Gradient-based Label Binning in Multi-label Classification

Michael Rapp (✉)[1], Eneldo Loza Mencía[1], Johannes Fürnkranz[2], and Eyke Hüllermeier[3]

[1] Knowledge Engineering Group, TU Darmstadt, Darmstadt, Germany
mrapp@ke.tu-darmstadt.de, research@eneldo.net
[2] Computational Data Analysis Group, JKU Linz, Linz, Austria
juffi@faw.jku.at
[3] Heinz Nixdorf Institute, Paderborn University, Paderborn, Germany
eyke@upb.de

Abstract. In multi-label classification, where a single example may be associated with several class labels at the same time, the ability to model dependencies between labels is considered crucial to effectively optimize non-decomposable evaluation measures, such as the Subset 0/1 loss. The gradient boosting framework provides a well-studied foundation for learning models that are specifically tailored to such a loss function and recent research attests the ability to achieve high predictive accuracy in the multi-label setting. The utilization of second-order derivatives, as used by many recent boosting approaches, helps to guide the minimization of non-decomposable losses, due to the information about pairs of labels it incorporates into the optimization process. On the downside, this comes with high computational costs, even if the number of labels is small. In this work, we address the computational bottleneck of such approach — the need to solve a system of linear equations — by integrating a novel approximation technique into the boosting procedure. Based on the derivatives computed during training, we dynamically group the labels into a predefined number of bins to impose an upper bound on the dimensionality of the linear system. Our experiments, using an existing rule-based algorithm, suggest that this may boost the speed of training, without any significant loss in predictive performance.

Keywords: Multi-label classification · Gradient boosting · Rule learning

1 Introduction

Due to its diverse applications, e.g., the annotation of text documents or images, *multi-label classification* (MLC) has become an established topic of research in the machine learning community (see, e.g., [24] or [8] for an overview). Unlike in traditional classification settings, like binary or multi-class classification, when dealing with multi-label data, a single example may correspond to several class labels at the same time. As the labels that are assigned by a predictive model may partially match the true labeling, rather than being correct or incorrect as a whole, the quality of such predictions can be assessed in various ways. Due to this ambiguity, several meaningful evaluation measures with different characteristics have been proposed in the past (see, e.g., [22]). Usually, a single model cannot provide optimal predictions in terms of all of these measures. Moreover, empirical and theoretical results suggest that many measures benefit from the ability to model dependencies between the labels, if such patterns exist in the data [6]. As this is often the case in real-world scenarios, research on MLC is heavily driven by the motivation to capture correlations in the label space.

To account for the different properties of commonly used multi-label measures, the ability to tailor the training of predictive models to a certain target measure is a desirable property of MLC approaches. Methods based on *gradient boosting*, which guide the construction of an ensemble of weak learners towards the minimization of a given loss function, appear to be appealing with regard to this requirement. In fact, several boosting-based approaches for MLC have been proposed in the literature. Though many of these methods are restricted to the use of label-wise decomposable loss functions (e.g., [25], [18] or [10]), including methods that focus on ranking losses (e.g., [11], [5] or [17]), gradient boosting has also been used to minimize non-decomposable losses (e.g., [15] or [1]).

To be able to take dependencies between labels into account, problem transformation methods, such as *Label Powerset* [22], *RAKEL* [23] or *(Probabilistic) Classifier Chains* [4, 16], transform the original learning task into several sub-problems that can be solved by the means of binary classification algorithms. Compared to binary relevance, where each label is considered in isolation, these approaches come with high computational demands. To compensate for this, methods like *HOMER* [21], *Compressed Sensing* [26], *Canonical Correlation Analysis* [19], *Principal Label Space Transformation* [20] or *Label Embeddings* [13, 9, 2] aim to reduce the complexity of the label space to be dealt with by multi-label classifiers. Notwithstanding that such a reduction in complexity is indispensable in cases where thousands or even millions of labels must be handled, it often remains unclear what measure such methods aim to optimize. In this work, we explicitly focus on the minimization of non-decomposable loss functions in cases where the original problem is tractable. We therefore aim at real-world problems with up to a few hundred labels, where such metrics, especially the Subset 0/1 loss, are considered as important quality measures.

As our contribution, we propose a novel method to be integrated into the gradient boosting framework. Based on the derivatives that guide the optimization process, it maps the labels to a predefined number of bins. If the loss function is non-decomposable, this reduction in dimensionality limits the computational efforts needed to evaluate potential weak learners. Unlike the reduction methods mentioned above, our approach dynamically adjusts to different regions in input space for which a learner may predict. Due to the exploitation of the derivatives, the impact of the approximation is kept at a minimum. We investigate the effects on training time and predictive performance using *BOOMER* [15], a boosting algorithm for learning multi-label rules. In general, the proposed method is not limited to rules and can easily be extended to gradient boosted decision trees.

2 Preliminaries

In this section, we briefly recapitulate the multi-label classification setting and introduce the notation used in this work. We also discuss the methodology used to tackle multi-label problems by utilizing the gradient boosting framework.

2.1 Multi-label Classification

We deal with multi-label classification as a supervised learning problem, where a model is fit to labeled training data $\mathcal{D} = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), \dots, (\boldsymbol{x}_N, \boldsymbol{y}_N)\} \subset \mathcal{X} \times \mathcal{Y}$. Each example in such data set is characterized by a vector $\boldsymbol{x} = (x_1, \dots, x_L) \in \mathcal{X}$ that assigns constant values to numerical or nominal attributes $A_1, \dots, A_L$. In addition, an example may be associated with an arbitrary number of labels out of a predefined label set $\mathcal{L} = \{\lambda_1, \dots, \lambda_K\}$. The information, whether individual labels are relevant (1) or irrelevant (-1) to a training example, is specified in the form of a binary label vector $\boldsymbol{y} = (y_1, \dots, y_K) \in \mathcal{Y}$. The goal is to learn a model $f : \mathcal{X} \rightarrow \mathcal{Y}$ that maps any given example to a predicted label vector $\hat{\boldsymbol{y}} = (\hat{y}_1, \dots, \hat{y}_K) \in \mathcal{Y}$. It should generalize beyond the given training examples such that it can be used to obtain predictions for unseen data.

Ideally, the training process can be tailored to a certain loss function such that the predictions minimize the expected risk with respect to that particular loss. In multi-label classification several meaningful loss functions with different characteristics exist. In the literature, one does usually distinguish between label-wise *decomposable* loss functions, such as the Hamming loss (see, e.g., [22] for a definition of this particular loss function), and *non-decomposable* losses. The latter are considered to be particularly difficult to minimize, as it is necessary to take interactions between the labels into account [6]. Among this kind of loss functions is the *Subset 0/1 loss*, which we focus on in this work. Given true and predicted label vectors, it is defined as

$$\ell_{\text{Subs.}}(\boldsymbol{y}_n, \hat{\boldsymbol{y}}_n) \coloneqq [\![\boldsymbol{y}_n \neq \hat{\boldsymbol{y}}_n]\!], \tag{1}$$

where $[\![x]\!]$ evaluates to 1 or 0, if the predicate x is true or false, respectively. As a wrong prediction for a single label is penalized as much as predicting incorrectly for several labels, the minimization of the Subset 0/1 loss is very challenging. Due to its interesting properties and its prominent role in the literature, we consider it as an important representative of non-decomposable loss functions.

2.2 Multivariate Gradient Boosting

We build on a recently proposed extension to the popular gradient boosting framework that enables to minimize decomposable, as well as non-decomposable, loss functions in a multi-label setting [15]. Said approach aims at learning ensembles $F_T = \{f_1, \dots, f_T\}$ that consist of several weak learners. In multi-label classification, each ensemble member can be considered as a predictive function that returns a vector of real-valued confidence scores

$$\hat{\boldsymbol{p}}_n^t = f_t(\boldsymbol{x}_n) = \left(\hat{p}_{n1}^t, \dots, \hat{p}_{nK}^t\right) \in \mathbb{R}^K \tag{2}$$

for any given example. Each confidence score $\hat{p}_{nk}$ expresses a preference towards predicting the corresponding label λ_k as relevant or irrelevant, depending on whether the score is positive or negative. To compute an ensemble's overall prediction, the vectors that are provided by its members are aggregated by calculating the element-wise sum

$$\hat{\boldsymbol{p}}_n = F_T(\boldsymbol{x}_n) = \hat{\boldsymbol{p}}_n^1 + \cdots + \hat{\boldsymbol{p}}_n^T \in \mathbb{R}^K, \tag{3}$$

which can be discretized in a second step to obtain a binary label vector.

An advantage of gradient boosting is the capability to tailor the training process to a certain (surrogate) loss function. Given a loss ℓ, an ensemble should be trained such that the global objective

$$\mathcal{R}(F_T) = \sum_{n=1}^{N} \ell(\boldsymbol{y}_n, \hat{\boldsymbol{p}}_n) + \sum_{t=1}^{T} \Omega(f_t) \tag{4}$$

is minimized. The use of a suitable *regularization term* Ω may help to avoid overfitting and to converge towards a global optimum, if the loss function is not convex.

Gradient boosting is based on constructing an ensemble of additive functions following an iterative procedure, where new ensemble members are added step by step. To direct the step-wise training process towards a model that optimizes the global objective in the limit, (4) is rewritten based on the derivatives of the loss function. Like many recent boosting-based approaches (e.g., [3], [12] or [25]), we rely on the second-order Taylor approximation. Given a loss function that is twice differentiable, this results in the stagewise objective function

$$\widetilde{\mathcal{R}}(f_t) = \sum_{n=1}^{N} \left(\boldsymbol{g}_n \hat{\boldsymbol{p}}_n^t + \frac{1}{2} \hat{\boldsymbol{p}}_n^t H_n \hat{\boldsymbol{p}}_n^t \right) + \Omega(f_t), \tag{5}$$

which should be minimized by the ensemble member that is added at the t-th training iteration. The gradient vector $\boldsymbol{g}_n = (g_{ni})_{1 \leq i \leq K}$ consist of the first-order partial derivatives of ℓ with respect to predictions of the current model for an example $\boldsymbol{x}_n$ and labels $\lambda_1, \ldots, \lambda_K$. Accordingly, the second-order partial derivatives form the Hessian matrix $H_n = (h_{nij})_{1 \leq i,j \leq K}$. The individual gradients and Hessians are formally defined as

$$g_i^n = \frac{\partial \ell}{\partial \hat{p}_{ni}} (\boldsymbol{y}_n, F_{t-1}(\boldsymbol{x}_n)) \quad \text{and} \quad h_{ij}^n = \frac{\partial \ell}{\partial \hat{p}_{ni} \partial \hat{p}_{nj}} (\boldsymbol{y}_n, F_{t-1}(\boldsymbol{x}_n)). \tag{6}$$

The confidence scores that are predicted by an ensemble member f_t for individual labels must be chosen such that the stagewise objective in (5) is minimized. To derive a formula for calculating the predicted scores, the partial derivative of (5) with respect to the prediction for individual labels must be equated to zero. In case of a decomposable loss function, this results in a closed form solution that enables to compute the prediction for each label independently. In the general case, i.e., when the loss function is non-decomposable and the prediction should not be restricted to a single label, one obtains a system of K linear equations

$$(H + R)\, \hat{\boldsymbol{p}} = -\boldsymbol{g}. \tag{7}$$

Whereas the elements of the Hessian matrix H and the gradient vector $\boldsymbol{g}$ can be considered as coefficients and ordinates, the vector $\hat{\boldsymbol{p}}$ consists of the unknowns to be determined. The matrix R is used to take the regularization into account. In this work, we use the L_2 regularization term

$$\Omega_{\mathrm{L2}}\left(f_t\right) = \frac{1}{2} \omega \left\| \hat{\boldsymbol{p}}^t \right\|_2^2 , \tag{8}$$

where $\left\| \boldsymbol{x} \right\|_2$ is the Euclidean norm and $\omega \geq 0$ controls the weight of the regularization. In this particular case, the regularization matrix $R = \operatorname{diag}(\omega)$ is a diagonal matrix with the value ω on the diagonal.

As the target function to be minimized, we use the logistic loss function

$$\ell_{\text{ex.w.-log}}\left(\boldsymbol{y}_n, \hat{\boldsymbol{p}}_n\right) \coloneqq \log\left(1 + \sum_{k=1}^{K} \exp\left(-y_{nk} \hat{p}_{nk}\right)\right), \tag{9}$$

which has previously been used with the BOOMER algorithm as a surrogate for the Subset 0/1 loss and was originally proposed by Amit et al. [1].

2.3 Ensembles of Multi-label Rules

We rely on multi-label rules as the individual building blocks of ensembles that are trained according to the methodology in Section 2.2. In accordance with (2), each rule can be considered as a function

$$f(\boldsymbol{x}) = b(\boldsymbol{x})\, \hat{\boldsymbol{p}} \tag{10}$$

that predicts a vector of confidence scores for a given example.

The *body* of a rule $b : \mathcal{X} \rightarrow \{0, 1\}$ is a conjunction of one or several conditions that compare a given example's value for a particular attribute A_l to a constant using a relational operator like $\leq$ and $>$, if the attribute is numerical, or $=$ and $\neq$, if it is nominal. If an example satisfies all conditions in the body, i.e., if $b(\boldsymbol{x}) = 1$, it is *covered* by the respective rule. In such case, the scores that are contained in the rule's *head* $\hat{\boldsymbol{p}} \in \mathbb{R}^K$ are returned. It assigns a positive or negative confidence score to each label, depending on whether the respective label is expected to be mostly relevant or irrelevant to the examples that belong to the region of the input space $\mathcal{X}$ that is covered by the rule. If an example is not covered, i.e., if $b(\boldsymbol{x}) = 0$, a null vector is returned. In such case, the rule does not have any effect on the overall prediction, as can be seen in (3).

If individual elements in a rule's head are set to zero, the rule does not provide a prediction for the corresponding labels. The experimental results reported by Rapp et al. [15] suggest that *single-label rules*, which only provide a non-zero prediction for a single label, tend to work well for minimizing decomposable loss functions. Compared to *multi-label rules*, which jointly predict for several labels, the induction of such rules is computationally less demanding, as a closed form solution for determining the predicted scores exists. However, the ability of multi-label rules to express local correlations between several labels, which hold for the examples they cover, has been shown to be crucial when it comes to non-decomposable losses.

To construct a rule that minimizes (5), we conduct a top-down greedy search, as it is commonly used in inductive rule learning (see, e.g., [7] for an overview on the topic). Initially, the search starts with an empty body that does not contain any conditions and therefore is satisfied by all examples. By adding new conditions to the body, the rule is successively specialized, resulting in less examples being covered in the process. For each candidate rule that results from adding a condition, the confidence scores to be predicted for the covered examples are calculated by solving (7). By substituting the calculated scores into (5), an estimate of the rule's quality is obtained. Among all possible refinements, the one that results in the greatest improvement in terms of quality is chosen. The search stops as soon as the rule cannot be improved by adding a condition.

Rules are closely related to the more commonly used decision trees, as each tree can be viewed as a set of non-overlapping rules. At each training iteration, a rule-based boosting algorithm focuses on a single region of the input space for which the model can be improved the most. In contrast, gradient boosted decision trees do always provide predictions for the entire input space. Due to their conceptual similarities, the ideas presented in this paper are not exclusive to rules, but can also be applied to decision trees.

3 Gradient-based Label Binning

In this section, we present *Gradient-based Label Binning* (GBLB), a novel method that aims at reducing the computational costs of the multivariate boosting algorithm discussed in Section 2.2. Although the method can be used with any loss function, it is intended for use cases where a non-decomposable loss should be minimized. This is, because it explicitly addresses the computational bottleneck of such training procedure — the need to solve the linear system in (7) — which reduces to an operation with linear complexity in the decomposable case.

3.1 Complexity Analysis

The objective function in (5), each training iteration aims to minimize, depends on gradient vectors and Hessian matrices that correspond to individual training examples. Given K labels, the former consist of K elements, whereas the latter are symmetric matrices with $K\,(K+1)\,/\,2$ non-zero elements, one for each label, as well as for each pair of labels. The induction of a new rule, using a search algorithm as described in Section 2.3, requires to sum up the gradient vectors and Hessian matrices of the covered examples to form the linear system in (7). Instead of computing the sums for each candidate rule individually, the

Algorithm 1: Candidate evaluation without (left) / with GBLB (right)

```
input  : Gradient vector $\boldsymbol{g}$, Hessian matrix $H$, $L_2$ regularization weight $\omega$
output: Predictions $\hat{\boldsymbol{y}}$, quality score $s$, mapping $\boldsymbol{m}$ (if GBLB is used)
1 Regularization matrix $R = \text{diag}(\omega)$             | Mapping $\boldsymbol{m} = \text{MAP\_TO\_BINS}(\boldsymbol{g}, H, \omega)$
                                                              | $\boldsymbol{g}, H, R = \text{AGGREGATE}(\boldsymbol{m}, \boldsymbol{g}, H, \omega)$
2 $\hat{\boldsymbol{y}} = \text{DSYSV}(-\boldsymbol{g}, H + R)$        ▷ cf. (7)   |
3 $s = \text{DDOT}(\hat{\boldsymbol{y}}, \boldsymbol{g}) +$                          | same as left
    $(0.5 \cdot \text{DDOT}(\hat{\boldsymbol{y}}, \text{DSPMV}(\hat{\boldsymbol{y}}, H)))$   ▷ cf. (5)   |
4 return $\hat{\boldsymbol{y}}, s$                                     | return $\hat{\boldsymbol{y}}, s, \boldsymbol{m}$
```

candidates are processed in a predetermined order, such that each one covers one or several additional examples compared to its predecessor (see, e.g., [14] for an early description of this idea). As a result, an update of the sums with complexity $\mathcal{O}(K^2)$ must be performed for each example and attribute that is considered for making up new candidates.

Alg. 1 shows the steps that are necessary to compute the confidence scores to be predicted by an individual candidate rule, as well as a score that assesses its quality, if the loss function is non-decomposable. The modifications that are necessary to implement GBLB are shown to the right of the original lines of code. Originally, the given gradient vector and Hessian matrix, which result from summation over the covered examples, are used as a basis to solve the linear system in (7) using the Lapack routine DSYSV (cf. Alg. 1, line 2). The computation of a corresponding quality score by substituting the calculated scores into (5), involves invocations of the Blas operations DDOT for vector-vector multiplication, as well as DSPMV for vector-matrix multiplication (cf. Alg. 1, line 3). Whereas the operation DDOT comes with linear costs, the DSPMV and DSYSV routines have quadratic and cubic complexity, i.e., $\mathcal{O}(K^2)$ and $\mathcal{O}(K^3)$, respectively[4]. As Alg. 1 must be executed for each candidate rule, it is the computationally most expensive operation that takes part in a multivariate boosting algorithm aimed at the minimization of a non-decomposable loss function.

GBLB addresses the computational complexity of Alg. 1 by mapping the available labels to a predefined number of bins B and aggregating the elements of the gradient vector and Hessian matrix accordingly. If $B \ll K$, this significantly reduces their dimensionality and hence limits the costs of the Blas and Lapack routines. As a result, given that the overhead introduced by the mapping and aggregation functions is small, we expect an overall reduction in training time.

In this work, we do not address the computational costs of summing up the gradients that correspond to individual examples. However, the proposed method has been designed such that it can be combined with methods that are dedicated to this aspect. Albeit restricting themselves to decomposable losses, Si et al. [18] have proposed a promising method that ensures that many gradients evaluate to zero. This approach, which was partly adopted by Zhang and Jung [25], restricts

[4] Information on the complexity of the Blas and Lapack routines used in this work can be found at `http://www.netlib.org/lapack/lawnspdf/lawn41.pdf`.

the labels that must be considered to those with non-zero gradients. However, to maintain sparsity among the gradients, strict requirements must be fulfilled by the loss function. Among many others, the logistic loss function in (9) does not meet these requirements. The approach that is investigated in this work does not impose any restrictions on the loss function.

3.2 Mapping Labels to Bins

GBLB evolves around the idea of assigning the available labels $\lambda_1, \dots, \lambda_K$ to a predefined number of bins $\mathcal{B}_1, \dots, \mathcal{B}_B$ whenever a potential ensemble member is evaluated during training (cf. MAP_TO_BINS in Alg. 1). To obtain the index of the bin, a particular label λ_k should be assigned to, we use a mapping function $m : \mathbb{R} \rightarrow \mathbb{N}^+$ that depends on a given criterion $c_k \in \mathbb{R}$. In this work, we use the criterion

$$c_k = -\frac{g_k}{h_{kk} + \omega}, \tag{11}$$

which takes the gradient and Hessian for the respective label, as well as the L_2 regularization weight, into account. It corresponds to the optimal prediction when considering the label in isolation, i.e., when assuming that the predictions for other labels will be zero. As the criterion can be obtained for each label individually, the computational overhead is kept at a minimum.

Based on the assignments that are provided by a mapping function m, we denote the set of label indices that belong to the b-th bin as

$$\mathcal{B}_b = \left\{ k \in \{1, \dots, K\} \mid m\left(c_k\right) = b \right\}. \tag{12}$$

Labels should be assigned to the same bin if the corresponding confidence scores, which will be presumably be predicted by an ensemble member, are close to each other. If the optimal scores to be predicted for certain labels are very different in absolute size or even differ in their sign, the respective labels should be mapped to different bins. Based on this premise, we limit the number of distinct scores, an ensemble member may predict, by enforcing the restriction

$$\hat{p}_i = \hat{p}_j, \forall i, j \in \mathcal{B}_b. \tag{13}$$

It requires that a single score is predicted for all labels that have been assigned to the same bin. Given that the mentioned prerequisites are met, we expect the difference between the scores that are predicted for a bin and those that are optimal with respect to its individual labels to be reasonably small.

3.3 Equal-width Label Binning

Principally, different approaches to implement the mapping function m are conceivable. We use *equal-width* binning, as this well-known method provides two advantages: First, unlike other methods, such as equal-frequency binning, it does not involve sorting and can therefore be applied in linear time. Second, the boundaries of the bins are chosen such that the absolute difference between the smallest and largest value in a bin, referred to as the *width*, is the same for all bins. As argued in Section 3.2, this is a desirable property in our particular use case. Furthermore, we want to prevent labels, for which the predicted score should be negative, from being assigned to the same bin as labels, for which the prediction should be positive. Otherwise, the predictions would be suboptimal for some of these labels. We therefore strictly separate between *negative* and *positive bins*. Given $B_\ominus$ negative and $B_\oplus$ positive bins, the width calculates as

$$w_\ominus = \frac{\max_\ominus - \min_\ominus}{B_\ominus} \quad \text{and} \quad w_\oplus = \frac{\max_\oplus - \min_\oplus}{B_\oplus}, \tag{14}$$

for the positive and negative bins, respectively. By $\max_\ominus$ and $\max_\oplus$ we denote the largest value in $\{c_1, \dots, c_K\}$ with negative and positive sign, respectively. Accordingly, $\min_\ominus$ and $\min_\oplus$ correspond to the smallest value with the respective sign. Labels for which $c_k = 0$, i.e., labels with zero gradients, can be ignored. As no improvement in terms of the loss function can be expected, we explicitly set the prediction to zero in such case.

Once the width of the negative and positive bins has been determined, the mapping from individual labels to one of the $B = B_\ominus + B_\oplus$ bins can be obtained via the function

$$m_{\text{eq.-width}}(c_k) = \begin{cases} \min\left(\lfloor \frac{c_k - \min_\ominus}{w_\ominus} \rfloor + 1, B_\ominus\right), & \text{if } c_k < 0 \\ \min\left(\lfloor \frac{c_k - \min_\oplus}{w_\oplus} \rfloor + 1, B_\oplus\right) + B_\ominus, & \text{if } c_k > 0. \end{cases} \tag{15}$$

3.4 Aggregation of Gradients and Hessians

By exploiting the restriction introduced in (13), the gradients and Hessians that correspond to labels in the same bin can be aggregated to obtain a gradient vector and a Hessian matrix with reduced dimensions (cf. AGGREGATE in Alg. 1). To derive a formal description of this aggregation, we first rewrite the objective function (5) in terms of sums instead of using vector and matrix multiplications. This results in the formula

$$\widetilde{\mathcal{R}}(f_t) = \sum_{n=1}^{N} \sum_{i=1}^{K} \left(g_i^n \hat{p}_i + \frac{1}{2} \hat{p}_i \left(h_{ii}^n \hat{p}_i + \sum_{\substack{j=1,\\ j \neq i}}^{K} h_{ij}^n \hat{p}_j \right) \right) + \Omega(f_t). \tag{16}$$

Based on the constraint given in (13) and due to the distribution property of the multiplication, the equality

$$\sum_{i=1}^{K} x_i \hat{p}_i = \sum_{j=1}^{B} \left(\hat{p}_j \sum_{i \in \mathcal{B}_j} x_i \right), \tag{17}$$

where x_i is any term dependent on i, holds. It can be used to rewrite (16) in terms of sums over the bins, instead of sums over the individual labels. For

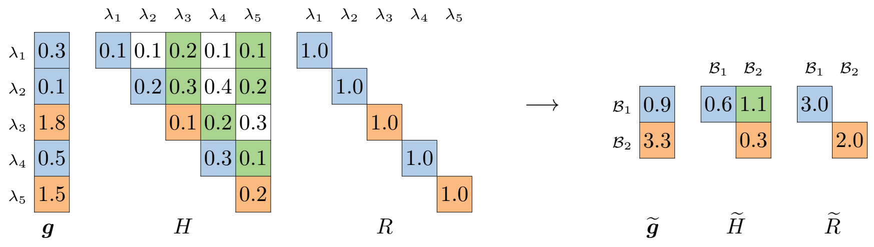

Fig. 1. Illustration of how a gradient vector, a Hessian matrix and a regularization matrix for five labels $\lambda_1, \dots, \lambda_5$ are aggregated with respect to two bins $\mathcal{B}_1 = \{1, 2, 4\}$ and $\mathcal{B}_2 = \{3, 5\}$ when using L_2 regularization with $\omega = 1$. Elements with the same color are added up for aggregation.

brevity, we denote the sum of the gradients, as well as the sum of the elements on the diagonal of the Hessian matrix, that correspond to the labels in bin $\mathcal{B}_b$ as

$$\widetilde{g}_b = \sum_{i \in \mathcal{B}_b} g_i \quad \text{and} \quad \widetilde{h}_{bb} = \sum_{i \in \mathcal{B}_b} h_{ii}. \tag{18}$$

To abbreviate the sum of Hessians that correspond to a pair of labels that have been assigned to different bins $\mathcal{B}_b$ and $\mathcal{B}_q$, we use the short-hand notation

$$\widetilde{h}_{bq} = \sum_{i \in \mathcal{B}_b} \sum_{j \in \mathcal{B}_q} h_{ij}. \tag{19}$$

By exploiting (17) and using the abbreviations introduced above, the objective function in (16) can be rewritten as

$$\widetilde{\mathcal{R}}(f_t) = \sum_{n=1}^{N} \sum_{b=1}^{B} \left(\hat{p}_b \widetilde{g}_b^n + \frac{1}{2} \hat{p}_b \left(\hat{p}_b \widetilde{h}_b^n + \sum_{\substack{q=1, \\ q \neq b}}^{B} \hat{p}_q \widetilde{h}_{bq}^n \right) \right) + \Omega(f_t), \tag{20}$$

which can afterwards be turned into the original notation based on vector and matrix multiplications. The resulting formula

$$\widetilde{\mathcal{R}}(f_t) = \sum_{n=1}^{N} \left(\widetilde{\boldsymbol{g}}_n \hat{\boldsymbol{p}}_n^t + \frac{1}{2} \hat{\boldsymbol{p}}_n^t \widetilde{H}_n \hat{\boldsymbol{p}}_n^t \right) + \Omega(f_t) \tag{21}$$

has the same structure as originally shown in (5). However, the gradient vector $\boldsymbol{g}$ and the Hessian matrix H have been replaced by $\widetilde{\boldsymbol{g}}$ and $\widetilde{H}$, respectively. Consequently, when calculating the scores to be predicted by an ensemble member by solving (7), the coefficients and ordinates that take part in the linear system do not correspond to individual labels, but result from the sums in (18) and (19). As a result, number of linear equations has been reduced from the number of labels K to the number of non-empty bins, which is at most B.

Table 1. Average training times (in seconds) per cross validation fold on different data sets (the number of labels is given in parentheses). The small numbers specify the speedup that results from using GBLB with the number of bins set to 32, 16, 8 and 4% of the labels, or using two bins. Variants that are equivalent to two bins are omitted.

		No	GBLB				
		GBLB	32%	16%	8%	4%	2 bins
Eurlex-sm	(201)	46947	54985 0.85	44872 1.05	38222 1.23	33658 1.39	**21703** 2.16
EukaryotePseAAC	(22)	16033	3593 4.46	2492 6.43	2195 7.30	—	**1534** 10.45
Reuters-K500	(103)	12093	6930 1.75	4197 2.88	3353 3.61	2803 4.31	**2743** 4.41
Bibtex	(159)	2507	2599 0.96	2765 0.91	2649 0.95	2456 1.02	**2125** 1.18
Yeast	(14)	2338	998 2.34	761 3.07	525 4.45	—	**521** 4.49
Birds	(19)	2027	701 2.89	505 4.01	337 6.01	—	**336** 6.03
Yahoo-Social	(39)	1193	261 4.57	217 5.50	192 6.21	**139** 8.58	175 6.82
Yahoo-Computers	(33)	874	172 5.08	134 6.52	126 6.94	**101** 8.65	123 7.11
Yahoo-Science	(40)	735	200 3.67	160 4.59	135 5.44	**106** 6.93	136 5.40
Yahoo-Reference	(33)	571	174 3.28	141 4.05	129 4.43	**110** 5.19	137 4.17
Slashdot	(20)	518	154 3.36	117 4.43	**86** 6.02	—	119 4.35
EukaryoteGO	(22)	191	79 2.42	74 2.58	**60** 3.18	—	64 2.98
Enron	(53)	181	69 2.62	52 3.48	48 3.77	47 3.85	**44** 4.11
Medical	(45)	170	60 2.83	57 2.98	55 3.09	**50** 3.40	51 3.33
Langlog	(75)	132	126 1.05	112 1.18	105 1.26	**101** 1.31	102 1.29
Avg. Speedup			2.81	3.58	4.61	**4.86**	4.00

An example that illustrates the aggregation of a gradient vector and a Hessian matrix is given in Fig. 1. It also takes into account how the regularization matrix R is affected. When dealing with bins instead of individual labels, the L_2 regularization term in (8) becomes

$$\Omega_{\mathrm{L2}}(f_t) = \frac{1}{2}\omega \sum_{b=1}^{B} \left(|\mathcal{B}_b| \, \hat{p}_b^2 \right), \tag{22}$$

where $|\mathcal{B}_b|$ denotes the number of labels that belong to a particular bin. As a consequence, the regularization matrix becomes $\widetilde{R} = \operatorname{diag}(\omega |\mathcal{B}_1|, \dots, \omega |\mathcal{B}_B|)$.

4 Evaluation

To investigate in isolation the effects GBLB has on predictive performance and training time, we chose a single configuration of the BOOMER algorithm as the basis for our experiments. We used 10-fold cross validation to train models that are aimed at the minimization of the Subset 0/1 loss on commonly used benchmark data sets[5]. Each model consists of 5.000 rules that have been learned on varying subsets of the training examples, drawn with replacement. The refinement of rules has been restricted to random subsets of the available attributes.

[5] All data sets are available at `https://www.uco.es/kdis/mllresources`

Table 2. Predictive performance of different approaches in terms of the Subset 0/1 loss and the Hamming loss (smaller values are better).

		Label-	No	GBLB				
		wise	GBLB	32%	16%	8%	4%	2 bins
Subset 0/1 loss	Eurlex-sm	61.63	69.53	45.07	**45.03**	45.30	45.08	47.32
	EukaryotePseAAC	85.09	65.43	65.37	**65.28**	65.68	—	65.52
	Reuters-K500	71.37	71.07	53.70	53.22	53.07	**52.90**	53.40
	Bibtex	85.99	81.31	**77.28**	77.44	77.55	77.32	78.95
	Yeast	84.94	76.54	76.91	76.42	76.87	—	**76.21**
	Birds	45.29	45.30	**45.14**	45.45	45.60	—	46.53
	Yahoo-Social	50.65	64.30	34.49	**34.40**	34.84	35.47	35.37
	Yahoo-Computers	58.04	**46.26**	46.65	47.09	46.94	47.70	47.42
	Yahoo-Science	74.00	85.80	**50.89**	51.20	52.07	52.79	52.04
	Yahoo-Reference	58.19	74.14	**39.82**	40.16	40.73	40.51	40.48
	Slashdot	63.88	**46.62**	46.64	46.64	47.73	—	47.22
	EukaryoteGO	30.63	28.35	28.39	28.24	**28.10**	—	28.55
	Enron	88.19	83.14	83.32	83.32	83.38	**82.91**	82.97
	Medical	28.25	28.82	23.13	**22.62**	23.08	23.23	22.77
	Langlog	79.59	78.84	79.11	79.25	**78.63**	79.45	79.45
Hamming loss	Eurlex-sm	0.55	0.91	0.40	**0.39**	0.40	0.40	0.42
	EukaryotePseAAC	**5.02**	5.65	5.64	5.63	5.67	—	5.66
	Reuters-K500	1.11	1.71	1.11	**1.09**	**1.09**	**1.09**	1.10
	Bibtex	**1.25**	1.45	1.27	1.27	1.27	1.28	1.31
	Yeast	19.75	19.01	18.87	19.08	19.01	—	**18.80**
	Birds	3.91	3.79	3.80	3.79	**3.73**	—	3.87
	Yahoo-Social	1.90	3.81	**1.79**	1.80	1.83	1.87	1.87
	Yahoo-Computers	3.10	**2.97**	3.00	3.02	3.03	3.08	3.06
	Yahoo-Science	2.83	5.85	**2.74**	2.75	2.81	2.84	2.79
	Yahoo-Reference	2.30	4.95	**2.28**	2.30	2.34	2.33	2.32
	Slashdot	**4.02**	4.24	4.24	4.25	4.37	—	4.30
	EukaryoteGO	**1.89**	1.95	1.95	1.94	1.92	—	1.98
	Enron	**4.53**	4.72	4.77	4.77	4.72	4.72	4.73
	Medical	0.84	1.05	0.80	**0.77**	0.79	0.81	0.79
	Langlog	1.52	1.52	**1.50**	1.51	**1.50**	1.52	1.52

As the learning rate and the L_2 regularization weight, we used the default values 0.3 and 1.0, respectively. Besides the original algorithm proposed in [15], we tested an implementation that makes use of GBLB[6]. For a broad analysis, we set the maximum number of bins to 32, 16, 8, and 4% of the available labels. In addition, we investigated an extreme setting with two bins, where all labels with positive and negative criteria are assigned to the same bin, respectively.

Table 1 shows the average time per cross validation fold that is needed by the considered approaches for training. Compared to the baseline that does not use GBLB, the training time can always be reduced by utilizing GBLB with a suitable number of bins. Using fewer bins tends to speed up the training process,

[6] An implementation is available at `https://www.github.com/mrapp-ke/Boomer`

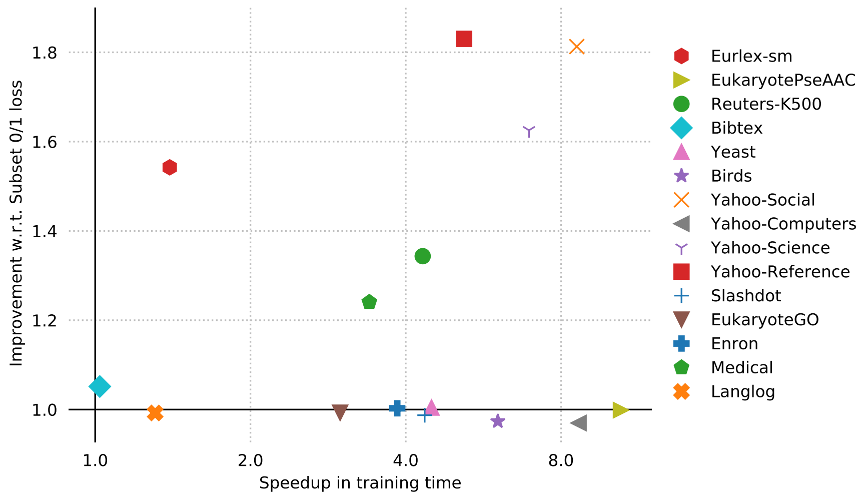

Fig. 2. Relative difference in training time and Subset 0/1 loss (both calculated as the baseline's value divided by the value of the respective approach) per cross validation fold that results from using GBLB with the number of bins set to 4% of the labels.

although approaches that use the fewest bins are not always the fastest ones. On average, limiting the number of bins to 4% of the labels results in the greatest speedup (by factor 5). However, the possible speedup depends on the data set at hand. E.g., on the data set "EukaryotePseAac" the average training time is reduced by factor 10, whereas no significant speedup is achieved for "Bibtex".

To be useful in practice, the speedup that results from GBLB should not come with a significant deterioration in terms of the target loss. We therefore report the predictive performance of the considered approaches in Table 2. Besides the Subset 0/1 loss, which we aim to minimize in this work, we also include the Hamming loss as a commonly used representative of decomposable loss functions. When focusing on the Subset 0/1 loss, we observe that the baseline algorithm without GBLB exhibits subpar performance on some data sets, namely "Eurlex-sm", "Reuters-K500", "Bibtex", "Yahoo-Social", "Yahoo-Science", "Yahoo-Reference" and "Medical". This becomes especially evident when compared to an instantiation of the algorithm that targets the Hamming loss via minimization of a label-wise decomposable logistic loss function (cf. [15], Eq. 6). In said cases, the latter approach performs better even though it is not tailored to the Subset 0/1 loss. Although the baseline performance could most probably be improved by tuning the regularization weight, we decided against parameter tuning, as it exposes an interesting property of GBLB. On the mentioned data sets, approaches that use GBLB appear to be less prone to converge towards local minima. Regardless of the number of bins, they clearly outperform the baseline. According to the Friedman test, these differences are significant with $\alpha = 0.01$. The Nemenyi post-hoc test yields critical distances

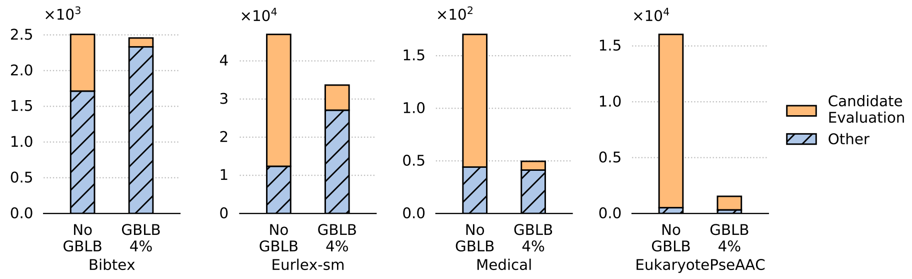

Fig. 3. Average proportion of training time per cross validation fold that is used for the evaluation of candidate rules without GBLB and when using GBLB with the number of bins set to 4% of the labels.

for each of the GBLB-based approaches, when compared to the baseline. On the remaining data sets, where the baseline without GBLB already performs well, the use of GBLB produces competitive results. In these cases, the Friedman test confirms the null hypothesis with $\alpha = 0.1$. An overview of how the training time and the predictive performance in terms of the Subset 0/1 loss is affected, when restricting the number of bins to 4% of the labels, is given in Fig. 2.

To better understand the differences in speedups that may be achieved by using GBLB, a detailed analysis is given in the following for four data sets with varying characteristics. In Fig. 3, we depict the training time that is needed by the baseline approach, as well as by a GBLB-based approach with the number of bins set to 4% of the labels. Besides the total training time, we also show the amount of time spent on the evaluation of candidate rules (cf. Alg. 1), which is the algorithmic aspect addressed by GBLB. For all given scenarios, it can be seen that the time needed for candidate evaluation could successfully be reduced. Nevertheless, the effects on the overall training time vastly differ. On the data sets “Bibtex” and “Eurlex-sm”, the time spent on parts of the algorithm other than the candidate evaluation increased when using GBLB, which is a result of more specific rules being learned. On the one hand, this required more candidates to be evaluated and therefore hindered the overall speedup. On the other hand, the resulting rules clearly outperformed the baseline according to Table 2. On the data set “Bibtex”, even without GBLB, the candidate evaluation was not the most expensive aspect of training. Due to its binary attributes, the number of potential candidates is small compared to the large number of examples. As a result, most of the computation time is spent on summing up the gradients and Hessians of individual examples (cf. Section 3.1). The impact of speeding up the candidate evaluation is therefore limited. On the data sets “Medical” and “EukaryotePseAAC”, where the candidate evaluation was the most expensive aspect to begin with, a significant reduction of training time could be achieved by making that particular operation more efficient. The time spent on other parts of the algorithm remained mostly unaffected in these cases. As mentioned earlier, this includes the summation of gradients and Hessians, which becomes the most time consuming operation when using GBLB. Addressing this aspect holds the greatest potential for further performance improvements.

5 Conclusion

In this work we presented a novel approximation technique for use in multivariate boosting algorithms. Based on the derivatives that guide the training process, it dynamically assigns the available labels to a predefined number of bins. Our experiments, based on an existing rule learning algorithm, confirm that this reduction in dimensionality successfully reduces the training time that is needed for minimizing non-decomposable loss functions, such as the Subset 0/1 loss. According to our results, this speedup does not come with any significant loss in predictive performance. In several cases the proposed method even outperforms the baseline by a large extend due to its ability to overcome local minima without the necessity for extensive parameter tuning.

Despite our promising results, the use of non-decomposable loss functions in the boosting framework remains computationally challenging. Based on the analysis in this paper, we plan to extend our methodology with the ability to exploit sparsity in the label space. When combined with additional measures, the proposed method could become an integral part of more efficient algorithms that are capable of natively minimizing non-decomposable loss functions.

Acknowledgments This work was supported by the German Research Foundation (DFG) under grant number 400845550. Computations were conducted on the Lichtenberg high performance computer of the TU Darmstadt.

References

1. Amit, Y., Dekel, O., Singer, Y.: A boosting algorithm for label covering in multilabel problems. In: Proc. 11th International Conference on AI and Statistics (AISTATS). pp. 27–34 (2007)
2. Bhatia, K., Jain, H., Kar, P., Varma, M., Jain, P.: Sparse local embeddings for extreme multi-label classification. In: Proc. 28th International Conference on Neural Information Processing Systems (NIPS). p. 730–738 (2015)
3. Chen, T., Guestrin, C.: XGBoost: A scalable tree boosting system. In: Proc. 22nd International Conference on Knowledge Discovery and Data Mining (KDD). p. 785–794 (2016)
4. Cheng, W., Hüllermeier, E., Dembczyński, K.: Bayes optimal multilabel classification via probabilistic classifier chains. In: Proc. 27th International Conference on Machine Learning (ICML). pp. 279–286 (2010)
5. Dembczyński, K., Kotłowski, W., Hüllermeier, E.: Consistent multilabel ranking through univariate losses. In: Proc. 29th International Conference on Machine Learning (ICML). pp. 1319–1326 (2012)
6. Dembczyński, K., Waegeman, W., Cheng, W., Hüllermeier, E.: On label dependence and loss minimization in multi-label classification. Machine Learning **88**(1-2), 5–45 (2012)

7. Fürnkranz, J., Gamberger, D., Lavrač, N.: Foundations of Rule Learning. Springer Science & Business Media (2012)
8. Gibaja, E., Ventura, S.: Multi-label learning: A review of the state of the art and ongoing research. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **4**(6), 411–444 (2014)
9. Huang, K.H., Lin, H.T.: Cost-sensitive label embedding for multi-label classification. Machine Learning **106**(9), 1725–1746 (2017)
10. Johnson, M., Cipolla, R.: Improved image annotation and labelling through multi-label boosting. In: Proc. British Machine Vision Conference (BMVC) (2005)
11. Jung, Y.H., Tewari, A.: Online boosting algorithms for multi-label ranking. In: Proc. 21st International Conference on AI and Statistics (AISTATS). pp. 279–287 (2018)
12. Ke, G., Meng, Q., Finely, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: LightGBM: A highly efficient gradient boosting decision tree. In: Proc. 31st International Conference on Neural Information Processing Systems (NIPS) (2017)
13. Kumar, V., Pujari, A.K., Padmanabhan, V., Kagita, V.R.: Group preserving label embedding for multi-label classification. Pattern Recognition **90**, 23–34 (2019)
14. Mehta, M., Agrawal, R., Rissanen, J.: SLIQ: A fast scalable classifier for data mining. In: Proc. International Conference on Extending Database Technology. pp. 18–32 (1996)
15. Rapp, M., Loza Mencía, E., Fürnkranz, J., Nguyen, V.L., Hüllermeier, E.: Learning gradient boosted multi-label classification rules. In: Proc. European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD). pp. 124–140 (2020)
16. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. In: Proc. European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD). pp. 254–269 (2009)
17. Schapire, R.E., Singer, Y.: BoosTexter: A boosting-based system for text categorization. Machine Learning **39**(2), 135–168 (2000)
18. Si, S., Zhang, H., Keerthi, S.S., Mahajan, D., Dhillon, I.S., Hsieh, C.J.: Gradient boosted decision trees for high dimensional sparse output. In: Proc. 34th International Conference on Machine Learning (ICML). pp. 3182–3190 (2017)
19. Sun, L., Ji, S., Ye, J.: Canonical correlation analysis for multilabel classification: A least-squares formulation, extensions, and analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence **33**(1), 194–200 (2010)
20. Tai, F., Lin, H.T.: Multilabel classification with principal label space transformation. Neural Computation **24**(9), 2508–2542 (2012)
21. Tsoumakas, G., Katakis, I., Vlahavas, I.: Effective and efficient multilabel classification in domains with large number of labels. In: Proc. ECML-PKDD 2008 Workshop on Mining Multidimensional Data. pp. 53–59 (2008)
22. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining multi-label data. In: Data Mining and Knowledge Discovery Handbook, pp. 667–685. Springer (2010)
23. Tsoumakas, G., Vlahavas, I.: Random k-labelsets: An ensemble method for multi-label classification. In: Proc. European Conference on Machine Learning (ECML). pp. 406–417 (2007)
24. Zhang, M.L., Zhou, Z.H.: A review on multi-label learning algorithms. IEEE Transactions on Knowledge and Data Engineering **26**(8), 1819–1837 (2014)
25. Zhang, Z., Jung, C.: GBDT-MO: Gradient-boosted decision trees for multiple outputs. IEEE Transactions on Neural Networks and Learning Systems (2020)
26. Zhou, T., Tao, D., Wu, X.: Compressed labeling on distilled labelsets for multi-label learning. Machine Learning **88**(1-2), 69–126 (2012)