the stopping condition. Note that during a single run, the number of iterations cannot be higher than the total number of features d. During a single iteration, the algorithm considers d weight vectors, and evaluates them on the whole dataset. Thus, the overall complexity of this algorithm is $O(md^2)$.

Based on (Vondrák, 2007, Theorem 2.9.) and Proposition 3, we get the following result.

Theorem 4 *Let $\mathcal{C}_r$ be the class of r-threshold concepts over $\{0,1\}^n$. For any training set $S = \{(x_t, y_t\}_{t=1}^m$, let $\mathbf{c}^*$ be a concept in $\mathcal{C}_r$ that minimizes $L_\gamma(\mathbf{c}; S)$. Then, the supermodular local search algorithm returns a solution $\mathbf{c} \in \mathcal{C}_r$ such that:*

$$L(\mathbf{c}, S) \leq (1/3 - o(1)) L(\mathbf{c}^*, S)$$

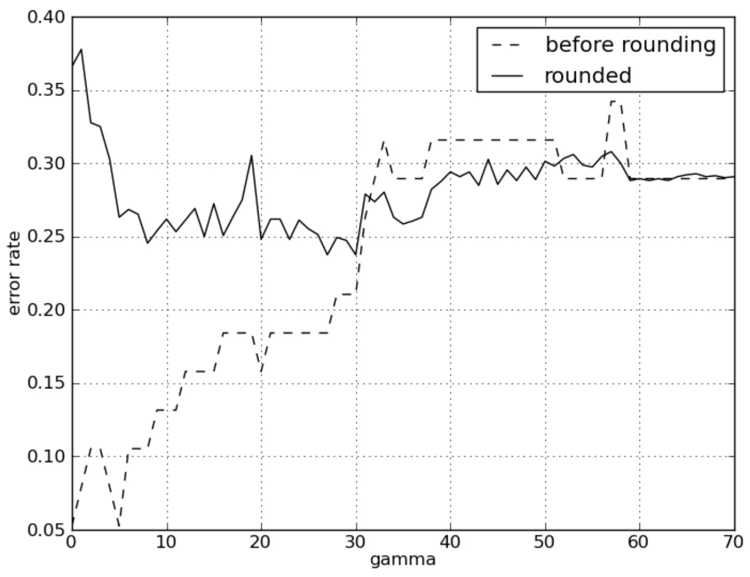

Figure 7: Empirical error rate of the non-rounded linear separator (dashed line) versus error rate of the concept after randomized rounding (straight line).

Concerning the class of zero-threshold concepts over $\{-1, 0, +1\}^n$, we cannot guarantee such a result because the hinge loss is no longer supermodular. However, our local search algorithm can be used as a basis for a random-restart hill climbing technique that conducts a series of local searches from randomly generated initial states until an acceptable accuracy is found. Actually, the random-restart hill-climbing technique is one of the most common approaches to tackle combinatorial optimization problems.

The performance of this technique is evaluated below.

5. Experiments

In this section, we briefly discuss the experiments ran on the metagenomic dataset with the ConvexRounding algorithm as well as the Supermodular Local Search algorithm. As mentioned before, this dataset contains 38 examples and 69 real-valued features. The dataset is divided into two well balanced classes: obese people and non obese. To apply our binary weights learning algorithms to this ternary weights learning problem, the features have been doubled as explained in section 2..

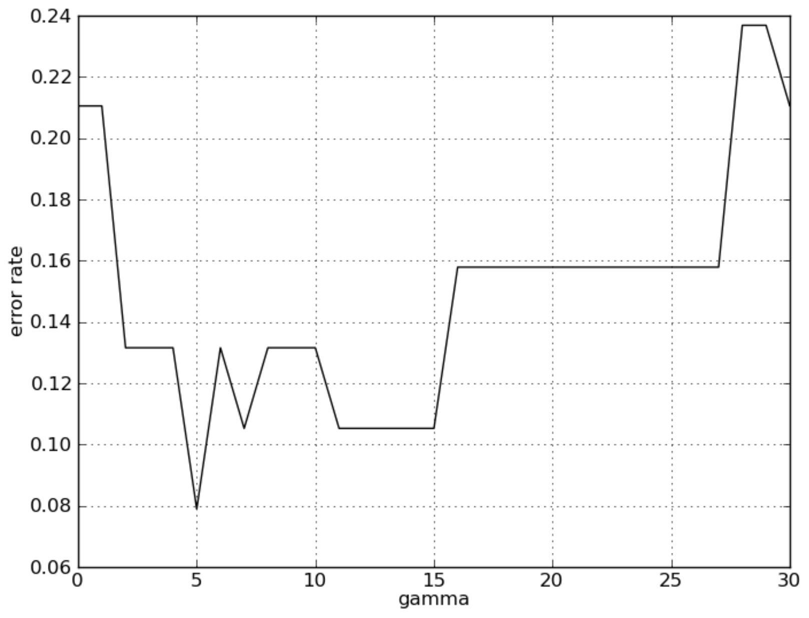

Figure 8: Empirical error rate of the submodular optimization algorithm, for various values of γ.

Figure 7 represents the error rate of the ConvexRounding algorithm. More precisely the upper curve represents the error rate of $\mathbf{c}$ (the rounded vector of weights) and the lower curve shows the error rate of $\hat{\mathbf{w}}$, before the rounding was applied. The x-axis corresponds to various values of γ. It is interesting to note that when the margin is low, the randomized rounding procedure is extremely harmful. This is consistent with the explanation given in section 2.

The Supermodular Local Search (figure 8) is also evaluated for various values of γ. Both algorithms exhibit a 'U' shaped error rate which is not surprising. It appears from these figures that the Supermodular Local Search dominates the Convex Rounding algorithm.

For low values of γ, the rounding procedure results in a huge drop of performance compared to the non-rounded solution. Similarly, for very low values of γ, the submodular optimization algorithm also performs quite badly. This can be explained by the fact that as γ grows, the objective function has less and less non linearity, and thus optimizing this objective function with $0, 1$ constraints becomes much easier.

Also note for $\gamma \in [5, 25]$, the submodular algorithm performs nearly as well as the non-rounded convex relaxation. This is both surprising and non surprising. Let us first look at the non-surprising aspect. Because we have more features than examples, it is likely that there exist many linear separators (including $\{-1, 0, 1\}$ valued separators) achieving low error rates. Thus, the convex relaxation should not yield significantly better results than the optimal ternary weighted separator. The more surprising aspect is that the submodular algorithm reaches an optimal solution, eventhough the problem is known to be NP-hard.

6. Conclusions

In this paper, we have proposed two algorithms to tackle NP-hard learning problems. The first algorithm is based on a convex relaxation (in a similar fashion as the Lasso) and the second algorithm is based on Supermodular maximization procedures. Both algorithms were theoretically analysed and tested on a metagenomic dataset.

The experiments should be considered as very preliminary. In the near-future, we are expecting to have datasets with a very large number of features.

7. Acknowledgments

We are greatly thankful to Dusko Ehrlich and Emmanuelle Le Chatellier for providing us with anonymised data from the MetaHit Project and hinting us towards the search of learning very sparse models.

References

ARUMUGAM M., RAES J., PELLETIER E., LE PASLIER D., YAMADA T., MENDE D. R., FERNANDES G. R., TAP J., BRULS T., BATTO J.-M., BERTALAN M., BORRUEL N., CASELLAS F., FERNANDEZ L., GAUTIER L., HANSEN T., HATTORI M., HAYASHI T., KLEEREBEZEM M., KUROKAWA K., LECLERC M., LEVENEZ F., MANICHANH C., NIELSEN H. B., NIELSEN T., PONS N., POULAIN J., QIN J., SICHERITZ-PONTEN T., TIMS S., TORRENTS D., UGARTE E., ZOETENDAL E. G., WANG J., GUARNER F., PEDERSEN O., DE VOS W. M., BRUNAK S., DORÉ J., CONSORTIUM M., WEISSENBACH J., EHRLICH S. D., BORK P., METAHIT CONSORTIUM (ADDITIONAL MEMBERS), ANTOLÍN M., ARTIGUENAVE F., BLOTTIERE H. M., ALMEIDA M., BRECHOT C., CARA C., CHERVAUX C., CULTRONE A., DELORME C., DENARIAZ G., DERVYN R., FOERSTNER K. U., FRISS C., VAN DE GUCHTE M., GUEDON E., HAIMET F., HUBER W., VAN HYLCKAMA-VLIEG J., JAMET A., JUSTE C., KACI G., KNOL J., LAKHDARI O., LAYEC S., LE ROUX K., MAGUIN E., MÉRIEUX A., MELO MINARDI R., M'RINI C., MULLER J., OOZEER R., PARKHILL J., RENAULT P., RESCIGNO M., SANCHEZ N., SUNAGAWA S., TORREJON A., TURNER K., VANDEMEULEBROUCK G., VARELA E., WINOGRADSKY Y. & ZELLER G. (2011). Enterotypes of the human gut microbiome. *Nature*.

EHRLICH S. D. & CONSORTIUM M. (2010). Metagenomics of the intestinal microbiota: potential applications. *Gastroentérologie Clinique et Biologique*, **34**(S1), S23–S28.

GOLEA M. & MARCHAND M. (1993a). Average case analysis of the clipped Hebb rule for nonoverlapping perception networks. In *Proceedings of the 6th annual conference on computational learning theory (COLT'93)*.

GOLEA M. & MARCHAND M. (1993b). On learning perceptrons with binary weights. *Neural Computation*, **5**(5), 767–782.

HANCZAR B., HUA J. & DOUGHERTY E. R. (2007). Decorrelation of the true and estimated classifier errors in high-dimensional settings. *EURASIP journal on Bioinformatics and Systems Biology*, **2007**, 1–12.

HASTIE T., TIBSHIRANI R. & FRIEDMAN J. (2005). *Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer-Verlag, 2nd edition edition.

IOANNIDIS J. P. (2005). Microarrays and molecular research : noise discovery. *Lancet,*, **365**(454–455).

LOVÁSZ L. (1983). *Mathematical Programming. The State of the Art*, chapter Submodular functions and convexity, p. 235–257. Springer.

NEMHAUSER G. L., WOLSEY L. A. & FISHER M. L. (1978). An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, **14**, 73–87.

QIN J., LI R., RAES J., ARUMUGAM M., BURGDORF K., MANICHANH C., NIELSEN T., PONS N., LEVENEZ F. & YAMADA T. (2010). A human gut microbial gene catalogue established by metagenomic sequencing. *Nature*, **464**(7285), 59–65.

RIESENFELD C., SCHLOSS P. & HANDELSMAN J. (2004). Metagenomics: genomic analysis of microbial communities. *Genetics*, **38**(1), 525.

SHALEV-SHWARTZ S., SREBRO N. & ZHANG T. (2010). Trading accuracy for sparsity in optimization problems with sparsity constraints. *SIAM Journal on Optimization*, **20**(6), 2807–2832.

VONDRÁK J. (2007). *Submodularity in Combinatorial Optimization*. PhD thesis, Department of Applied Mathematics, Prague, Czech Republic.