Online Boosting Algorithms for Multi-label Ranking

Young Hun Jung
Department of Statistics
University of Michigan
Ann Arbor, MI 48109
yhjung@umich.edu

Ambuj Tewari
Department of Statistics
University of Michigan
Ann Arbor, MI 48109
tewaria@umich.edu

Abstract

We consider the multi-label ranking approach to multi-label learning. Boosting is a natural method for multi-label ranking as it aggregates weak predictions through majority votes, which can be directly used as scores to produce a ranking of the labels. We design online boosting algorithms with provable loss bounds for multi-label ranking. We show that our first algorithm is optimal in terms of the number of learners required to attain a desired accuracy, but it requires knowledge of the edge of the weak learners. We also design an adaptive algorithm that does not require this knowledge and is hence more practical. Experimental results on real data sets demonstrate that our algorithms are at least as good as existing batch boosting algorithms.

1 INTRODUCTION

Multi-label learning has important practical applications (e.g., Schapire and Singer [2000]), and its theoretical properties continue to be studied (e.g., Koyejo et al. [2015]). In contrast to standard multi-class classifications, multi-label learning problems allow multiple correct answers. In other words, we have a fixed set of basic labels, and the actual label is a *subset* of the basic labels. Since the number of subsets increases exponentially as the number of basic labels grows, thinking of each subset as a different class leads to intractability.

It is quite common in applications for the multi-label learner to output a *ranking* of the labels on a new test instance. For example, the popular MULAN library designed by Tsoumakas et al. [2011] allows the output of multi-label learning to be a multi-label ranker. In this paper, we focus on the multi-label ranking (MLR) setting. That is to say, the learner produces a *score vector* such that a label with a higher score will be ranked above a label with a lower score. We are particularly interested in *online* MLR settings where the data arrive sequentially. The online framework is designed to handle a large volume of data that accumulates rapidly. In contrast to classical *batch learners*, which observe the entire training set, online learners do not require the storage of a large amount of data in memory and can also adapt to non-stationarity in the data by updating the internal state as new instances arrive.

Boosting, first proposed by Freund and Schapire [1997], aggregates mildly powerful learners into a strong learner. It has been used to produce state-of-the-art results in a wide range of fields (e.g., Korytkowski et al. [2016] and Zhang and Wang [2014]). Boosting algorithms take weighted majority votes among weak learners' predictions, and the cumulative votes can be interpreted as a score vector. This feature makes boosting very well suited to MLR problems.

The theory of boosting has emerged in batch binary settings and became arguably complete (cf. Schapire and Freund [2012]), but its extension to an online setting is relatively new. To our knowledge, Chen et al. [2012] first introduced an online boosting algorithm with theoretical justifications, and Beygelzimer et al. [2015] pushed the state-of-the-art in online binary settings further by proposing two online algorithms and proving optimality of one. Recent work by Jung et al. [2017] has extended the theory to multi-class settings, but their scope remained limited to single-label problems.

In this paper, we present the first online MLR boosting algorithms along with their theoretical justifications. Our work is mainly inspired by the online single-label work (Jung et al. [2017]). The main contribution is to allow general forms of weak predictions whereas the previous online boosting algorithms only considered

Proceedings of the 21^{st} International Conference on Artificial Intelligence and Statistics (AISTATS) 2018, Lanzarote, Spain. PMLR: Volume 84. Copyright 2018 by the author(s).

homogeneous prediction formats. By introducing a general way to encode weak predictions, our algorithms can combine binary, single-label, and MLR predictions.

After introducing the problem setting, we define an *edge* of an online learner over a random learner (Definition 1). Under the assumption that every weak learner has a known positive edge, we design an optimal way to combine their predictions (Section 3.1). In order to deal with practical settings where such an assumption is untenable, we present an adaptive algorithm that can aggregate learners with arbitrary edges (Section 3.2). In Section 4, we test our two algorithms on real data sets, and find that their performance is often comparable with, and sometimes better than, that of existing batch boosting algorithms for MLR.

2 PRELIMINARIES

The number of candidate labels is fixed to be k, which is known to the learner. Without loss of generality, we may write the labels using integers in $[k] := \{1, \cdots, k\}$. We are allowing multiple correct answers, and the label Y_t is a subset of $[k]$. The labels in Y_t is called *relevant*, and those in Y_t^c, *irrelevant*. At time $t = 1, \cdots, T$, an *adversary* sequentially chooses a labeled example $(\mathbf{x}_t, Y_t) \in \mathcal{X} \times 2^{[k]}$, where $\mathcal{X}$ is some domain. Only the instance $\mathbf{x}_t$ is shown to the learner, and the label Y_t is revealed once the learner makes a prediction $\hat{\mathbf{y}}_t$. As we are interested in MLR settings, $\hat{\mathbf{y}}_t$ is a k dimensional score vector. The learner suffers a loss $L^{Y_t}(\hat{\mathbf{y}}_t)$ where the loss function will be specified later in Section 3.1.

In our boosting framework, we assume that the learner consists of a *booster* and N *weak learners*, where N is fixed before the training starts. This resembles a *manager-worker* framework in that booster distributes tasks by specifying losses, and each learner makes a prediction to minimize the loss. Booster makes the final decision by aggregating weak predictions. Once the true label is revealed, the booster shares this information so that weak learners can update their parameters for the next example.

2.1 Online Weak Learners and Cost Vector

We keep the form of weak predictions $\mathbf{h}_t$ general in that we only assume it is a distribution over $[k]$. This can in fact represent various types of predictions. For example, a *single-label prediction*, $l \in [k]$, can be encoded as a standard basis vector $\mathbf{e}_l$, or a *multi-label prediction* $\{l_1, \cdots, l_n\}$ by $\frac{1}{n}\sum_{i=1}^n \mathbf{e}_{l_i}$. Due to this general format, our boosting algorithm can even combine weak predictions of different formats. This implies that if a researcher has a strong family of binary learners, she can simply boost them without transforming them into multi-class learners through well known techniques such as *one-vs-all* or *one-vs-one* [Allwein et al., 2000].

We extend the *cost matrix framework*, first proposed by Mukherjee and Schapire [2013] and then adopted in online settings by Jung et al. [2017], as a means of communication between booster and weak learners. At round t, booster computes a cost vector $\mathbf{c}_t^i$ for the i^{th} weak learner WL^i, whose prediction $\mathbf{h}_t^i$ suffers the cost $\mathbf{c}_t^i \cdot \mathbf{h}_t^i$. The cost vector is unknown to WL^i until it produces $\mathbf{h}_t^i$, which is usual in online settings. Otherwise, WL^i can trivially minimize the cost.

A binary weak learning condition states a learner can attain over 50% accuracy however the sample weights are assigned. In our setting, cost vectors play the role of sample weights, and we will define the edge of a learner in similar manner.

Finally, we assume that weak learners can take an importance weight as an input, which is possible for many online algorithms.

2.2 General Online Boosting Schema

We introduce a general algorithm schema shared by our algorithms. We denote the weight of WL^i at iteration t by α_t^i. We keep track of weighted cumulative votes through $\mathbf{s}_t^j := \sum_{i=1}^j \alpha_t^i \mathbf{h}_t^i$. That is to say, we can give more credits to well performing learners by setting larger weights. Furthermore, allowing negative weights, we can avoid poor learner's predictions. We call $\mathbf{s}_t^j$ a prediction made by *expert* j. In the end, the booster makes the final decision by following one of these experts.

The schema is summarized in Algorithm 1. We want to emphasize that the true label Y_t is only available once the final prediction $\hat{\mathbf{y}}_t$ is made. Computation of weights and cost vectors requires the knowledge of Y_t, and thus it happens after the final decision is made. To keep our theory general, the schema does not specify which weak learners to use (line 4 and 12). The specific ways to calculate other variables such as α_t^i, $\mathbf{c}_t^i$, and i_t depend on algorithms, which will be introduced in the next section.

3 ALGORITHMS WITH THEORETICAL LOSS BOUNDS

An essential factor in the performance of boosting algorithms is the predictive power of the individual weak learners. For example, if weak learners make completely random predictions, they cannot produce meaningful outcomes according to the booster's intention. We deal with this matter in two different ways. One way is to define an *edge* of a learner over a completely ran-

Algorithm 1 Online Boosting Schema

```
1: Initialize: $\alpha_1^i$ for $i \in [N]$
2: for $t = 1, \cdots, T$ do
3:     Receive example $\mathbf{x}_t$
4:     Gather weak predictions $\mathbf{h}_t^i = WL^i(\mathbf{x}_t),\ \forall i$
5:     Record expert predictions $\mathbf{s}_t^j := \sum_{i=1}^{j} \alpha_t^i \mathbf{h}_t^i$
6:     Choose an index $i_t \in [N]$
7:     Make a final decision $\hat{\mathbf{y}}_t = \mathbf{s}_t^{i_t}$
8:     Get the true label $Y_t$
9:     Compute weights $\alpha_{t+1}^i,\ \forall i$
10:    Compute cost vectors $\mathbf{c}_t^i,\ \forall i$
11:    Weak learners suffer the loss $\mathbf{c}_t^i \cdot \mathbf{h}_t^i$
12:    Weak learners update the internal parameters
13:    Update booster's parameters, if any
14: end for
```

dom learner and assume all weak learners have positive edges. Another way is to measure each learner's *empirical edge* and manipulate the weight α_t^i to maximize the accuracy of the final prediction. Even a learner that is worse than random guessing can contribute positively if we allow negative weights. The first method leads to OnlineBMR (Section 3.1), and the second to Ada.OLMR (Section 3.2).

3.1 Optimal Algorithm

We first define the edge of a learner. Recall that weak learners suffer losses determined by cost vectors. Given the true label Y, the booster chooses a cost vector from

$$\mathcal{C}_0^{eor} := \{\mathbf{c} \in [0,1]^k \mid \max_{l \in Y} \mathbf{c}[l] \le \min_{r \notin Y} \mathbf{c}[r], \\ \min_l \mathbf{c}[l] = 0 \text{ and } \max_l \mathbf{c}[l] = 1\},$$

where the name $\mathcal{C}_0^{eor}$ is used by Jung et al. [2017] and "eor" stands for *edge-over-random*. Since the booster wants weak learners to put higher scores at the relevant labels, costs at the relevant labels should be less than those at the irrelevant ones. Restriction to $[0,1]^k$ makes sure that the learner's cost is bounded. Along with cost vectors, the booster passes the importance weights $w_t \in [0,1]$ so that the learner's cost becomes $w_t \mathbf{c}_t \cdot \mathbf{h}_t$.

We also construct a *baseline* learner that has edge γ. Its prediction $\mathbf{u}_\gamma^Y$ is also a distribution over $[k]$ that puts γ more probability for the relevant labels. That is to say, we can write

$$\mathbf{u}_\gamma^Y[l] = \begin{cases} a + \gamma & \text{if } l \in Y \\ a & \text{if } l \notin Y, \end{cases}$$

where the value of a depends on the number of relevant labels, $|Y|$.

Now we state our online weak learning condition.

Definition 1. (OnlineWLC) *For parameters $\gamma, \delta \in (0,1)$, and $S > 0$, a pair of an online learner and an adversary is said to satisfy OnlineWLC (δ, γ, S) if for any T, with probability at least $1 - \delta$, the learner can generate predictions that satisfy*

$$\sum_{t=1}^{T} w_t \boldsymbol{c}_t \cdot \boldsymbol{h}_t \le \sum_{t=1}^{T} w_t \boldsymbol{c}_t \cdot \boldsymbol{u}_\gamma^{Y_t} + S.$$

γ is called an edge, and S an excess loss.

This extends the condition made by Jung et al. [2017, Definition 1]. The probabilistic statement is needed as many online learners produce randomized predictions. The excess loss can be interpreted as a *warm-up period*. Throughout this section, we assume our learners satisfy OnlineWLC (δ, γ, S) with a fixed adversary.

Cost Vectors The optimal design of a cost vector depends on the choice of loss. We will use $L^Y(\mathbf{s})$ to denote the loss without specifying it where $\mathbf{s}$ is the predicted score vector. The only constraint that we impose on our loss is that it is *proper*, which implies that it is decreasing in $\mathbf{s}[l]$ for $l \in Y$, and increasing in $\mathbf{s}[r]$ for $r \notin Y$ (readers should note that "proper loss" has at least one other meaning in the literature).

Then we introduce *potential function*, a well known concept in game theory which is first introduced to boosting by Schapire [2001]:

$$\begin{aligned} \phi_t^0(\mathbf{s}) &:= L^{Y_t}(\mathbf{s}) \\ \phi_t^i(\mathbf{s}) &:= \mathbb{E}_{l \sim \mathbf{u}_\gamma^{Y_t}} \phi_t^{i-1}(\mathbf{s} + \mathbf{e}_l). \end{aligned} \tag{1}$$

The potential $\phi_t^i(\mathbf{s})$ aims to estimate booster's final loss when i more weak learners are left until the final prediction and $\mathbf{s}$ is the current state. It can be easily shown by induction that many attributes of L are inherited by potentials. Being proper or convex are good examples.

Essentially, we want to set

$$\mathbf{c}_t^i[l] := \phi_t^{N-i}(\mathbf{s}_t^{i-1} + \mathbf{e}_l), \tag{2}$$

where $\mathbf{s}_t^{i-1}$ is the prediction of expert $i-1$. The proper property inherited by potentials ensures the relevant labels have less costs than the irrelevant. To satisfy the boundedness condition of $\mathcal{C}_0^{eor}$, we normalize (2) to get

$$\mathbf{d}_t^i[l] := \frac{\mathbf{c}_t^i[l] - \min_r \mathbf{c}_t^i[r]}{\mathbf{w}^i[t]}, \tag{3}$$

where $\mathbf{w}^i[t] := \max_r \mathbf{c}_t^i[r] - \min_r \mathbf{c}_t^i[r]$. Since Definition 1 assumes that $w_t \in [0,1]$, we have to further normalize $\mathbf{w}^i[t]$. This requires the knowledge of $w^{i*} := \max_t \mathbf{w}^i[t]$. This is unavailable until we observe all the instances, which is fine because we only need this value in proving the loss bound.

Algorithm Details The algorithm is named by OnlineBMR (Online Boost-by-majority for Multi-label Ranking) as its potential function based design has roots in the classical *boost-by-majority* algorithm (Schapire [2001]). In OnlineBMR, we simply set $\alpha_t^i = 1$, or in other words, the booster takes simple cumulative votes. Cost vectors are computed using (2), and the booster always follows the last expert N, or $i_t = N$. These datails are summarized in Algorithm 2.

Algorithm 2 OnlineBMR Details

```
1: **Initialize:** $\alpha_1^i = 1$ for $i \in [N]$
6: Set $i_t = N$
9: Set the weights $\alpha_{t+1}^i = 1,\ \forall i \in [N]$
10: Set $\mathbf{c}_t^i[l] = \phi_t^{N-i}(\mathbf{s}_t^{i-1} + \mathbf{e}_l),\ \forall l \in [k],\ \forall i \in [N]$
13: No extra parameters to be updated
```

The following theorem holds either if weak learners are single-label learners or if the loss L is convex.

Theorem 2. (BMR, General Loss Bound) *For any T and $N \ll \frac{1}{\delta}$, the final loss suffered by OnlineBMR satisfies the following inequality with probability $1 - N\delta$:*

$$\sum_{t=1}^T L^{Y_t}(\hat{\boldsymbol{y}}_t) \leq \sum_{t=1}^T \phi_t^N(\boldsymbol{0}) + S\sum_{i=1}^N w^{i*}. \tag{4}$$

Proof. From (1) and (2), we can write

$$\begin{aligned}\phi_t^{N-i+1}(\mathbf{s}_t^{i-1}) &= \mathbb{E}_{l\sim\mathbf{u}_\gamma^{Y_t}}\phi_t^{N-i}(\mathbf{s}_t^{i-1} + \mathbf{e}_l) \\ &= \mathbf{c}_t^i \cdot \mathbf{u}_\gamma^{Y_t} \\ &= \mathbf{c}_t^i \cdot (\mathbf{u}_\gamma^{Y_t} - \mathbf{h}_t^i) + \mathbf{c}_t^i \cdot \mathbf{h}_t^i \\ &\geq \mathbf{c}_t^i \cdot (\mathbf{u}_\gamma^{Y_t} - \mathbf{h}_t^i) + \phi_t^{N-i}(\mathbf{s}_t^i),\end{aligned}$$

where the last inequality is in fact equality if weak learners are single-label learners, or holds by Jensen's inequality if the loss is convex (which implies the convexity of potentials). Also note that $\mathbf{s}_t^i = \mathbf{s}_t^{i-1} + \mathbf{h}_t^i$. Since both $\mathbf{u}_\gamma^{Y_t}$ and $\mathbf{h}_t^i$ have ℓ_1 norm 1, we can subtract common numbers from every entry of $\mathbf{c}_t^i$ without changing the value of $\mathbf{c}_t^i \cdot (\mathbf{u}_\gamma^{Y_t} - \mathbf{h}_t^i)$. This implies we can plug in $\mathbf{w}^i[t]\mathbf{d}_t^i$ at the place of $\mathbf{c}_t^i$. Then we have

$$\begin{aligned}&\phi_t^{N-i+1}(\mathbf{s}_t^{i-1}) - \phi_t^{N-i}(\mathbf{s}_t^i) \\ &\qquad \geq \mathbf{w}^i[t]\mathbf{d}_t^i \cdot \mathbf{u}_\gamma^{Y_t} - \mathbf{w}^i[t]\mathbf{d}_t^i \cdot \mathbf{h}_t^i.\end{aligned}$$

By summing this over t, we have

$$\begin{aligned}&\sum_{t=1}^T \phi_t^{N-i+1}(\mathbf{s}_t^{i-1}) - \sum_{t=1}^T \phi_t^{N-i}(\mathbf{s}_t^i) \\ &\quad \geq \sum_{t=1}^T \mathbf{w}^i[t]\mathbf{d}_t^i \cdot \mathbf{u}_\gamma^{Y_t} - \sum_{t=1}^T \mathbf{w}^i[t]\mathbf{d}_t^i \cdot \mathbf{h}_t^i.\end{aligned} \tag{5}$$

OnlineWLC (δ, γ, S) provides, with probability $1 - \delta$,

$$\sum_{t=1}^T \frac{\mathbf{w}^i[t]}{w^{i*}}\mathbf{d}_t \cdot \mathbf{h}_t \leq \frac{1}{w^{i*}}\sum_{t=1}^T \mathbf{w}^i[t]\mathbf{d}_t \cdot \mathbf{u}_\gamma^{Y_t} + S.$$

Plugging this in (5), we get

$$\sum_{t=1}^T \phi_t^{N-i+1}(\mathbf{s}_t^{i-1}) - \sum_{t=1}^T \phi_t^{N-i}(\mathbf{s}_t^i) \geq -Sw^{i*}.$$

Now summing this over i, we get with probability $1-N\delta$ (due to union bound),

$$\sum_{t=1}^T \phi_t^N(\mathbf{0}) + S\sum_{i=1}^N w^{i*} \geq \sum_{t=1}^T \phi_t^0(\mathbf{s}_t^N) = \sum_{t=1}^T L^{Y_t}(\hat{\mathbf{y}}_t),$$

which completes the proof. □

Now we evaluate the efficiency of OnlineBMR by fixing a loss. Unfortunately, there is no canonical loss in MLR settings, but following *rank loss* is a strong candidate (cf. Cheng et al. [2010] and Gao and Zhou [2011]):

$$L_{\mathrm{rnk}}^Y(\mathbf{s}) := w_Y \sum_{l\in Y}\sum_{r\notin Y} \mathbb{1}(\mathbf{s}[l] < \mathbf{s}[r]) + \frac{1}{2}\mathbb{1}(\mathbf{s}[l] = \mathbf{s}[r]),$$

where $w_Y = \frac{1}{|Y|\cdot|Y^c|}$ is a normalization constant that ensures the loss lies in $[0, 1]$. Note that this loss is not convex. In case weak learners are in fact single-label learners, we can simply use rank loss to compute potentials, but in more general case, we may use the following *hinge loss* to compute potentials:

$$L_{\mathrm{hinge}}^Y(\mathbf{s}) := w_Y \sum_{l\in Y}\sum_{r\notin Y} (1 + \mathbf{s}[r] - \mathbf{s}[l])_+,$$

where $(\cdot)_+ := \max(\cdot, 0)$. It is convex and always greater than rank loss, and thus Theorem 2 can be used to bound rank loss. In Appendix A, we bound two terms in the RHS of (4) when the potentials are built upon rank and hinge losses. Here we record the results.

Table 1: Upper Bounds for $\phi_t^N(\mathbf{0})$ and w^{i*}

loss	$\phi_t^N(\mathbf{0})$	w^{i*}
rank loss	$e^{-\frac{\gamma^2 N}{2}}$	$O(\frac{1}{\sqrt{N-i}})$
hinge loss	$(N+1)e^{-\frac{\gamma^2 N}{2}}$	2

For the case that we use rank loss, we can check

$$\sum_{i=1}^N w^{i*} \leq \sum_{i=1}^N O(\frac{1}{\sqrt{N-i}}) \leq O(\sqrt{N}).$$

Combining these results with Theorem 2, we get the following corollary.

Corollary 3. (BMR, Rank Loss Bound) *For any T and $N \ll \frac{1}{\delta}$, OnlineBMR satisfies following rank loss bounds with probability $1 - N\delta$.*

With single-label learners, we have

$$\sum_{t=1}^{T} L_{rnk}^{Y_t}(\hat{\boldsymbol{y}}_t) \leq e^{-\frac{\gamma^2 N}{2}} T + O(\sqrt{N}S), \tag{6}$$

and with general learners, we have

$$\sum_{t=1}^{T} L_{rnk}^{Y_t}(\hat{\boldsymbol{y}}_t) \leq (N+1) e^{-\frac{\gamma^2 N}{2}} T + 2NS. \tag{7}$$

Remark. *When we divide both sides by T, we find the average loss is asymptotically bounded by the first term. The second term determines the sample complexity. In both cases, the first term decreases exponentially as N grows, which means the algorithm does not require too many learners to achieve a desired loss bound.*

Matching Lower Bounds From (6), we can deduce that to attain average loss less than ϵ, OnlineBMR needs $\Omega(\frac{1}{\gamma^2} \ln \frac{1}{\epsilon})$ learners and $\tilde{\Omega}(\frac{S}{\epsilon\gamma})$ samples. A natural question is whether these numbers are optimal. In fact the following theorem constructs a circumstance that matches these bounds up to logarithmic factors. Throughout the proof, we consider k as a fixed constant.

Theorem 4. *For any $\gamma \in (0, \frac{1}{2k})$, $\delta, \epsilon \in (0,1)$, and $S \geq \frac{k \ln(\frac{1}{\delta})}{\gamma}$, there exists an adversary with a family of learners satisfying OnlineWLC (δ, γ, S) such that to achieve error rate less than ϵ, any boosting algorithm requires at least $\Omega(\frac{1}{\gamma^2} \ln \frac{1}{\epsilon})$ learners and $\Omega(\frac{S}{\epsilon\gamma})$ samples.*

Proof. We introduce a sketch here and postpone the complete discussion to Appendix B. We assume that an adversary draws a label Y_t uniformly at random from $2^{[k]} - \{\emptyset, [k]\}$, and the weak learners generate single-label prediction l_t w.r.t. $\mathbf{p}_t \in \Delta[k]$. We manipulate $\mathbf{p}_t$ such that weak learners satisfy OnlineWLC (δ, γ, S) but the best possible performance is close to (6).

Boundedness conditions in $\mathcal{C}_0^{eor}$ and the Azuma-Hoeffding inequality provide that with probability $1-\delta$,

$$\sum_{t=1}^{T} w_t \mathbf{c}_t[l_t] \leq \sum_{t=1}^{T} w_t \mathbf{c}_t \cdot \mathbf{p}_t + \frac{\gamma ||\mathbf{w}||_1}{k} + \frac{k \ln(\frac{1}{\delta})}{2\gamma}.$$

For the optimality of the number of learners, we let $\mathbf{p}_t = \mathbf{u}_{2\gamma}^{Y_t}$ for all t. The above inequality guarantees OnlineWLC is met. Then a similar argument of Schapire and Freund [2012, Section 13.2.6] can show that the optimal choice of weights over the learners is $(\frac{1}{N}, \cdots, \frac{1}{N})$. Finally, adopting the argument in the proof of Jung et al. [2017, Theorem 4], we can show

$$\mathbb{E} L_{\text{rnk}}^{Y}(\hat{\mathbf{y}}_t) \geq \Omega(e^{-4Nk^2\gamma^2}).$$

Setting this value equal to ϵ, we have $N \geq \Omega(\frac{1}{\gamma^2} \ln \frac{1}{\epsilon})$, considering k as a fixed constant. This proves the first part of the theorem.

For the second part, let $T_0 := \frac{S}{4\gamma}$ and define $\mathbf{p}_t = \mathbf{u}_0^{Y_t}$ for $t \leq T_0$ and $\mathbf{p}_t = \mathbf{u}_{2\gamma}^{Y_t}$ for $t > T_0$. Then OnlineWLC can be shown to be met in a similar fashion. Observing that weak learners do not provide meaningful information for $t \leq T_0$, we can claim any online boosting algorithm suffers a loss at least $\Omega(T_0)$. Therefore to obtain the certain accuracy ϵ, the number of instances T should be at least $\Omega(\frac{T_0}{\epsilon}) = \Omega(\frac{S}{\epsilon\gamma})$, which completes the second part of the proof. □

3.2 Adaptive Algorithm

Despite the optimal loss bound, OnlineBMR has a few drawbacks when it is applied in practice. Firstly, potentials do not have a closed form, and their computation becomes a major bottleneck (cf. Table 3). Furthermore, the edge γ becomes an extra tuning parameter, which increases the runtime even more. Finally, it is possible that learners have different edges, and assuming a constant edge can lead to inefficiency. To overcome these drawbacks, rather than assuming positive edges for weak learners, our second algorithm chooses the weight α_t^i adaptively to handle variable edges.

Surrogate Loss Like other adaptive boosting algorithms (e.g., Beygelzimer et al. [2015] and Freund et al. [1999]), our algorithm needs a surrogate loss. The choice of loss is broadly discussed by Jung et al. [2017], and logistic loss seems to be a valid choice in online settings as its gradient is uniformly bounded. In this regard, we will use the following *logistic loss*:

$$L_{\text{log}}^{Y}(\mathbf{s}) := w_Y \sum_{l \in Y} \sum_{r \notin Y} \log(1 + \exp(\mathbf{s}[r] - \mathbf{s}[l])).$$

It is proper and convex. We emphasize that booster's prediction suffers the rank loss, and this surrogate only plays an intermediate role in optimizing parameters.

Algorithm Details The algorithm is inspired by Jung et al. [2017, Adaboost.OLM], and we call it by Ada.OLMR[1]. Since it internally aims to minimize the logistic loss, we set the cost vector to be the gradient of the surrogate:

$$\mathbf{c}_t^i := \nabla L_{\text{log}}^{Y_t}(\mathbf{s}_t^{i-1}). \tag{8}$$

Next we present how to set the weights α_t^i. Essentially, Ada.OLMR wants to choose α_t^i to minimize the cumulative logistic loss:

$$\sum_t L_{\text{log}}^{Y_t}(\mathbf{s}_t^{i-1} + \alpha_t^i \mathbf{h}_t^i).$$

[1]Online, Logistic, Multi-label, and Ranking

After initializing α_1^i equals to 0, we use *online gradient descent* method, proposed by Zinkevich [2003], to compute the next weights. If we write $f_t^i(\alpha) := L_{\log}^{Y_t}(\mathbf{s}_t^{i-1} + \alpha \mathbf{h}_t^i)$, we want α_t^i to satisfy

$$\sum_t f_t^i(\alpha_t^i) \leq \min_{\alpha \in F} \sum_t f_t^i(\alpha) + R^i(T),$$

where F is some *feasible set*, and $R^i(T)$ is a sublinear regret. To apply the result by Zinkevich [2003, Theorem 1], f_t^i needs to be convex, and F should be compact. The former condition is met by our choice of logistic loss, and we will use $F = [-2, 2]$ for the feasible set. Since the booster's loss is invariant under the scaling of weights, we can shrink the weights to fit in F.

Taking derivative, we can check $f_t^{i\prime}(\alpha) \leq 1$. Now let $\Pi(\cdot)$ denote a projection onto F: $\Pi(\cdot) := \max\{-2, \min\{2, \cdot\}\}$. By setting

$$\alpha_{t+1}^i = \Pi(\alpha_t^i - \eta_t f_t^{i\prime}(\alpha_t^i)) \text{ where } \eta_t = \frac{1}{\sqrt{t}},$$

we get $R^i(T) \leq 9\sqrt{T}$. Considering that $\mathbf{s}_t^i = \mathbf{s}_t^{i-1} + \alpha_t^i \mathbf{h}_t^i$, we can also write $f_t^{i\prime}(\alpha_t^i) = \mathbf{c}_t^{i+1} \cdot \mathbf{h}_t^i$.

Finally, it remains to address how to choose i_t. In contrast to OnlineBMR, we cannot show that the last expert is reliably sophisticated. Instead, what can be shown is that at least one of the experts is good enough. Thus we use classical *Hedge algorithm* (cf. Freund and Schapire [1997] and Littlestone and Warmuth [1989]) to randomly choose an expert at each iteration with adaptive probability distribution depending on each expert's prediction history. In particular, we introduce new variables v_t^i, which are initialized as $v_1^i = 1$, $\forall i$. At each iteration, i_t is randomly drawn such that

$$\mathbb{P}(i_t = i) \propto v_t^i,$$

and then v_t^i is updated based on the expert's rank loss:

$$v_{t+1}^i := v_t^i e^{-L_{\text{rnk}}^{Y_t}(\mathbf{s}_t^i)}.$$

The details are summarized in Algorithm 3.

Algorithm 3 Ada.OLMR Details

```
 1: Initialize: $\alpha_1^i = 0$ and $v_1^i = 1$, $\forall i \in [N]$
 6: Randomly draw $i_t$ s.t. $\mathbb{P}(i_t = i) \propto v_t^i$
 9: Compute $\alpha_{t+1}^i = \Pi(\alpha_t^i - \frac{1}{\sqrt{t}} f_t^{i\prime}(\alpha_t^i))$, $\forall i \in [N]$
10: Compute $\mathbf{c}_t^i = \nabla L_{\log}^{Y_t}(\mathbf{s}_t^{i-1})$, $\forall i \in [N]$
13: Update $v_{t+1}^i = v_t^i e^{-L_{\text{rnk}}^{Y_t}(\mathbf{s}_t^i)}$, $\forall i \in [N]$
```

Empirical Edges As we are not imposing OnlineWLC, we need another measure of the learner's predictive power to prove the loss bound. From (8), it can be observed that the relevant labels have negative costs and the irrelevant ones have positive cost. Furthermore, the summation of entries of $\mathbf{c}_t^i$ is exactly 0. This observation suggests a new definition of weight:

$$\begin{aligned} \mathbf{w}^i[t] &:= w_{Y_t} \sum_{l \in Y_t} \sum_{r \notin Y_t} \frac{1}{1 + \exp(\mathbf{s}_t^{i-1}[l] - \mathbf{s}_t^{i-1}[r])} \\ &= -\sum_{l \in Y_t} \mathbf{c}_t^i[l] = \sum_{r \notin Y_t} \mathbf{c}_t^i[r] = \frac{||\mathbf{c}_t^i||_1}{2}. \end{aligned} \tag{9}$$

This does not directly correspond to the weight used in (3), but plays a similar role. Then we define the *empirical edge*:

$$\gamma_i := -\frac{\sum_{t=1}^T \mathbf{c}_t^i \cdot \mathbf{h}_t^i}{||\mathbf{w}^i||_1}. \tag{10}$$

The baseline learner $\mathbf{u}_\gamma^{Y_t}$ has this value exactly γ, which suggests that it is a good proxy for the edge defined in Definition 1.

Now we present the loss bound of Ada.OLMR.

Theorem 5. (Ada.OLMR, Rank loss bound) *For any T and N, with probability $1 - \delta$, the rank loss suffered by Ada.OLMR is bounded as follows:*

$$\sum_{t=1}^T L_{rnk}^{Y_t}(\hat{\boldsymbol{y}}_t) \leq \frac{8}{\sum_i |\gamma_i|} T + \tilde{O}(\frac{N^2}{\sum_i |\gamma_i|}), \tag{11}$$

where $\tilde{O}$ notation suppresses dependence on $\log \frac{1}{\delta}$.

Proof. We start the proof by defining the rank loss suffered by expert i as below:

$$M_i := \sum_{t=1}^T L_{\text{rnk}}^{Y_t}(\mathbf{s}_t^i).$$

According to the formula, there is no harm to define $M_0 = \frac{T}{2}$ since $\mathbf{s}_t^0 = \mathbf{0}$. As the booster chooses an expert through the Hedge algorithm, a standard analysis (cf. [Cesa-Bianchi and Lugosi, 2006, Corollary 2.3]) along with the Azuma-Hoeffding inequality provides with probability $1 - \delta$,

$$\sum_{t=1}^T L_{\text{rnk}}^{Y_t}(\hat{\mathbf{y}}_t) \leq 2 \min_i M_i + 2 \log N + \tilde{O}(\sqrt{T}), \tag{12}$$

where $\tilde{O}$ notation suppresses dependence on $\log \frac{1}{\delta}$.

It is not hard to check that $\frac{1}{1+\exp(a-b)} \geq \frac{1}{2}\mathbb{1}(a \leq b)$, from which we can infer

$$\mathbf{w}^i[t] \geq \frac{1}{2} L_{\text{rnk}}^{Y_t}(\mathbf{s}_t^{i-1}) \text{ and } ||\mathbf{w}^i||_1 \geq \frac{M_{i-1}}{2}, \tag{13}$$

where $\mathbf{w}^i$ is defined in (9). Note that this relation holds for the case $i = 1$ as well.

Now let Δ_i denote the difference of the cumulative logistic loss between two consecutive experts:

$$\begin{aligned}\Delta_i &:= \sum_{t=1}^{T} L_{\log}^{Y_t}(\mathbf{s}_t^i) - L_{\log}^{Y_t}(\mathbf{s}_t^{i-1}) \\ &= \sum_{t=1}^{T} L_{\log}^{Y_t}(\mathbf{s}_t^{i-1} + \alpha_t^i \mathbf{h}_t^i) - L_{\log}^{Y_t}(\mathbf{s}_t^{i-1}).\end{aligned}$$

Then the online gradient descent algorithm provides

$$\begin{aligned}\Delta_i \le \min_{\alpha \in [-2,2]} \sum_{t=1}^{T} [L_{\log}^{Y_t}(\mathbf{s}_t^{i-1} + \alpha \mathbf{h}_t^i) - L_{\log}^{Y_t}(\mathbf{s}_t^{i-1})] \\ + 9\sqrt{T}.\end{aligned} \tag{14}$$

Here we record an univariate inequality:

$$\begin{aligned}\log(1+e^{s+\alpha}) - \log(1+e^s) &= \log(1 + \frac{e^\alpha - 1}{1+e^{-s}}) \\ &\le \frac{1}{1+e^{-s}}(e^\alpha - 1).\end{aligned}$$

We expand the difference to get

$$\begin{aligned}&\sum_{t=1}^{T} [L_{\log}^{Y_t}(\mathbf{s}_t^{i-1} + \alpha \mathbf{h}_t^i) - L_{\log}^{Y_t}(\mathbf{s}_t^{i-1})] \\ &= \sum_{t=1}^{T} \sum_{l \in Y_t} \sum_{r \notin Y_t} \log \frac{1 + e^{\mathbf{s}_t^{i-1}[r] - \mathbf{s}_t^{i-1}[l] + \alpha(\mathbf{h}_t^i[r] - \mathbf{h}_t^i[l])}}{1 + e^{\mathbf{s}_t^{i-1}[r] - \mathbf{s}_t^{i-1}[l]}} \\ &\le \sum_{t=1}^{T} \sum_{l \in Y_t} \sum_{r \notin Y_t} \frac{1}{1 + e^{\mathbf{s}_t^{i-1}[l] - \mathbf{s}_t^{i-1}[r]}} (e^{\alpha(\mathbf{h}_t^i[r] - \mathbf{h}_t^i[l])} - 1) \\ &=: f(\alpha).\end{aligned} \tag{15}$$

We claim that $\min_{\alpha \in [-2,2]} f(\alpha) \le -\frac{|\gamma_i|}{2} ||\mathbf{w}^i||_1$. Let us rewrite $||\mathbf{w}^i||_1$ in (9) and γ_i in (10) as following.

$$\begin{aligned}||\mathbf{w}^i||_1 &= \sum_{t=1}^{T} \sum_{l \in Y_t} \sum_{r \notin Y_t} \frac{1}{1 + e^{\mathbf{s}_t^{i-1}[l] - \mathbf{s}_t^{i-1}[r]}} \\ \gamma_i &= \sum_{t=1}^{T} \sum_{l \in Y_t} \sum_{r \notin Y_t} \frac{1}{||\mathbf{w}^i||_1} \frac{\mathbf{h}_t^i[l] - \mathbf{h}_t^i[r]}{1 + e^{\mathbf{s}_t^{i-1}[l] - \mathbf{s}_t^{i-1}[r]}}.\end{aligned} \tag{16}$$

For the ease of notation, let j denote an index that moves through all tuples of $(t, l, r) \in [T] \times Y_t \times Y_t^c$, and a_j and b_j denote following terms.

$$\begin{aligned}a_j &= \frac{1}{||\mathbf{w}^i||_1} \frac{1}{1 + e^{\mathbf{s}_t^{i-1}[l] - \mathbf{s}_t^{i-1}[r]}} \\ b_j &= \mathbf{h}_t^i[l] - \mathbf{h}_t^i[r].\end{aligned}$$

Then from (16), we have $\sum_j a_j = 1$ and $\sum_j a_j b_j = \gamma_i$. Now we express $f(\alpha)$ in terms of a_j and b_j as below.

$$\frac{f(\alpha)}{||\mathbf{w}^i||_1} = \sum_j a_j (e^{-\alpha b_j} - 1) \le e^{-\alpha \sum_j a_j b_j} - 1 = e^{-\alpha \gamma_i} - 1,$$

where the inequality holds by Jensen's inequality. From this, we can deduce that

$$\min_{\alpha \in [-2,2]} \frac{f(\alpha)}{||\mathbf{w}^i||_1} \le e^{-2|\gamma_i|} - 1 \le -\frac{|\gamma_i|}{2},$$

where the last inequality can be checked by investigating $|\gamma_i| = 0, 1$ and observing the convexity of the exponential function. This proves our claim that

$$\min_{\alpha \in [-2,2]} f(\alpha) \le -\frac{|\gamma_i|}{2} ||\mathbf{w}^i||_1. \tag{17}$$

Combining (13), (14), (15) and (17), we have

$$\Delta_i \le -\frac{|\gamma_i|}{4} M_{i-1} + 9\sqrt{T}.$$

Summing over i, we get by telescoping rule

$$\begin{aligned}&\sum_{t=1}^{T} L_{\log}^{Y_t}(\mathbf{s}_t^N) - \sum_{t=1}^{T} L_{\log}^{Y_t}(\mathbf{0}) \\ &\le -\frac{1}{4} \sum_{i=1}^{N} |\gamma_i| M_{i-1} + 9N\sqrt{T} \\ &\le -\frac{1}{4} \sum_{i=1}^{N} |\gamma_i| \min_i M_i + 9N\sqrt{T}.\end{aligned}$$

Note that $L_{\log}^{Y_t}(\mathbf{0}) = \log 2$ and $L_{\log}^{Y_t}(\mathbf{s}_t^N) \ge 0$. Therefore we have

$$\min_i M_i \le \frac{4 \log 2}{\sum_i |\gamma_i|} T + \frac{36N\sqrt{T}}{\sum_i |\gamma_i|}.$$

Plugging this in (12), we get with probability $1 - \delta$,

$$\begin{aligned}\sum_{t=1}^{T} L_{\mathrm{rnk}}^{Y_t}(\hat{\mathbf{y}}_t) &\le \frac{8 \log 2}{\sum_i |\gamma_i|} T + \tilde{O}(\frac{N\sqrt{T}}{\sum_i |\gamma_i|} + \log N) \\ &\le \frac{8}{\sum_i |\gamma_i|} T + \tilde{O}(\frac{N^2}{\sum_i |\gamma_i|}),\end{aligned}$$

where the last inequality holds from AM-GM inequality: $cN\sqrt{T} \le \frac{c^2N^2 + T}{2}$. This completes our proof. ☐

Comparison with OnlineBMR We finish this section by comparing our two algorithms. For a fair comparison, assume that all learners have edge γ. Since the baseline learner $\mathbf{u}_\gamma^Y$ has empirical edge γ, for sufficiently large T, we can deduce that $\gamma_i \ge \gamma$ with high probability. Using this relation, (11) can be written as

$$\sum_{t=1}^{T} L_{\mathrm{rnk}}^{Y_t}(\hat{\mathbf{y}}_t) \le \frac{8}{N\gamma} T + \tilde{O}(\frac{N}{\gamma}).$$

Comparing this to either (6) or (7), we can see that OnlineBMR indeed has better asymptotic loss bound and sample complexity. Despite this sub-optimality (in upper bounds), Ada.OLMR shows comparable results in real data sets due to its adaptive nature.

4 EXPERIMENTS

We performed an experiment on benchmark data sets taken from MULAN[2]. We chose these four particular data sets because Dembczynski and Hüllermeier [2012] already provided performances of batch setting boosting algorithms, giving us a benchmark to compare with. The authors in fact used five data sets, but *image* data set is no longer available from the source. Table 2 summarizes the basic statistics of data sets, including training and test set sizes, number of features and labels, and three statistics of the sizes of relevant sets. The data set *m-reduced* is a reduced version of *mediamill* obtained by random sampling without replacement. We keep the original split for training and test sets to provide more relevant comparisons.

Table 2: Summary of Data Sets

data	#train	#test	dim	k	min	mean	max
emotions	391	202	72	6	1	1.87	3
scene	1211	1196	294	6	1	1.07	3
yeast	1500	917	103	14	1	4.24	11
mediamill	30993	12914	120	101	0	4.38	18
m-reduced	1500	500	120	101	0	4.39	13

VFDT algorithms presented by Domingos and Hulten [2000] were used as weak learners. Every algorithm used 100 trees whose parameters were randomly chosen. VFDT is trained using single-label data, and we fed individual relevant labels along with importance weights that were computed as $\max_l \mathbf{c}_t - \mathbf{c}_t[l]$. Instead of using all covariates, the booster fed to trees randomly chosen 20 covariates to make weak predictions less correlated.

All computations were carried out on a Nehalem architecture 10-core 2.27 GHz Intel Xeon E7-4860 processors with 25 GB RAM per core. Each algorithm was trained at least ten times[3] with different random seeds, and the results were aggregated through mean. Predictions were evaluated by rank loss. The algorithm's loss was only recorded for test sets, but it kept updating its parameters while exploring test sets as well.

Since VFDT outputs a conditional distribution, which is not of a single-label format, we used hinge loss to compute potentials. Furthermore, OnlineBMR has an additional parameter of edge γ. We tried four different values[4], and the best result is recorded as *best BMR*. Table 3 summarizes the results.

Table 3: Average Loss and Runtime in seconds

data	batch[5]	Ada.OLMR		best BMR	
emotions	.1699	.1600	253	.1654	611
scene	.0720	.0881	341	.0743	1488
yeast	.1820	.1874	2675	.1836	9170
mediamill	.0665	.0508	69565	-	-
m-reduced	-	.0632	4148	.0630	288204

Two algorithms' average losses are comparable to each other and to batch setting results, but OnlineBMR requires much longer runtimes. Based on the fact that best BMR's performance is reported on the best edge parameter out of four trials, Ada.OLMR is far more favorable in practice. With large number of labels, runtime for OnlineBMR grows rapidly, and it was even impossible to run *mediamill* data within a week, and this was why we produced the reduced version. The main bottleneck is the computation of potentials as they do not have closed form.

5 CONCLUSION

In this paper, we presented two online boosting algorithms that make multi-label ranking (MLR) predictions. The algorithms are quite flexible in their choice of weak learners in that various types of learners can be combined to produce a strong learner. OnlineBMR is built upon the assumption that all weak learners are strictly better than random guessing, and its loss bound is shown to be tight under certain conditions. Ada.OLMR adaptively chooses the weights over the learners so that learners with arbitrary (even negative) edges can be boosted. Despite its suboptimal loss bound, it produces comparable results with OnlineBMR and runs much faster.

Online MLR boosting provides several opportunities for further research. A major issue in MLR problems is that there does not exist a canonical loss. Fortunately, Theorem 2 holds for any proper loss, but Ada.OLMR only has a rank loss bound. An adaptive algorithm that can handle more general losses will be desirable. The existence of an *optimal* adaptive algorithm is another interesting open question.

Acknowledgments

We acknowledge the support of NSF via CAREER grant IIS-1452099 and CIF-1422157 and of the Sloan Foundation via a Sloan Research Fellowship.

[2]Tsoumakas et al. [2011], `http://mulan.sourceforge.net/datasets.html`

[3]OnlineBMR for *m-reduced* was tested 10 times due to long runtimes, and others were tested 20 times

[4]$\{.2, .1, .01, .001\}$ for small k and $\{.05, .01, .005, .001\}$ for large k

[5]The best result from batch boosting algorithms in Dembczynski and Hüllermeier [2012]

References

Robert E Schapire and Yoram Singer. Boostexter: A boosting-based system for text categorization. *Machine learning*, 39(2-3):135–168, 2000.

Oluwasanmi O Koyejo, Nagarajan Natarajan, Pradeep K Ravikumar, and Inderjit S Dhillon. Consistent multilabel classification. In *Advances in Neural Information Processing Systems*, 2015.

Grigorios Tsoumakas, Eleftherios Spyromitros-Xioufis, Jozef Vilcek, and Ioannis Vlahavas. Mulan: A java library for multi-label learning. *Journal of Machine Learning Research*, 12:2411–2414, 2011.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

Marcin Korytkowski, Leszek Rutkowski, and Rafał Scherer. Fast image classification by boosting fuzzy classifiers. *Information Sciences*, 327:175–182, 2016.

Xiao-Lei Zhang and DeLiang Wang. Boosted deep neural networks and multi-resolution cochleagram features for voice activity detection. In *INTERSPEECH*, pages 1534–1538, 2014.

Robert E Schapire and Yoav Freund. *Boosting: Foundations and algorithms*. MIT press, 2012.

Shang-Tse Chen, Hsuan-Tien Lin, and Chi-Jen Lu. An online boosting algorithm with theoretical justifications. In *Proceedings of the International Conference on Machine Learning*, 2012.

Alina Beygelzimer, Satyen Kale, and Haipeng Luo. Optimal and adaptive algorithms for online boosting. In *Proceedings of the International Conference on Machine Learning*, 2015.

Young Hun Jung, Jack Goetz, and Ambuj Tewari. Online multiclass boosting. In *Advances in Neural Information Processing Systems*, 2017.

Erin L Allwein, Robert E Schapire, and Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1(Dec):113–141, 2000.

Indraneel Mukherjee and Robert E Schapire. A theory of multiclass boosting. *Journal of Machine Learning Research*, 14(Feb):437–497, 2013.

Robert E Schapire. Drifting games. *Machine Learning*, 43(3):265–291, 2001.

Weiwei Cheng, Eyke Hüllermeier, and Krzysztof J Dembczynski. Bayes optimal multilabel classification via probabilistic classifier chains. In *Proceedings of the International Conference on Machine Learning*, 2010.

Wei Gao and Zhi-Hua Zhou. On the consistency of multi-label learning. In *Proceedings of the annual Conference on Learning Theory*, 2011.

Yoav Freund, Robert Schapire, and Naoki Abe. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the International Conference on Machine Learning*, 2003.

Nick Littlestone and Manfred K Warmuth. The weighted majority algorithm. In *Foundations of Computer Science, 1989., 30th Annual Symposium on*. IEEE, 1989.

Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

Krzysztof Dembczynski and Eyke Hüllermeier. Consistent multilabel ranking through univariate loss minimization. In *Proceedings of the International Conference on Machine Learning*, 2012.

Pedro Domingos and Geoff Hulten. Mining high-speed data streams. In *Proceedings of the International Conference on Knowledge Discovery and Data mining*, 2000.