On Multilabel Classification and Ranking with Partial Feedback

Claudio Gentile
DiSTA, Università dell'Insubria, Italy
claudio.gentile@uninsubria.it

Francesco Orabona
TTI Chicago, USA
francesco@orabona.com

Abstract

We present a novel multilabel/ranking algorithm working in partial information settings. The algorithm is based on 2nd-order descent methods, and relies on upper-confidence bounds to trade-off exploration and exploitation. We analyze this algorithm in a partial adversarial setting, where covariates can be adversarial, but multilabel probabilities are ruled by (generalized) linear models. We show $O(T^{1/2} \log T)$ regret bounds, which improve in several ways on the existing results. We test the effectiveness of our upper-confidence scheme by contrasting against full-information baselines on real-world multilabel datasets, often obtaining comparable performance.

1 Introduction

Consider a book recommendation system. Given a customer's profile, the system recommends a few possible books to the user by means of, e.g., a limited number of banners placed at different positions on a webpage. The system's goal is to select books that the user likes and possibly purchases. Typical feedback in such systems is the actual action of the user or, in particular, what books he has bought/preferred, if any. The system cannot observe what would have been the user's actions had other books got recommended, or had the same book ads been placed in a different order within the webpage. Such problems are collectively referred to as learning with partial feedback. As opposed to the full information case, where the system (the learning algorithm) knows the outcome of each possible response (e.g., the user's action for each and every possible book recommendation placed in the largest banner ad), in the partial feedback setting, the system only observes the response to very limited options and, specifically, the option that was actually recommended. In this and many other examples of this sort, it is reasonable to assume that recommended options are not given the same treatment by the system, e.g., large banners which are displayed on top of the page should somehow be more committing as a recommendation than smaller ones placed elsewhere. Moreover, it is often plausible to interpret the user feedback as a preference (if any) *restricted to* the displayed alternatives.

We consider instantiations of this problem in the multilabel and learning-to-rank settings. Learning proceeds in rounds, in each time step t the algorithm receives an instance $\boldsymbol{x}_t$ and outputs an ordered subset $\hat{Y}_t$ of labels from a finite set of possible labels $[K] = \{1, 2, \ldots, K\}$. Restrictions might apply to the size of $\hat{Y}_t$ (due, e.g., to the number of available slots in the webpage). The set $\hat{Y}_t$ corresponds to the aforementioned recommendations, and is intended to approximate the true set of preferences associated with $\boldsymbol{x}_t$. However, the latter set is never observed. In its stead, the algorithm receives $Y_t \cap \hat{Y}_t$, where $Y_t \subseteq [K]$ is a *noisy version* of the true set of user preferences on $\boldsymbol{x}_t$. When we are restricted to $|\hat{Y}_t| = 1$ for all t, this becomes a multiclass classification problem with bandit feedback – see below.

Related work. This paper lies at the intersection between online learning with partial feedback and multilabel classification/ranking. Both fields include a substantial amount of work, so we can hardly do it justice here. We outline some of the main contributions in the two fields, with an emphasis on those we believe are the most related to this paper.

A well-known and standard tool of facing the problem of partial feedback in online learning is to trade off exploration and exploitation through upper confidence bounds [16]. In the so-called *bandit* setting with contextual information (sometimes called bandits with side information or bandits with covariates, e.g., [3, 4, 5, 7, 15], and references therein) an online algorithm receives at each time step a *context* (typically, in the form of a feature vector $\boldsymbol{x}$) and is compelled to select an action (e.g., a label), whose goodness is quantified by a predefined loss function. Full information about the loss function is not available. The specifics of the interaction model determines which pieces of loss will be observed by the algorithm, e.g., the actual value of the loss on the chosen action, some information on more profitable directions on the action space, noisy versions thereof, etc. The overall goal is to compete against classes of functions that map contexts to (expected) losses in a regret sense, that is, to obtain *sublinear* cumulative regret bounds. For instance, [1, 3, 5, 7] work in a finite action space where the mappings context-to-loss for each action are linear (or generalized linear, as in [7]) functions of the features. They all obtain $T^{1/2}$-like regret bounds, where T is the time horizon. This is extended in [15], where the loss function is modeled as a sample from a Gaussian process over the joint context-action space. We are using a similar (generalized) linear modeling here. Linear multiclass classification problems with bandit feedback are considered in, e.g., [4, 11, 13], where either $T^{2/3}$ or $T^{1/2}$ or even logarithmic regret bounds are proven, depending on the noise model and the underlying loss functions.

All the above papers do not consider *structured* action spaces, where the learner is afforded to select *sets* of actions, which is more suitable to multilabel and ranking problems. Along these lines are the papers [10, 14, 19, 20, 22]. The general problem of online minimization of a submodular loss function under both full and bandit information without covariates is considered in [10], achieving a regret $T^{2/3}$ in the bandit case. In [22] the problem of online learning of assignments is considered, where an algorithm is requested to assign positions (e.g., rankings) to sets of items (e.g., ads) with given constraints on the set of items that can be placed in each position. Their problem shares similar motivations as ours but, again, the bandit version of their algorithm does not explicitly take side information into account, and leads to a $T^{2/3}$ regret bound. In [14] the aim is to learn a suitable ordering of the available actions. Among other things, the authors prove a $T^{1/2}$ regret bound in the bandit setting with a multiplicative weight updating scheme. Yet, no contextual information is incorporated. In [20] the ability of selecting sets of actions is motivated by a problem of diverse retrieval in large document collections which are meant to live in a general metric space. The generality of this approach does not lead to strong regret guarantees for specific (e.g., smooth) loss functions. [19] uses a simple linear model for the hidden utility function of users interacting with a web system and providing partial feedback in any form that allows the system to make significant progress in learning this function. A regret bound of $T^{1/2}$ is again provided that depends on the degree of informativeness of the feedback. It is experimentally argued that this feedback is typically made available by a user that clicks on relevant URLs out of a list presented by a search engine. Despite the neatness of the argument, no formal effort is put into relating this information to the context information at hand or to the way data are generated. Finally, the recent paper [2] investigates classes of graphical models for contextual bandit settings that afford richer interaction between contexts and actions leading again to a $T^{2/3}$ regret bound.

The literature on multilabel learning and learning to rank is overwhelming. The wide attention this literature attracts is often motivated by its web-search-engine or recommender-system applications, and many of the papers are experimental in nature. Relevant references include [6, 9, 23], along with references therein. Moreover, when dealing with multilabel, the typical assumption is full supervision, an important concern being modeling correlations among classes. In contrast to that, the specific setting we are considering here need not face such a modeling. Other related references are [8, 12], where learning is by pairs of examples. Yet, these approaches need i.i.d. assumptions on the data, and typically deliver batch learning procedures. To summarize, whereas we are technically close to [1, 3, 4, 5, 7, 15], from a motivational standpoint we are perhaps closest to [14, 19, 22].

Our results. We investigate the multilabel and learning-to-rank problems in a partial feedback scenario with contextual information, where we assume a probabilistic linear model over the labels, although the contexts can be chosen by an adaptive adversary. We consider two families of loss functions, one is a cost-sensitive multilabel loss that generalizes the standard Hamming loss in several respects, the other is a kind of (unnormalized) ranking loss. In both cases, the learning algorithm is maintaining a (generalized) linear predictor for the probability that a given label occurs, the ranking being produced by upper confidence-corrected estimated probabilities. In such settings, we prove

$T^{1/2}\log T$ cumulative regret bounds — these bounds are optimal, up to log factors, when the label probabilities are fully linear in the contexts. A distinguishing feature of our user feedback model is that, unlike previous papers (e.g., [1, 10, 15, 22]), we are not assuming the algorithm is observing a noisy version of the risk function on the currently selected action. In fact, when a generalized linear model is adopted, the mapping context-to-risk turns out to be nonconvex in the parameter space. Furthermore, when operating on structured action spaces this more traditional form of bandit model does not seem appropriate to capture the typical user preference feedback. Our approach is based on having the loss decouple from the label generating model, the user feedback being a noisy version of the gradient of a *surrogate* convex loss associated with the model itself. As a consequence, the algorithm is not directly dealing with the original loss when making exploration. Though the emphasis is on theoretical results, we also validate our algorithms on two real-world multilabel datasets w.r.t. a number of loss functions, showing good comparative performance against simple multilabel/ranking baselines that operate with full information.

2 Model and preliminaries

We consider a setting where the algorithm receives at time t the side information vector $\boldsymbol{x}_t \in \mathbb{R}^d$, is allowed to output at a (possibly ordered) subset $\hat{Y}_t \subseteq [K]$ of the set of possible labels, then the subset of labels $Y_t \subseteq [K]$ associated with $\boldsymbol{x}_t$ is generated, and the algorithm gets as feedback $\hat{Y}_t \cap Y_t$. The loss suffered by the algorithm may take into account several things: the *distance* between Y_t and $\hat{Y}_t$ (both viewed as sets), as well as the *cost* for playing $\hat{Y}_t$. The cost $c(\hat{Y}_t)$ associated with $\hat{Y}_t$ might be given by the sum of costs suffered on each class $i \in \hat{Y}_t$, where we possibly take into account the *order* in which i occurs within $\hat{Y}_t$ (viewed as an ordered list of labels). Specifically, given constant $a \in [0,1]$ and costs $c = \{c(i,s), i = 1,\ldots,s, s \in [K]\}$, such that $1 \geq c(1,s) \geq c(2,s) \geq \ldots c(s,s) \geq 0$, for all $s \in [K]$, we consider the loss function

$$\ell_{a,c}(Y_t,\hat{Y}_t) = a\,|Y_t \setminus \hat{Y}_t| + (1-a)\,\textstyle\sum_{i\in\hat{Y}_t\setminus Y_t} c(j_i,|\hat{Y}_t|),$$

where j_i is the position of class i in $\hat{Y}_t$, and $c(j_i,\cdot)$ depends on $\hat{Y}_t$ only through its size $|\hat{Y}_t|$. In the above, the first term accounts for the false negative mistakes, hence there is no specific ordering of labels therein. The second term collects the loss contribution provided by all false positive classes, taking into account through the costs $c(j_i,|\hat{Y}_t|)$ the order in which labels occur in $\hat{Y}_t$. The constant a serves as weighting the relative importance of false positive vs. false negative mistakes. As a specific example, suppose that $K = 10$, the costs $c(i,s)$ are given by $c(i,s) = (s-i+1)/s, i = 1,\ldots,s$, the algorithm plays $\hat{Y}_t = (4,3,6)$, but Y_t is $\{1,3,8\}$. In this case, $|Y_t \setminus \hat{Y}_t| = 2$, and $\sum_{i\in\hat{Y}_t\setminus Y_t} c(j_i,|\hat{Y}_t|) = 3/3+1/3$, i.e., the cost for mistakingly playing class 4 in the top slot of $\hat{Y}_t$ is more damaging than mistakingly playing class 6 in the third slot. In the special case when all costs are unitary, there is no longer need to view $\hat{Y}_t$ as an ordered collection, and the above loss reduces to a standard Hamming-like loss between sets Y_t and $\hat{Y}_t$, i.e., $a\,|Y_t \setminus \hat{Y}_t| + (1-a)\,|\hat{Y}_t \setminus Y_t|$. Notice that the partial feedback $\hat{Y}_t \cap Y_t$ allows the algorithm to know which of the chosen classes in $\hat{Y}_t$ are good or bad (and to what extent, because of the selected ordering within $\hat{Y}_t$). Yet, the algorithm does not observe the value of $\ell_{a,c}(Y_t,\hat{Y}_t)$ because $Y_t \setminus \hat{Y}_t$ remains hidden.

Working with the above loss function makes the algorithm's output $\hat{Y}_t$ become a *ranked* list of classes, where ranking is restricted to the deemed relevant classes only. In our setting, only a relevance feedback among the selected classes is observed (the set $Y_t \cap \hat{Y}_t$), but no supervised ranking information (e.g., in the form of pairwise preferences) is provided to the algorithm within this set. Alternatively, we can think of a ranking framework where restrictions on the size of $\hat{Y}_t$ are set by an exogenous (and possibly time-varying) parameter of the problem, and the algorithm is required to provide a ranking complying with these restrictions. More on the connection to the ranking setting with partial feedback is in Section 4.

The problem arises as to which noise model we should adopt so as to encompass significant real-world settings while at the same time affording *efficient implementation* of the resulting algorithms. For any subset $Y_t \subseteq [K]$, we let $(y_{1,t},\ldots,y_{K,t}) \in \{0,1\}^K$ be the corresponding indicator vector. Then it is easy to see that $\ell_{a,c}(Y_t,\hat{Y}_t) = a\sum_{i=1}^K y_{i,t} + (1-a)\sum_{i\in\hat{Y}_t}\Big(c(j_i,|\hat{Y}_t|) - \big(\frac{a}{1-a} + c(j_i,|\hat{Y}_t|)\big)\,y_{i,t}\Big)$. Moreover, because the first sum does not depend on $\hat{Y}_t$, for the sake of optimizing over $\hat{Y}_t$ we can equivalently define

$$\ell_{a,c}(Y_t, \hat{Y}_t) = (1-a) \sum_{i \in \hat{Y}_t} \Big(c(j_i, |\hat{Y}_t|) - \Big(\tfrac{a}{1-a} + c(j_i, |\hat{Y}_t|) \Big) \, y_{i,t} \Big) \ . \qquad (1)$$

Let $\mathbb{P}_t(\cdot)$ be a shorthand for the conditional probability $\mathbb{P}_t(\cdot \,|\, \boldsymbol{x}_t)$, where the side information vector $\boldsymbol{x}_t$ can in principle be generated by an adaptive adversary as a function of the past. Then $\mathbb{P}_t(y_{1,t}, \ldots, y_{K,t}) = \mathbb{P}(y_{1,t}, \ldots, y_{K,t} \,|\, \boldsymbol{x}_t)$, where the marginals $\mathbb{P}_t(y_{i,t} = 1)$ satisfy[1]

$$\mathbb{P}_t(y_{i,t} = 1) = \frac{g(-\boldsymbol{u}_i^\top \boldsymbol{x}_t)}{g(\boldsymbol{u}_i^\top \boldsymbol{x}_t) + g(-\boldsymbol{u}_i^\top \boldsymbol{x}_t)}, \qquad i = 1, \ldots, K, \qquad (2)$$

for some K vectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_K \in \mathcal{R}^d$ and some (known) function $g \,:\, D \subseteq \mathcal{R} \to \mathcal{R}^+$. The model is well defined if $\boldsymbol{u}_i^\top \boldsymbol{x} \in D$ for all i and all $\boldsymbol{x} \in \mathcal{R}^d$ chosen by the adversary. We assume for the sake of simplicity that $||\boldsymbol{x}_t|| = 1$ for all t. Notice that at this point the variables $y_{i,t}$ *need not* be conditionally independent. We are only defining a family of allowed joint distributions $\mathbb{P}_t(y_{1,t}, \ldots, y_{K,t})$ through the properties of their marginals $\mathbb{P}_t(y_{i,t})$.

The function g above will be instantiated to the negative derivative of a suitable convex and nonincreasing loss function L which our algorithm will be based upon. For instance, if L is the square loss $L(\Delta) = (1-\Delta)^2/2$, then $g(\Delta) = 1 - \Delta$, resulting in $\mathbb{P}_t(y_{i,t} = 1) = (1 + \boldsymbol{u}_i^\top \boldsymbol{x}_t)/2$, under the assumption $D = [-1, 1]$. If L is the logistic loss $L(\Delta) = \ln(1 + e^{-\Delta})$, then $g(\Delta) = (e^{\Delta} + 1)^{-1}$, and $\mathbb{P}_t(y_{i,t} = 1) = e^{\boldsymbol{u}_i^\top \boldsymbol{x}_t}/(e^{\boldsymbol{u}_i^\top \boldsymbol{x}_t} + 1)$, with domain $D = \mathcal{R}$.

Set for brevity $\Delta_{i,t} = \boldsymbol{u}_i^\top \boldsymbol{x}_t$. Taking into account (1), this model allows us to write the (conditional) expected loss of the algorithm playing $\hat{Y}_t$ as

$$\mathbb{E}_t[\ell_{a,c}(Y_t, \hat{Y}_t)] = (1-a) \sum_{i \in \hat{Y}_t} \Big(c(j_i, |\hat{Y}_t|) - \Big(\tfrac{a}{1-a} + c(j_i, |\hat{Y}_t|) \Big) \, p_{i,t} \Big) \ , \qquad (3)$$

where $p_{i,t} = \frac{g(-\Delta_{i,t})}{g(\Delta_{i,t}) + g(-\Delta_{i,t})}$, and the expectation $\mathbb{E}_t$ above is w.r.t. the generation of labels Y_t, conditioned on both $\boldsymbol{x}_t$, and all previous $\boldsymbol{x}$ and Y. A key aspect of this formalization is that the Bayes optimal ordered subset $Y_t^* = \mathrm{argmin}_{Y = (j_1, j_2, \ldots, j_{|Y|}) \subseteq [K]} \mathbb{E}_t[\ell_{a,c}(Y_t, Y)]$ can be computed efficiently when knowing $\Delta_{1,t}, \ldots, \Delta_{K,t}$. This is handled by the next lemma. In words, this lemma says that, in order to minimize (3), it suffices to try out all possible sizes $s = 0, 1, \ldots, K$ for Y_t^* and, for each such value, determine the sequence $Y_{s,t}^*$ that minimizes (3) over all sequences of size s. In turn, $Y_{s,t}^*$ can be computed just by sorting classes $i \in [K]$ in decreasing order of $p_{i,t}$, sequence $Y_{s,t}^*$ being given by the first s classes in this sorted list.[2]

Lemma 1. *With the notation introduced so far, let $p_{i_1,t} \geq p_{i_2,t} \geq \ldots p_{i_K,t}$ be the sequence of $p_{i,t}$ sorted in nonincreasing order. Then we have that $Y_t^* = \mathrm{argmin}_{s=0,1,\ldots K} \mathbb{E}_t[\ell_{a,c}(Y_t, Y_{s,t}^*)]$, where $Y_{s,t}^* = (i_1, i_2, \ldots, i_s)$, and $Y_{0,t}^* = \emptyset$.*

Notice the way costs $c(i, s)$ influence the Bayes optimal computation. We see from (3) that placing class i within $\hat{Y}_t$ in position j_i is beneficial (i.e., it leads to a reduction of loss) if and only if $p_{i,t} > c(j_i, |\hat{Y}_t|)/(\frac{a}{1-a} + c(j_i, |\hat{Y}_t|))$. Hence, the higher is the slot i_j in $\hat{Y}_t$ the larger should be $p_{i,t}$ in order for this inclusion to be convenient.[3] *It is Y_t^* that we interpret as the true set of user preferences on $\boldsymbol{x}_t$.*

We would like to compete against the above Y_t^* in a cumulative regret sense, i.e., we would like to bound $R_T = \sum_{t=1}^T \mathbb{E}_t[\ell_{a,c}(Y_t, \hat{Y}_t)] - \mathbb{E}_t[\ell_{a,c}(Y_t, Y_t^*)]$ with high probability. We use a similar but largely more general analysis than [4]'s, to devise an online second-order descent algorithm whose updating rule makes the comparison vector $U = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_K) \in \mathcal{R}^{dK}$ defined through (2) be Bayes optimal w.r.t. a surrogate convex loss $L(\cdot)$ such that $g(\Delta) = -L'(\Delta)$. Observe that the expected loss function (3) is, generally speaking, nonconvex in the margins $\Delta_{i,t}$ (consider, for instance the logistic case $g(\Delta) = \frac{1}{e^{\Delta}+1}$). Thus, we cannot directly minimize this expected loss.

[1] The reader familiar with generalized linear models will recognize the derivative of the function $p(\Delta) = \frac{g(-\Delta)}{g(\Delta)+g(-\Delta)}$ as the (inverse) link function of the associated canonical exponential family of distributions [17].

[2] Due to space limitations, all proofs are given in the supplementary material.

[3] Notice that this depends on the actual size of $\hat{Y}_t$, so we cannot decompose this problem into K independent problems. The decomposition does occur if the costs $c(i, s)$ are constants, independent of i and s, and the criterion for inclusion becomes $p_{i,t} \geq \theta$, for some constant threshold θ.

```
**Parameters:** loss parameters $a \in [0,1]$, cost values $c(i,s)$, interval $D = [-R,R]$, function $g : D \rightarrow \mathcal{R}$, confidence level $\delta \in [0,1]$.
**Initialization:** $A_{i,0} = I \in \mathcal{R}^{d\times d}, i = 1,\dots,K,\ \boldsymbol{w}_{i,1} = 0 \in \mathcal{R}^d,\ i = 1,\dots,K;$

**For** $t = 1, 2 \dots, T$ :
    1. Get instance $\boldsymbol{x}_t \in \mathcal{R}^d\ :\ ||\boldsymbol{x}_t|| = 1;$
    2. For $i \in [K]$, set $\widehat{\Delta}'_{i,t} = \boldsymbol{x}_t^\top \boldsymbol{w}'_{i,t}$, where
       $$\boldsymbol{w}'_{i,t} = \begin{cases} \boldsymbol{w}_{i,t} & \text{if } \boldsymbol{w}_{i,t}^\top \boldsymbol{x}_t \in [-R,R], \\ \boldsymbol{w}_{i,t} - \left(\frac{\boldsymbol{w}_{i,t}^\top \boldsymbol{x}_t - R\,\mathrm{sign}(\boldsymbol{w}_{i,t}^\top \boldsymbol{x}_t)}{\boldsymbol{x}_t^\top A_{i,t-1}^{-1} \boldsymbol{x}_t}\right) A_{i,t-1}^{-1}\boldsymbol{x}_t & \text{otherwise;} \end{cases}$$
    3. Output
       $\hat{Y}_t = \mathrm{argmin}_{Y=(j_1,j_2,\dots j_{|Y|})\subseteq[K]} \left(\sum_{i\in Y}\left(c(j_i,|Y|) - \left(\frac{a}{1-a} + c(j_i,|Y|)\right)\widehat{p}_{i,t}\right)\right)$ ,
       where : $\widehat{p}_{i,t} = \frac{g(-[\widehat{\Delta}'_{i,t}+\epsilon_{i,t}]_D)}{g([\widehat{\Delta}'_{i,t}+\epsilon_{i,t}]_D)+g(-[\widehat{\Delta}'_{i,t}+\epsilon_{i,t}]_D)}$,
       $\epsilon^2_{i,t} = \boldsymbol{x}_t^\top A_{i,t-1}^{-1}\boldsymbol{x}_t \left(U^2 + \frac{d\,c'_L}{(c''_L)^2}\ln\left(1+\frac{t-1}{d}\right) + \frac{12}{c''_L}\left(\frac{c'_L}{c''_L} + 3L(-R)\right)\ln\frac{K(t+4)}{\delta}\right);$
    4. Get feedback $Y_t \cap \hat{Y}_t$;
    5. For $i \in [K]$, update $A_{i,t} = A_{i,t-1} + |s_{i,t}|\boldsymbol{x}_t\boldsymbol{x}_t^\top$, $\boldsymbol{w}_{i,t+1} = \boldsymbol{w}'_{i,t} - \frac{1}{c''_L}A_{i,t}^{-1}\nabla_{i,t}$, where
       $$s_{i,t} = \begin{cases} 1 & \text{If } i \in Y_t \cap \hat{Y}_t \\ -1 & \text{If } i \in \hat{Y}_t \setminus Y_t = \hat{Y}_t \setminus (Y_t \cap \hat{Y}_t) \\ 0 & \text{otherwise;} \end{cases}$$
       and $\nabla_{i,t} = \nabla_{\boldsymbol{w}} L(s_{i,t}\,\boldsymbol{w}^\top\boldsymbol{x}_t)|_{\boldsymbol{w}=\boldsymbol{w}'_{i,t}} = -g(s_{i,t}\,\widehat{\Delta}'_{i,t})\,s_{i,t}\,\boldsymbol{x}_t.$
```

Figure 1: The partial feedback algorithm in the (ordered) multiple label setting.

3 Algorithm and regret bounds

In Figure 1 is our bandit algorithm for (ordered) multiple labels. The algorithm is based on replacing the unknown model vectors $\boldsymbol{u}_1,\dots,\boldsymbol{u}_K$ with prototype vectors $\boldsymbol{w}'_{1,t},\dots,\boldsymbol{w}'_{K,t}$, being $\boldsymbol{w}'_{i,t}$ the time-t approximation to $\boldsymbol{u}_i$, satisying similar constraints we set for the $\boldsymbol{u}_i$ vectors. For the sake of brevity, we let $\widehat{\Delta}'_{i,t} = \boldsymbol{x}_t^\top \boldsymbol{w}'_{i,t}$, and $\Delta_{i,t} = \boldsymbol{u}_i^\top \boldsymbol{x}_t$, $i \in [K]$. The algorithm uses $\widehat{\Delta}'_{i,t}$ as proxies for the underlying $\Delta_{i,t}$ according to the (upper confidence) approximation scheme $\Delta_{i,t} \approx [\widehat{\Delta}'_{i,t} + \epsilon_{i,t}]_D$, where $\epsilon_{i,t} \geq 0$ is a suitable upper-confidence level for class i at time t, and $[\cdot]_D$ denotes the clipping-to-D operation, i.e., $[x]_D = \max(\min(x,R),-R)$. The algorithm's prediction at time t has the same form as the computation of the Bayes optimal sequence Y_t^*, where we replace the true (and unknown) $p_{i,t} = \frac{g(-\Delta_{i,t})}{g(\Delta_{i,t})+g(-\Delta_{i,t})}$ with the corresponding upper confidence proxy $\widehat{p}_{i,t} = \frac{g(-[\widehat{\Delta}'_{i,t}+\epsilon_{i,t}]_D)}{g([\widehat{\Delta}'_{i,t}+\epsilon_{i,t}]_D)+g(-[\widehat{\Delta}'_{i,t}+\epsilon_{i,t}]_D)}$. Computing $\hat{Y}_t$ can be done by mimicking the computation of the Bayes optimal Y_t^* (just replace $p_{i,t}$ by $\widehat{p}_{i,t}$), i.e., order of $K\log K$ running time per prediction. Thus the algorithm is producing a ranked list of relevant classes based on upper-confidence-corrected scores $\widehat{p}_{i,t}$. Class i is deemed relevant and ranked high among the relevant ones when either $\widehat{\Delta}'_{i,t}$ is a good approximation to $\Delta_{i,t}$ and $p_{i,t}$ is large, or when the algorithm is not very confident on its own approximation about i (that is, the upper confidence level $\epsilon_{i,t}$ is large).

The algorithm receives in input the loss parameters a and $c(i,s)$, the model function $g(\cdot)$ and the associated margin domain $D = [-R,R]$, and maintains both K positive definite matrices $A_{i,t}$ of dimension d (initially set to the $d \times d$ identity matrix), and K weight vector $\boldsymbol{w}_{i,t} \in \mathcal{R}^d$ (initially set to the zero vector). At each time step t, upon receiving the d-dimensional instance vector $\boldsymbol{x}_t$ the algorithm uses the weight vectors $\boldsymbol{w}_{i,t}$ to compute the prediction vectors $\boldsymbol{w}'_{i,t}$. These vectors can easily be seen as the result of projecting $\boldsymbol{w}_{i,t}$ onto the space of $\boldsymbol{w}$ where $|\boldsymbol{w}^\top\boldsymbol{x}_t| \leq R$ w.r.t. the distance function $d_{i,t-1}$, i.e., $\boldsymbol{w}'_{i,t} = \mathrm{argmin}_{\boldsymbol{w}\in\mathcal{R}^d\,:\,\boldsymbol{w}^\top\boldsymbol{x}_t\in D}\, d_{i,t-1}(\boldsymbol{w},\boldsymbol{w}_{i,t}), i \in [K]$, where $d_{i,t}(\boldsymbol{u},\boldsymbol{w}) = (\boldsymbol{u}-\boldsymbol{w})^\top A_{i,t}\,(\boldsymbol{u}-\boldsymbol{w})$. Vectors $\boldsymbol{w}'_{i,t}$ are then used to produce prediction values $\widehat{\Delta}'_{i,t}$ involved in the upper-confidence calculation of $\hat{Y}_t \subseteq [K]$. Next, the feedback $Y_t \cap \hat{Y}_t$ is observed, and the algorithm in Figure 1 promotes all classes $i \in Y_t \cap \hat{Y}_t$ (sign $s_{i,t} = 1$), demotes all classes $i \in \hat{Y}_t \setminus Y_t$ (sign $s_{i,t} = -1$), and leaves all remaining classes $i \notin \hat{Y}_t$ unchanged (sign $s_{i,t} = 0$). The update $\boldsymbol{w}'_{i,t} \to \boldsymbol{w}_{i,t+1}$ is based on the gradients $\nabla_{i,t}$ of a loss function $L(\cdot)$ satisfying $L'(\Delta) = -g(\Delta)$. On the other hand, the update $A_{i,t-1} \to A_{i,t}$ uses the rank one matrix[4] $\boldsymbol{x}_t \boldsymbol{x}_t^\top$. In both the update of $\boldsymbol{w}'_{i,t}$ and the one involving $A_{i,t-1}$, the reader should observe the role played by the signs $s_{i,t}$. Finally, the constants c'_L and c''_L occurring in the expression for $\epsilon^2_{i,t}$ are related to smoothness properties of $L(\cdot)$ – see next theorem.

Theorem 2. *Let $L : D = [-R, R] \subseteq \mathcal{R} \to \mathcal{R}^+$ be a $C^2(D)$ convex and nonincreasing function of its argument, $(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_K) \in \mathcal{R}^{dK}$ be defined in (2) with $g(\Delta) = -L'(\Delta)$ for all $\Delta \in D$, and such that $\|\boldsymbol{u}_i\| \le U$ for all $i \in [K]$. Assume there are positive constants c_L, c'_L and c''_L such that: i. $\frac{L'(\Delta)\,L''(-\Delta)+L''(\Delta)\,L'(-\Delta)}{(L'(\Delta)+L'(-\Delta))^2} \ge -c_L$ and ii. $(L'(\Delta))^2 \le c'_L$, and iii. $L''(\Delta) \ge c''_L$ hold for all $\Delta \in D$. Then the cumulative regret R_T of the algorithm in Figure 1 satisfies, with probability at least $1-\delta$,*

$$R_T = O\left((1-a)\, c_L\, K \sqrt{T\, C\, d \ln\left(1 + \tfrac{T}{d}\right)}\right),$$

where $C = O\left(U^2 + \frac{d\, c'_L}{(c''_L)^2} \ln\left(1 + \frac{T}{d}\right) + \left(\frac{c'_L}{(c''_L)^2} + \frac{L(-R)}{c''_L}\right) \ln \frac{KT}{\delta}\right)$.

It is easy to see that when $L(\cdot)$ is the square loss $L(\Delta) = (1-\Delta)^2/2$ and $D = [-1, 1]$, we have $c_L = 1/2$, $c'_L = 4$ and $c''_L = 1$; when $L(\cdot)$ is the logistic loss $L(\Delta) = \ln(1 + e^{-\Delta})$ and $D = [-R, R]$, we have $c_L = 1/4$, $c'_L \le 1$ and $c''_L = \frac{1}{2(1+\cosh(R))}$, where $\cosh(x) = \frac{e^x + e^{-x}}{2}$.

Remark 1. *A drawback of Theorem 2 is that, in order to properly set the upper confidence levels $\epsilon_{i,t}$, we assume prior knowledge of the norm upper bound U. Because this information is often unavailable, we present here a simple modification to the algorithm that copes with this limitation. We change the definition of $\epsilon^2_{i,t}$ in Figure 1 to $\epsilon^2_{i,t} = \max\left\{ \boldsymbol{x}^\top A^{-1}_{i,t-1} \boldsymbol{x} \left(\frac{2\, d\, c'_L}{(c''_L)^2} \ln\left(1 + \frac{t-1}{d}\right) + \frac{12}{c''_L} \left(\frac{c'_L}{c''_L} + 3L(-R) \right) \ln \frac{K(t+4)}{\delta} \right), 4\, R^2 \right\}$. This immediately leads to the following result.*

Theorem 3. *With the same assumptions and notation as in Theorem 2, if we replace $\epsilon^2_{i,t}$ as explained above we have that, with probability at least $1-\delta$, R_T satisfies*

$$R_T = O\left((1-a)\, c_L\, K \sqrt{T\, C\, d \ln\left(1 + \tfrac{T}{d}\right)} + (1-a)\, c_L\, K\, R\, d \left(\exp\left(\frac{(c''_L)^2\, U^2}{c'_L\, d}\right) - 1\right)\right).$$

4 On ranking with partial feedback

As Lemma 1 points out, when the cost values $c(i, s)$ in $\ell_{a,c}$ are *stricly* decreasing then the Bayes optimal ordered sequence Y^*_t on $\boldsymbol{x}_t$ can be obtained by sorting classes in decreasing values of $p_{i,t}$, and then decide on a cutoff point[5] induced by the loss parameters, so as to tell relevant classes apart from irrelevant ones. In turn, because $p(\Delta) = \frac{g(-\Delta)}{g(\Delta)+g(-\Delta)}$ is increasing in Δ, this ordering corresponds to sorting classes in decreasing values of $\Delta_{i,t}$. Now, if parameter a in $\ell_{a,c}$ is very close[6] to 1, then $|Y^*_t| = K$, and the algorithm itself will produce ordered subsets $\hat{Y}_t$ such that $|\hat{Y}_t| = K$. Moreover, it does so by receiving *full* feedback on the relevant classes at time t (since $Y_t \cap \hat{Y}_t = Y_t$). As is customary (e.g., [6]), one can view any multilabel assignment $Y = (y_1, \ldots, y_K) \in \{0, 1\}^K$ as a ranking among the K classes in the most natural way: i preceeds j if and only if $y_i > y_j$. The (unnormalized) ranking loss function $\ell_{rank}(Y, \widehat{f})$ between the multilabel Y and a ranking function $\widehat{f} : \mathcal{R}^d \to \mathcal{R}^K$, representing degrees of class relevance sorted in a decreasing order $\widehat{f}_{j_1}(\boldsymbol{x}_t) \ge \widehat{f}_{j_2}(\boldsymbol{x}_t) \ge \ldots \ge \widehat{f}_{j_K}(\boldsymbol{x}_t) \ge 0$, counts the number of class pairs that disagree in the two rankings: $\ell_{rank}(Y, \widehat{f}) = \sum_{i,j \in [K]\,:\, y_i > y_j} \left(\{\widehat{f}_i(\boldsymbol{x}_t) < \widehat{f}_j(\boldsymbol{x}_t)\} + \frac{1}{2} \{\widehat{f}_i(\boldsymbol{x}_t) = \widehat{f}_j(\boldsymbol{x}_t)\} \right)$,

[4] Notice that $A^{-1}_{i,t}$ can be computed incrementally in $O(d^2)$ time per update. [4] and references therein also use *diagonal* approximations thereof, reporting good empirical performance with just $O(d)$ time per update.

[5] This is called the *zero point* in [9].

[6] If $a = 1$, the algorithm only cares about false negative mistakes, the best strategy being always predicting $\hat{Y}_t = [K]$. Unsurprisingly, this yields zero regret in both Theorems 2 and 3.

where $\{\ldots\}$ is the indicator function of the predicate at argument. As pointed out in [6], the ranking function $\widehat{f}(\boldsymbol{x}_t) = (p_{1,t}, \ldots, p_{K,t})$ is also Bayes optimal w.r.t. $\ell_{rank}(Y, \widehat{f})$, *no matter if* the class labels y_i are conditionally independent or not. Hence we can use this algorithm for tackling ranking problems derived from multilabel ones, when the measure of choice is ℓ_{rank} and the feedback is full.

In fact, a partial information version of the above can easily be obtained. Suppose that at each time t, the environment discloses both $\boldsymbol{x}_t$ and a maximal *size* S_t for the ordered subset $\hat{Y}_t = (j_1, j_2, \ldots, j_{|\hat{Y}_t|})$ (both $\boldsymbol{x}_t$ and S_t can be chosen adaptively by an adversary). Here S_t might be the number of available slots in a webpage or the number of URLs returned by a search engine in response to query $\boldsymbol{x}_t$. Then it is plausible to compete in a regret sense against the best time-t offline ranking of the form $f(\boldsymbol{x}_t) = (f_1(\boldsymbol{x}_t), f_2(\boldsymbol{x}_t), \ldots, f_h(\boldsymbol{x}_t), 0, \ldots, 0)$, with $h \le S_t$. Further, the ranking loss could be reasonably restricted to count the number of class pairs disagreeing within $\hat{Y}_t$ plus a quantity related to number of false negative mistakes. E.g., if $\widehat{f}_{j_1}(\boldsymbol{x}_t) \ge \widehat{f}_{j_2}(\boldsymbol{x}_t) \ge \ldots \ge \widehat{f}_{j_{|\hat{Y}_t|}}(\boldsymbol{x}_t)$, we can set (the factor S_t below serves as balancing the contribution of the two main terms):

$$\ell_{rank,t}(Y, \widehat{f}) = \textstyle\sum_{i,j \in \hat{Y}_t \,:\, y_i > y_j} \left(\{\widehat{f}_i(\boldsymbol{x}_t) < \widehat{f}_j(\boldsymbol{x}_t)\} + \frac{1}{2}\,\{\widehat{f}_i(\boldsymbol{x}_t) = \widehat{f}_j(\boldsymbol{x}_t)\} \right) + S_t\, |Y_t \setminus \hat{Y}_t| \,.$$

It is not hard to see that if classes are conditionally independent, $\mathbb{P}_t(y_{1,t}, ..., y_{K,t}) = \prod_{i \in [K]} p_{i,t}$, then the Bayes optimal ranking for $\ell_{rank,t}$ is given by $f^*(\boldsymbol{x}_t; S_t) = (p_{i_1,t}, \ldots, p_{i_{S_t},t}, 0, \ldots, 0)$. If we put on the argmin (Step 3 in Figure 1) the further constraint $|Y| \le S_t$ (we are still sorting classes according to decreasing values of $\widehat{p}_{i,t}$), one can prove the following ranking version of Theorem 2.

Theorem 4. *With the same assumptions and notation as in Theorem 2, let the classes in $[K]$ be conditionally independent, i.e., $\mathbb{P}_t(y_{1,t}, ..., y_{K,t}) = \prod_{i \in [K]} p_{i,t}$ for all t, and let the cumulative regret R_T w.r.t. $\ell_{rank,t}$ be defined as*

$$R_T = \textstyle\sum_{t=1}^{T} \mathbb{E}_t[\ell_{rank,t}(Y_t, (\widehat{p}_{j_1,t}, ..., \widehat{p}_{j_{S_t},t}, 0, ..., 0))] - \mathbb{E}_t[\ell_{rank,t}(Y_t, (p_{i_{1,t}}, ..., p_{i_{S_t},t}, 0, ..., 0))],$$

where $\widehat{p}_{j_1,t} \ge \ldots \ge \widehat{p}_{j_{S_t},t} \ge 0$ and $p_{i_1,t} \ge \ldots \ge p_{i_{S_t},t} \ge 0$. Then, with probability at least $1 - \delta$, we have $R_T = O\left(c_L \sqrt{S\,K\,T\,C\,d\,\ln\left(1 + \frac{T}{d}\right)}\right)$, where $S = \max_{t=1,\ldots,T} S_t$.

The proof (see the appendix) is very similar to the one of Theorem 2. This suggests that, to some extent, we are decoupling the label generating model from the loss function ℓ under consideration. Notice that the linear dependence on the total number of classes K (which is often much larger than S in a multilabel/ranking problem) is replaced by $\sqrt{SK}$. One could get similar benefits out of Theorem 2. Finally, one could also combine Theorem 4 with the argument contained in Remark 1.

5 Experiments and conclusions

The experiments we report here are meant to validate the exploration-exploitation tradeoff implemented by our algorithm under different conditions (restricted vs. nonrestricted number of classes), loss measures ($\ell_{a,c}$, $\ell_{rank,t}$, and Hamming loss) and model/parameter settings (L = square loss, L = logistic loss, with varying R).

Datasets. We used two multilabel datasets. The first one, called Mediamill, was introduced in a video annotation challenge [21]. It comprises 30,993 training samples and 12,914 test ones. The number of features d is 120, and the number of classes K is 101. The second dataset is Sony CSL Paris [18], made up of 16,452 train samples and 16,519 test samples, each sample being described by $d = 98$ features. The number of classes K is 632. In both cases, feature vectors have been normalized to unit L2 norm.

Parameter setting and loss measures. We used the algorithm in Figure 1 with two different loss functions, the square loss and the logistic loss, and varied the parameter R for the latter. The setting of the cost function $c(i, s)$ depends on the task at hand, and for this preliminary experiments we decided to evaluate two possible settings only. The first one, denoted by "decreasing c" is $c(i, s) = \frac{s-i+1}{s}, i = 1, \ldots, s$, the second one, denoted by "constant c", is $c(i, s) = 1$, for all i and s. In all experiments, the a parameter was set to 0.5, so that $\ell_{a,c}$ with constant c reduces to half the Hamming loss. In the decreasing c scenario, we evaluated the performance of the algorithm on the loss $\ell_{a,c}$ that the algorithm is minimizing, but also its ability to produce meaningful (partial)

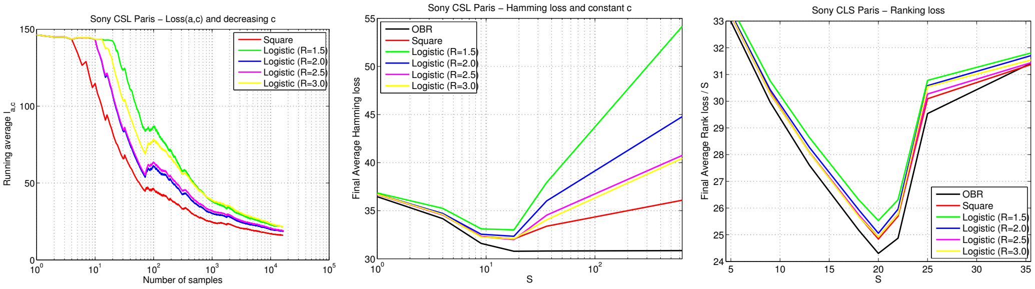

Figure 2: Experiments on the Sony CSL Paris dataset.

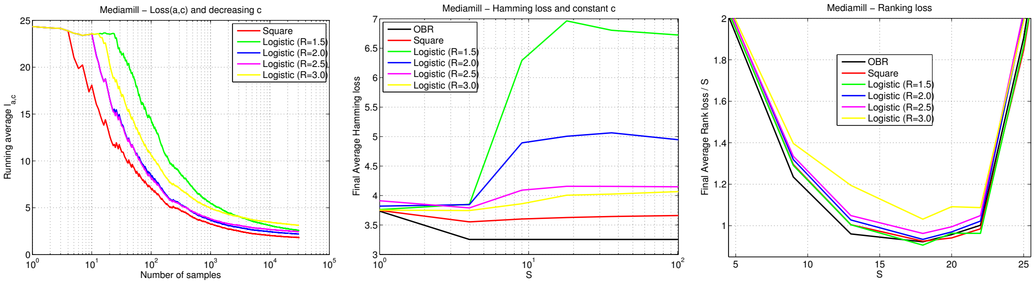

Figure 3: Experiments on the Mediamill dataset.

rankings through $\ell_{rank,t}$. On the constant c setting, we evaluated the Hamming loss. As is typical of multilabel problems, the label *density*, i.e., the average fraction of labels associated with the examples, is quite small. For instance, on Mediamill this is 4.3%. Hence, it is clearly beneficial to impose an upper bound S on $|\hat{Y}_t|$. For the constant c and ranking loss experiments we tried out different values of S, and reported the final performance.

Baseline. As baseline, we considered a full information version of Algorithm 1 using the square loss, that receives after each prediction the full array of true labels Y_t for each sample. We call this algorithm OBR (Online Binary Relevance), because it is a natural online adaptation of the binary relevance algorithm, widely used as a baseline in the multilabel literature. Comparing to OBR stresses the effectiveness of the exploration/exploitation rule above and beyond the details of underlying generalized linear predictor. OBR was used to produce subsets (as in the Hamming loss case), and restricted rankings (as in the case of $\ell_{rank,t}$).

Results. Our results are summarized in Figures 2 and 3. The algorithms have been trained by sweeping only once over the training data. Though preliminary in nature, these experiments allow us to draw a few conclusions. Our results for the avarage $\ell_{a,c}(Y_t, \hat{Y}_t)$ with decreasing c are contained in the two left plots. We can see that the performance is improving over time on both datasets, as predicted by Theorem 2. In the middle plots are the final cumulative Hamming losses with constant c divided by the number of training samples, as a function of S. Similar plots are on the right with the final average ranking losses $\ell_{rank,t}$ divided by S. In both cases we see that there is an optimal value of S that allows to balance the exploration and the exploitation of the algorithm. Moreover the performance of our algorithm is always pretty close to the performance of OBR, even if our algorithm is receiving only partial feedback. In many experiments the square loss seems to give better results. Exception is the ranking loss on the Mediamill dataset (Figure 3, right).

Conclusions. We have used generalized linear models to formalize the exploration-exploitation tradeoff in a multilabel/ranking setting with partial feedback, providing $T^{1/2}$-like regret bounds under semi-adversarial settings. Our analysis decouples the multilabel/ranking loss at hand from the label-generation model. Thanks to the usage of calibrated score values $\widehat{p}_{i,t}$, our algorithm is capable of automatically inferring where to split the ranking between relevant and nonrelevant classes [9], the split being clearly induced by the loss parameters in $\ell_{a,c}$. We are planning on using more general label models that explicitly capture label correlations to be applied to other loss functions (e.g., F-measure, 0/1, average precision, etc.). We are also planning on carrying out a more thorough experimental comparison, especially to full information multilabel methods that take such correlations into account. Finally, we are currenty working on extending our framework to structured output tasks, like (multilabel) hierarchical classification.

References

[1] Y. Abbasi-Yadkori, D. Pal, and C. Szepesvári. Improved algorithms for linear stochastic bandits. In *25th NIPS*, 2011.

[2] K. Amin, M. Kearns, and U. Syed. Graphical models for bandit problems. In *UAI*, 2011.

[3] P. Auer. Using confidence bounds for exploitation-exploration trade-offs. *JMLR*, 3, 2003.

[4] K. Crammer and C. Gentile. Multiclass classification with bandit feedback using adaptive regularization. In *28th ICML*, 2011.

[5] V. Dani, T. Hayes, and S. Kakade. Stochastic linear optimization under bandit feedback. In *21th Colt*, 2008.

[6] K. Dembczynski, W. Waegeman, W. Cheng, and E. Hullermeier. On label dependence and loss minimization in multi-label classification. *Machine Learning*, to appear.

[7] S. Filippi, O. Cappé, A. Garivier, and C. Szepesvári. Parametric bandits: The generalized linear case. In *NIPS*, pages 586–594, 2010.

[8] Y. Freund, R. D. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *JMLR*, 4:933–969, 2003.

[9] J. Furnkranz, E. Hullermeier, E. Loza Menca, and K. Brinker. Multilabel classification via calibrated label ranking. *Machine Learning*, 73:133–153, 2008.

[10] E. Hazan and S. Kale. Online submodular minimization. In *NIPS 22*, 2009.

[11] E. Hazan and S. Kale. Newtron: an efficient bandit algorithm for online multiclass prediction. In *NIPS*, 2011.

[12] R. Herbrich, T. Graepel, and K. Obermayer. Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*. MIT Press, 2000.

[13] S. Kakade, S. Shalev-Shwartz, and A. Tewari. Efficient bandit algorithms for online multiclass prediction. In *25th ICML*, 2008.

[14] S. Kale, L. Reyzin, and R. Schapire. Non-stochastic bandit slate problems. In *24th NIPS*, 2010.

[15] A. Krause and C. S. Ong. Contextual gaussian process bandit optimization. In *25th NIPS*, 2011.

[16] T. H. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Adv. Appl. Math*, 6, 1985.

[17] P. McCullagh and J.A. Nelder. *Generalized linear models*. Chapman and Hall, 1989.

[18] F. Pachet and P. Roy. Improving multilabel analysis of music titles: A large-scale validation of the correction approach. *IEEE Trans. on Audio, Speech, and Lang. Proc.*, 17(2):335–343, February 2009.

[19] P. Shivaswamy and T. Joachims. Online structured prediction via coactive learning. In *29th ICML, 2012*, to appear.

[20] A. Slivkins, F. Radlinski, and S. Gollapudi. Learning optimally diverse rankings over large document collections. In *27th ICML*, 2010.

[21] C. G. M. Snoek, M. Worring, J.C. van Gemert, J.-M. Geusebroek, and A. W. M. Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *Proc. of the 14th ACM international conference on Multimedia*, MULTIMEDIA '06, pages 421–430, New York, NY, USA, 2006.

[22] M. Streeter, D. Golovin, and A. Krause. Online learning of assignments. In *23rd NIPS*, 2009.

[23] G. Tsoumakas, I. Katakis, and I. Vlahavas. Random k-labelsets for multilabel classification. *IEEE Transactions on Knowledge and Data Engineering*, 23:1079–1089, 2011.