# Does Tail Label Help for Large-Scale Multi-Label Learning

**Tong Wei** and **Yu-Feng Li**[*]
National Key Laboratory for Novel Software Technology, Nanjing University
Collaborative Innovation Center of Novel Software Technology and Industrialization
Nanjing 210023, China
{weit, liyf}@lamda.nju.edu.cn

## Abstract

Large-scale multi-label learning annotates relevant labels for unseen data from a huge number of candidate labels. It is well known that in large-scale multi-label learning, labels exhibit a long tail distribution in which a significant fraction of labels are tail labels. Nonetheless, how tail labels make impact on the performance metrics in large-scale multi-label learning was not explicitly quantified. In this paper, we disclose that whatever labels are randomly missing or misclassified, tail labels impact much less than common labels in terms of commonly used performance metrics (Top-$k$ precision and nDCG@$k$). With the observation above, we develop a low-complexity large-scale multi-label learning algorithm with the goal of facilitating fast prediction and compact models by trimming tail labels adaptively. Experiments clearly verify that both the prediction time and the model size are significantly reduced without sacrificing much predictive performance for state-of-the-art approaches.

## 1 Introduction

Large-scale multi-label learning [Zhang and Zhou, 2014; Hsu *et al.*, 2009] annotates data object with the relevant labels from an extremely large number of candidate labels, which recently owns many real applications. For example, in webpage categorization, millions of labels (categories) are collected in Wikipedia and one needs to annotate a new webpage with relevant labels from such a big candidate set; in image annotation, millions of people tags are in the repository and one wishes to tag each individual picture from such a big candidate tags; in recommendation system, millions of items are presented and one hopes to make informative personalized recommendation from the big candidate items.

An important statistical characteristic of large-scale multi-label learning is that labels follow a power law distribution (as illustrated in Figure 1). There are more than 70% labels which occur in at most 15 examples on each dataset. The infrequently occurring labels are referred as tail labels and the frequently occurring ones are referred as common labels.

How do tail labels impact the performance? It turns out that this intrinsic issue is persistently neglected in most large-scale multi-label learning studies, though tail label has recently attracted increasing attention [Babbar and Schölkopf, 2018]. Most approaches usually believe that the final performance would benefit from leveraging tail labels [Bhatia *et al.*, 2015; Xu *et al.*, 2016; Jain *et al.*, 2016].

To answer this question, in this paper we compute the impact of tail labels on popular performance metrics through analyzing the missing labels and the misclassified labels. Our analyses consistently show that tail labels impact much less than common labels in terms of commonly used performance metrics (Top-$k$ precision and nDCG@$k$). This implies that simply optimizing the Top-$k$ precision and nDCG@$k$ metrics in large-scale multi-label learning does not need to take tail labels into account. We conduct a simple empirical study by trimming off 50% labels with fewest positive examples. As illustrated in Figure 2, both the prediction time and the model size are reduced without sacrificing much performance.

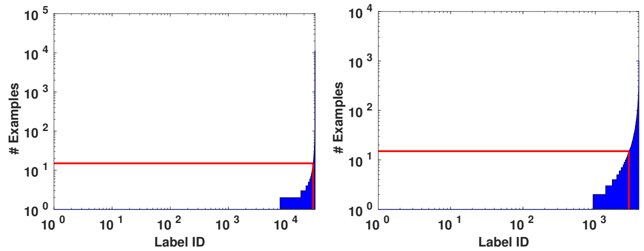
Based on the observations mentioned above, we develop a low-complexity large-scale multi-label learning algorithm with the goal of facilitating fast prediction and compact models through trimming tail labels adaptively. We simulate the training procedure and correlate the reduction in terms of predictive performance, prediction time and model size with the fraction of trimmed tail labels by polynomial functions of label set size. Through modelling such correlations, the objective function is optimized based on golden section search and parabolic interpolation [Forsythe *et al.*, 1977; Brent, 2013]. Experiments verify the effectiveness of trimming tail labels in terms of prediction time and model size reduction, and promising predictive performance. In addition, the strategy mentioned above is a wrap of model and hence can be applicable to many large-scale multi-label models.

In the rest of the paper, we first briefly introduce related works. Then, we study the usefulness of tail labels in large-scale multi-label learning, and propose to trim off tail label adaptively to facilitate fast prediction and compact models. Experimental results on a number of data sets are conducted to verify our idea, followed by discussion and conclusion.

## 2 Related Work

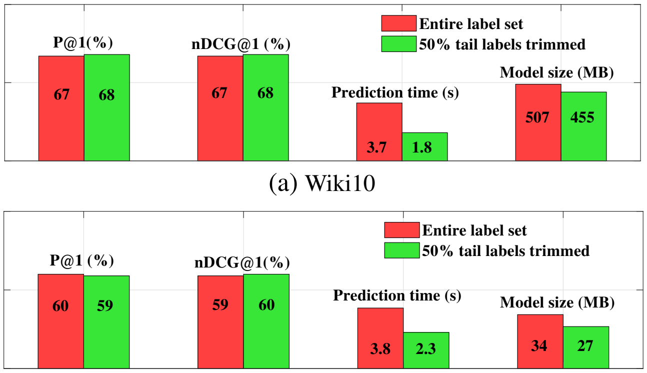
In previous large-scale multi-label learning, prediction time and model size are two crucial issues because fast prediction

[*] Yu-Feng Li is the corresponding author.



(a) Wiki10 (b) EUR-Lex

Figure 1: The number of examples for each label is presented on (a) Wiki10 and (b) EUR-Lex datasets. The horizontal axis indicates the indices of labels, while the vertical axis indicates the number of associated examples in the training data. The vertical red line indicates that labels to the left of it (more than 70%) occur in at most 15 examples on each dataset.



(a) Wiki10

(b) EUR-Lex

Figure 2: Performance of state-of-the-art large-scale multi-label method (LEML [Yu *et al.*, 2014]) on (a) Wiki10 and (b) EUR-Lex datasets with entire label set and label set after trimming off 50% tail labels. As we can see, P@1 and nDCG@1 do not deteriorate while prediction time and model size are reduced.

and compact models are desired for large label set, especially for real-time applications in portable devices.

Many approaches exploit structural assumptions to reduce prediction time. For example, embedding-based methods have been proposed to project label vectors onto a low dimensional space based on the assumption that label matrix is low-rank [Hsu *et al.*, 2009; Zhang and Schneider, 2011; Chen and Lin, 2012; Tai and Lin, 2012; Cisse *et al.*, 2013; Bi and Kwok, 2013; Lin *et al.*, 2014; Yeh *et al.*, 2017]. Another recent thread of research is tree-based methods that recursively divide the space of labels or features to achieve fast prediction speed [Prabhu and Varma, 2014; Choromanska and Langford, 2015; Daume III *et al.*, 2016].

Several methods are proposed recently to yield sparse solutions explicitly or prune spurious weights to achieve sparse models. Yen *et al.* [2016] maximized the margin loss with $\ell_1$ penalty and yielded extremely sparse solution without sacrificing the expressive power of predictor. Babbar and Schölkopf [2017] proposed an framework which controls the model size by filtering out the billions of stored spurious parameters by ad-hoc usage of the off-the-shelf solvers. By weeding out ambiguous parameters, one can obtain model sizes which are three orders of magnitude smaller.

| Notation | Meaning |
|---|---|
| $N$ | Number of training instances |
| $L$ | Number of labels |
| $d$ | Number of feature dimensionality |
| $L_c$ | Number of common labels |
| $L_t = L - L_c$ | Number of tail labels |
| $\mathbf{Y} = [\mathbf{y}_1; \ldots; \mathbf{y}_N]$ | Observed or predictive label matrix |
| $\mathbf{Y}^*$ | Ground-truth label matrix |

Table 1: Summary of Notation

Recently, there are some discussions on the power law distribution in multi-label learning. Bhatia *et al.* [2015] represented tail labels by learning a embedding, which captures non-linear label correlations by preserving the pairwise distances between label vectors. For prediction, a $k$-nearest neighbor classifier is used in the embedding space, leveraging the preserved nearest neighbors in the training phrase. Jain *et al.* [2016] explained that infrequently occurring tail labels are harder to predict than frequently occurring ones since they have little training examples. Xu *et al.* [2016] treated tail labels as outliers and decomposed the label matrix into a low-rank matrix which depicts label correlations and a sparse one capturing the influence of tail labels. Wang *et al.* [2017] cast the tail label problem as transfer learning by transferring knowledge from the data-rich head to the data-poor tail classes. Li *et al.* [2017] handled the long-tail recommendation problem. They decomposed the recommendations into two part, a low-rank part to address short-head items and a sparse part to handle long-tail items. Due to the scarcity of positive training instances for tail labels, Babbar and Schölkopf [2018] viewed this phenomenon as a setup in which an adversary is generating test examples such that the features of the test set instances is quite different from those in the training set. Most of these studies, believe that the final performance would benefit from leveraging tail labels.

## 3 Usefulness of Tail Labels in Large-Scale Multi-Label Learning

In this section, focusing on large-scale multi-label learning, we study the usefulness of tail labels for the final performance. We first briefly introduce the setup of large-scale multi-label learning and its commonly used evaluation metrics. Then, we compute the impact of tail labels on popular performance metrics through analyzing the missing and the misclassification of labels, respectively.

### 3.1 Preliminaries

Let $D = \{(\mathbf{x}_1, \mathbf{y}_1) \ldots (\mathbf{x}_N, \mathbf{y}_N)\}$ be the given training set, where $\mathbf{x}_i \in \mathbb{R}^d$ is the input feature of the $i$-th example and $\mathbf{y}_i \in \{0, 1\}^L$ is the corresponding label vector. $\mathbf{Y}_{ij} = 1$ if example $\mathbf{x}_i$ is relevant with the $j$-th label, and 0 otherwise. Large-scale multi-label learning aims to learn a classifier $f : \mathbb{R}^d \to \{0, 1\}^L$ that predicts the label vector for unseen data. Unlike traditional multi-label learning, the label set size is very large and the labels usually follows a long tail distribution. Detail notations are summarized in Table 1.

## 3.2 Commonly Used Performance Metrics

**Top-$k$ precision**

Top-$k$ precision is a commonly used ranking based performance measure in large-scale multi-label learning and has been widely adopted for ranking tasks [Prabhu and Varma, 2014; Bhatia *et al.*, 2015]. In Top-$k$ precision, only a few top predictions of an instance will be considered. For each instance $\mathbf{x}$, the Top-$k$ precision is defined for a predicted score vector $\hat{\mathbf{y}} \in \mathbb{R}^L$ and ground truth label vector $\mathbf{y} \in \{-1, 1\}^L$ as

$$\text{P@}k := \frac{1}{k} \sum_{l \in \text{rank}_k(\hat{\mathbf{y}})} \mathbf{y}_l, \tag{1}$$

where $\text{rank}_k(\hat{\mathbf{y}})$ returns the indices of $k$ largest value in $\hat{\mathbf{y}}$ ranked in descending order.

**nDCG@$k$**

nDCG@$k$ is another commonly used ranking based performance measure and is defined as

$$\text{nDCG@}k := \frac{\text{DCG@}k}{\sum_{l=1}^{\min(k, \|\mathbf{y}\|_0)} \frac{1}{\log(l+1)}}, \tag{2}$$

where $\text{DCG@}k := \sum_{l \in \text{rank}_k(\hat{\mathbf{y}})} \frac{\mathbf{y}_l}{\log(l+1)}$ and $||\mathbf{y}||_0$ returns the 0-norm of the true-label vector.

## 3.3 For Labels are Randomly Missing

Due to the large number of labels, human annotators may not be able to go through every label and make out all the relevant labels. We first consider the impact of tail labels under the scenario that *relevant labels go missing randomly with probability* $\epsilon$ [Lim *et al.*, 2015], and compute the expectation of performance for discussion.

**Proposition 1.** *Under the assumption that relevant labels are randomly missing with probability $\epsilon$, common labels have more impact than tail labels in terms of P@$k$ and nDCG@$k$.*

*Proof.* (a) P@$k$: In this analysis, for simplicity we suppose that each instance associates with a constant $c$ number of labels in the ground-truth label vector, that is, $v = (1 - \epsilon)c$ relevant labels are observed for each instance. As only $k$ out of $v$ relevant labels are considered in the calculation of P@$k$, we choose a random subset of size $k$ from $v$ relevant labels which has $\binom{v}{k}$ distinct ways for each instance and compute the expected number of times for the $j$-th label is chosen as

$$\sum_{i=1 \wedge \mathbf{Y}_{ij}=1}^{N} \frac{\binom{v-1}{k-1}}{\binom{v}{k}} = (1 - \epsilon) u_j \frac{k}{v}, \tag{3}$$

As we can see, the value of Eq. (3) increases as $u_j$ becomes larger. Since Eq. (3) only depends on $u_j$ for the $j$-th label, we can compute the contribution of each label separately. By considering the contribution to P@$k$ of tail labels (referred as $T_{\text{P@}k}$) and common labels (referred as $C_{\text{P@}k}$) respectively, we have $T_{\text{P@}k} = b \sum_{j=1}^{L_t} u_j$ and $C_{\text{P@}k} = b \sum_{j=L_t+1}^{L} u_j$, where $b = (1 - \epsilon)\frac{k}{v}$ and labels are sorted according to their occurrences in ascending order beforehand with tail labels at the front.

Then, we get

$$\frac{C_{\text{P@}k}}{T_{\text{P@}k}} = \frac{\sum_{j=L_t+1}^{L} u_j}{\sum_{j=1}^{L_t} u_j} \tag{4}$$

In practical cases, tail labels only occur with a handful of examples while there can be as many as hundreds or even thousands of examples having common labels (as shown in Figure 1). Therefore, $C_{\text{P@}k}$ is significantly larger than $T_{\text{P@}k}$, which evidently discloses that common labels impact much more importantly than tail labels in terms of Top-$k$ precision.

(b) nDCG@$k$: Note that every observed label has the same rank, hence $\sum_{l=1}^{k} \frac{1}{\log(l+1)}$ is a constant. The analysis for nDCG@$k$ is reduced to the one in P@$k$, and we get the same conclusive remark in terms of nDCG@$k$, i.e., common labels impact much more importantly than tail labels.

□

## 3.4 For Labels are Randomly Misclassified

It's a common practice to consider the probability of misclassification [Schapire, 1990; Bartlett, 1998]. We further consider the impact of tail labels under the scenario that *labels are randomly misclassified with probability* $\epsilon$, and similar to Section 3.3, we compute the expectation of performance for discussion.

**Proposition 2.** *Under the assumption that labels are randomly misclassified with probability $\epsilon$, common labels have more impact on P@$k$ and nDCG@$k$ than tail labels.*

*Proof.* (a) P@$k$: Similar to the proof of the previous proposition, we suppose that each instance associates with a constant $c$ number of labels in the ground-truth label vector. Hence, there will be $v = (1 - \epsilon)c + \epsilon(L - c)$ relevant labels in the predicted label vector. By choosing a random subset of size $k$ from $v$ relevant labels, the expected number of times the $j$-th label is chosen can be computed as

$$\sum_{i=1 \wedge \mathbf{Y}_{ij}=1}^{N} \frac{\binom{v-1}{k-1}}{\binom{v}{k}} = \Big(\epsilon N + (1 - 2\epsilon) u_j\Big) \frac{k}{v} \tag{5}$$

Since Eq. (5) only depends on $u_j$ for the $j$-th label, we can compute the contribution of each label separately. By considering contribution of tail labels and common labels respectively, we have $T_{\text{P@}k} = b \sum_{j=1}^{L_t} u_j + \epsilon N L_t \frac{k}{v}$ and $C_{\text{P@}k} = b \sum_{j=L_t+1}^{L} u_j + \epsilon N (L - L_t) \frac{k}{v}$, where $b = (1 - 2\epsilon)\frac{k}{v}$. Therefore, we have

$$\frac{C_{\text{P@}k}}{T_{\text{P@}k}} = \frac{\epsilon N (L - L_t) + (1 - 2\epsilon) \sum_{j=L_t+1}^{L} u_j}{\epsilon N L_t + (1 - 2\epsilon) \sum_{j=1}^{L_t} u_j} \tag{6}$$

As tail labels only occur in a handful of examples and common labels can associate with as many as hundreds or even thousands of examples in practical cases, similarly, disclosing that common labels impact much more importantly than tail labels in terms of Top-$k$ precision.

(b) nDCG@$k$: Similar with the reason in the proposition 1, The analysis for nDCG@$k$ is reduced to the one in P@$k$, and we obtain a same conclusive remark in terms of nDCG@$k$.

□

With the analysis above, in both label-missing and label-misclassified scenarios, we conclude that common labels have a significant larger impact compared with that of tail labels in terms of P@$k$ and nDCG@$k$. This analysis motivates us to trim off tail labels, which may have little impact on popular performance metrics.

## 4 Adaptively Trimming off Tail Labels

A straightforward way to trim off tail labels is to remove a constant proportion of labels. However, due to variety of data, it may result in performance deterioration in case too many tail labels are pruned. Therefore, a crucial issue is to adaptively select a cut-off threshold for tail labels, taking the predictive performance, prediction time and model size into account simultaneously.

In this section, we present the proposed Adaptively Trimming off Tail Labels (ADATTL) method which selects a threshold $\lambda$ adaptively on a variety of data where $\lambda$ represents a fraction of tail labels to be trimmed, to trade off the predictive performance, prediction time and model size. Since it is hard to foresee the influence of different value of $\lambda$, we consider to build a regressor using different label set size and different training set size, to predict the potential reduction of predictive performance, prediction time and model size, respectively, and thus determine an appropriate threshold.

Our approach is data-driven and achieved by sampling technique. We start collecting data for regressors by trimming off a random fraction $\lambda_\tau$ of tail labels resulting in the remaining label set $L_\tau$ and sampling a subset of training examples $D_\tau$ uniformly at random. Then, we train an large-scale multi-label classifier $f_\tau$ on $(D_\tau, L_\tau)$ and compute P@$k$, nDCG@$k$, prediction time and model size. This process is repeated multiple times. We fit the obtained set of triplets $\{(|D_{\tau j}|, \lambda_{\tau j}, \mathit{perf}_j)\}_{j=1}^T$, $\{(|D_{\tau j}|, \lambda_{\tau j}, \mathit{time}_j)\}_{j=1}^T$ and $\{(|D_{\tau j}|, \lambda_{\tau j}, \mathit{size}_j)\}_{j=1}^T$ with polynomial functions, where $T$ is sample size, $\mathbf{perf}$ is the testing performance vector, $\mathbf{time}$ is the prediction time vector and $\mathbf{size}$ is the model size vector calculated during the simulation. From this, regression functions can predict the potential reduction in testing performance, prediction time and model size given the training set size and $\lambda$. After building the regressors, we formalize ADATTL as

$$\max \mathcal{L} = \max_{0<\lambda<1} f(N_{tr}, \lambda) - \alpha g(N_{tr}, \lambda) - \beta h(N_{tr}, \lambda), \quad (7)$$

where $\alpha$, $\beta$ are trade-off parameters, $N_{tr}$ is the training set size and functions $f, g, h$ are regressors for testing performance, prediction time and model size, respectively. To maximize Eq. (7), functions $g$ and $h$ are desired to be as small as possible which indicates a large reduction in prediction time and model size respectively, whereas function $f$ is preferred to be as large as possible which means losing less testing performance. We obtain the final threshold $\lambda$ by optimizing $\mathcal{L}$ based on golden section search and parabolic interpolation [Forsythe *et al.*, 1977; Brent, 2013]. The steps discussed above are summarized in Algorithm 1.

**Algorithm 1** ADATTL

```
Input: feature vectors $\mathbf{X} \in \mathbb{R}^{N \times d}$; label vectors $\mathbf{Y} \in \mathbb{R}^{N \times L}$; hyper-parameters $\alpha, \beta$ and sample size $T$
Output: the fraction $\lambda$ of tail labels
for $t = 1, 2, \cdots, T$ do
    trim a randomly selected fraction $\lambda_{\tau t}$ of tail labels resulting in the remaining label set $L_{\tau t}$
    sample a subset of training examples $D_{\tau t}$ randomly
    train large-scale multi-label model $f_t$ on ($D_{\tau t}$, $L_{\tau t}$)
    compute $\mathit{perf}_t$, $\mathit{time}_t$ and $\mathit{size}_t$
end for
fit $\{(|D_{\tau j}|, \lambda_{\tau j}, \mathit{perf}_j)\}_{j=1}^T$, $\{(|D_{\tau j}|, \lambda_{\tau j}, \mathit{time}_j)\}_{j=1}^T$ and $\{(|D_{\tau j}|, \lambda_{\tau j}, \mathit{size}_j)\}_{j=1}^T$ with polynomial surfaces
optimize Eq. (7) and obtain $\lambda$
return $\lambda$
```

## 5 Experiments

We conduct experiments with a leading embedding-based method LEML [Yu *et al.*, 2014] and a state-of-the-art tree-based method FastXML [Prabhu and Varma, 2014] to validate our theoretical findings and effectiveness of ADATTL. Experiments are carried out on multi-label datasets including Bibtex (159 labels), Delicious (983 labels), EUR-Lex (3993 labels) and Wiki10 (30K labels). All the datasets and implementation of LEML and FastXML are publicly available and can be downloaded from the Extreme Classification Repository[1]. To demonstrate that ADATTL selects threshold properly, we trim tail labels with varying fractions ranging from [10%, 20%, . . . , 90%] for comparison. As the fraction of trimmed tail labels increases at each time, we decrease the number of trees trained in FastXML by two. We use quartic polynomial functions to model functions $f$, $g$ and $h$. When modelling function $f$, we use the sum of P@$k$ and nDCG@$k$, $k = \{1, 2, 3\}$, as the whole testing performance. Default value of parameters for LEML and FastXML are used and hyper-parameters $\alpha$ and $\beta$ in Eq. (7) are set to 1.

### 5.1 For Embedding-Based Methods

We compare the performance of ADATTL with LEML. The classification performance in terms of P@$k$ and nDCG@$k$ is presented in Figure 3, while Table 2 presents the results of prediction time and model size. It can be seen that ADATTL selects threshold properly which do not result in performance deterioration, while stably performs better than LEML on all datasets in terms of prediction time and model size. For example, ADATTL improves over LEML by as much as 92.1% and 20% in terms of prediction time and model size on Wiki10 dataset. This is because that the success of LEML mainly depends on the low-rank assumption, which tends to be violated due to the presence of tail

[1] http://manikvarma.org/downloads/XC/XMLRepository.html

labels. Therefore, tail labels make very limited contribution to the performance of LEML. ADATTL provides an appropriate approach to preserve the validity of low-rank assumption by elegantly trimming off the tail labels. Hence, ADATTL is able to achieve comparable results.

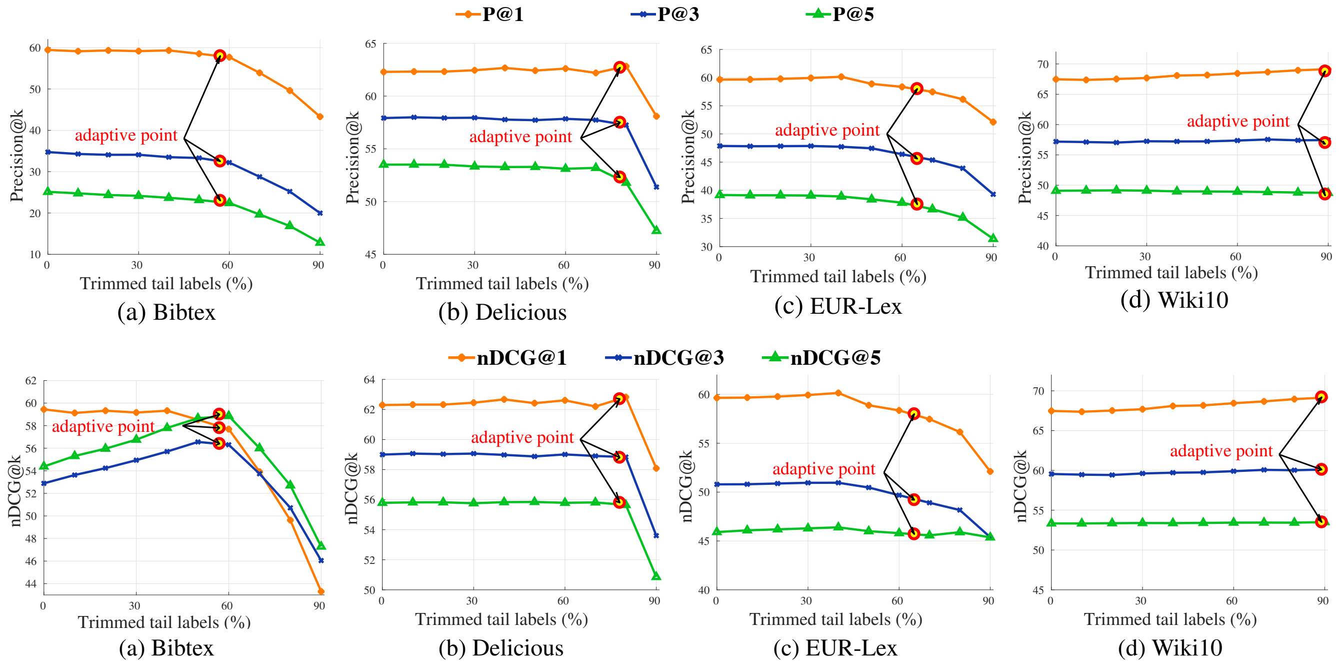
## 5.2 For Tree-Based Methods

We next conduct experiments with FastXML, which is a leading tree-based method. Figure 4 and Table 3 show the comparison results and we have the following observations. Compared with FastXML, ADATTL achieves highly competitive performance in terms of P@$k$ and nDCG@$k$, meanwhile saves considerable prediction time and model size in all cases. The reason lies in the fact that, the number of split partitions and the depth of trees during the training process are both reduced as the number of labels decreases, therefore the size of tree model and the prediction time spent on leaf nodes are cut down consequently.

From the experimental results, we conclude that the proposed strategy selects threshold adaptively and yield fast prediction speed as well as compact models. Moreover, the effectiveness is valid for both embedding-based and tree-based approaches, which validates the applicability of our strategy.

## 6 Discussion

Our analysis and empirical studies suggest that, in order to evaluate the performance of large-scale multi-label methods on tail labels, the choice of performance metric is critical due to the power-law distribution. Jain *et al.* [2016] claimed that existing performance metrics, such as the Hamming loss, are unsuitable for performance evaluation. They developed propensity scored variants of top-$k$ precision, nDCG@$k$ and other popular performance metrics which treat tail labels as



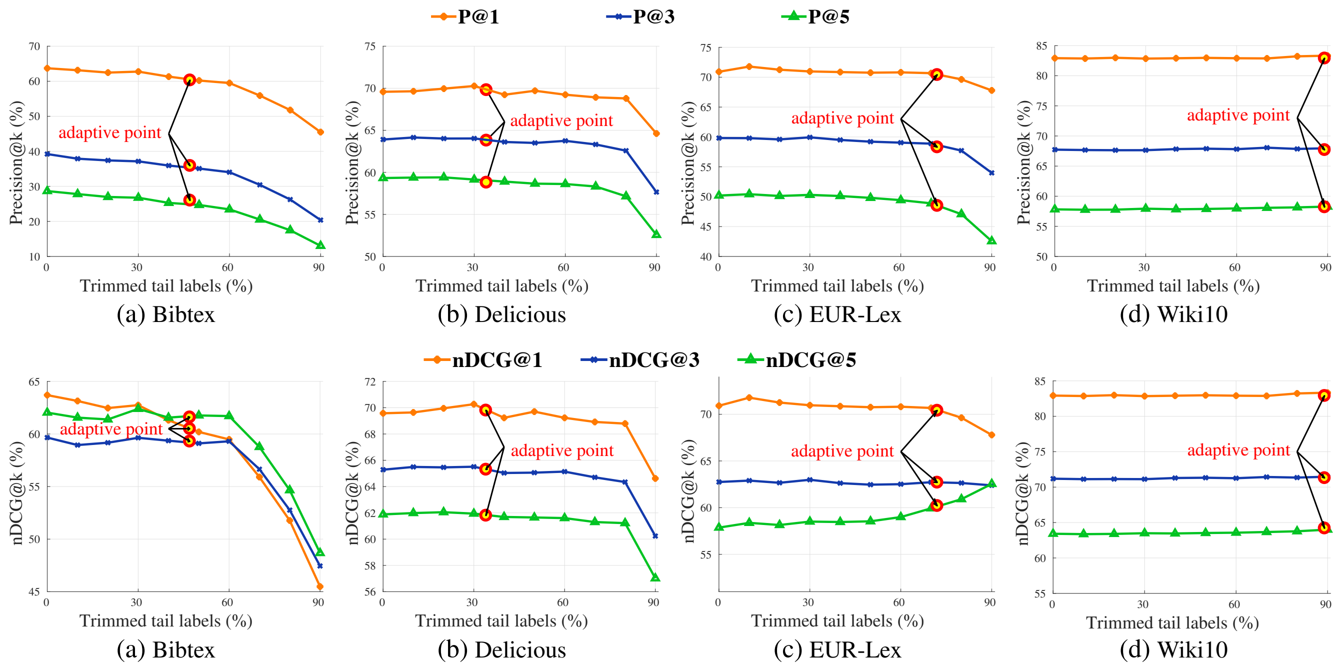
(a) Bibtex (b) Delicious (c) EUR-Lex (d) Wiki10

(a) Bibtex (b) Delicious (c) EUR-Lex (d) Wiki10

Figure 3: Top@$k$ precision and nDCG@$k$ of state-of-the-art embedding-based method LEML with different ratios of trimmed tail labels, where "adaptive point" refers to the adaptive threshold selected by our ADATTL method.

| Dataset | | LEML | ADATTL | Reduction over LEML |
|---|---|---|---|---|
| Bibtex | Prediction time | 0.31 s | 0.26 s | 16.13 % |
| | Model size | 0.76 MB | 0.61 MB | 19.74 % |
| Delicious | Prediction time | 0.02 s | 0.01 s | 50.00 % |
| | Model size | 2.26 MB | 1.13 MB | 50.00 % |
| EUR-Lex | Prediction time | 3.85 s | 1.85 s | 51.92 % |
| | Model size | 34.31 MB | 24.22 MB | 29.41 % |
| Wiki10 | Prediction time | 3.67 s | 0.29 s | 92.10 % |
| | Model size | 506.88 MB | 405.52 MB | 20.00 % |

Table 2: Prediction Time (s) and Model Size (MB) with comparison embedding-based method LEML.



(a) Bibtex (b) Delicious (c) EUR-Lex (d) Wiki10

(a) Bibtex (b) Delicious (c) EUR-Lex (d) Wiki10

Figure 4: Top@$k$ precision and nDCG@$k$ of state-of-the-art tree-based method FastXML with different ratios of trimmed tail labels, where "adaptive point" refers to the adaptive threshold selected by our AdaTTL method.

| Dataset | | FastXML | AdaTTL | Reduction over FastXML |
|---|---|---|---|---|
| Bibtex | Prediction time | 1.47 s | 1.12 s | 23.80 % |
| | Model size | 18.72 MB | 15.10 MB | 19.34 % |
| Delicious | Prediction time | 5.82 s | 4.12 s | 29.21 % |
| | Model size | 71.29 MB | 53.28 MB | 25.26 % |
| EUR-Lex | Prediction time | 14.22 s | 8.05 s | 43.34 % |
| | Model size | 194.40 MB | 130.09 MB | 33.08 % |
| Wiki10 | Prediction time | 45.74 s | 25.10 s | 45.12 % |
| | Model size | 501.47 MB | 301.42 MB | 39.89 % |

Table 3: Prediction Time (s) and Model Size (MB) with comparison tree-based method FastXML.

being more important than common ones. Nonetheless, the weights for labels were set in an ad hoc fashion and unavailable for general cases. To make tail labels have great merit, more preferable performance metrics need to be designed.

## 7 Conclusion

In this paper, we propose to study how tail labels make impact on the commonly used performance metrics in large-scale multi-label learning. Through examining the missing labels and the misclassified labels, our analysis discloses that tail labels consistently impact much less than common labels on popular performance metrics. We then develop a low-complexity large-scale multi-label algorithm to facilitate fast prediction and compact models by trimming tail labels adaptively. Experiments verify the effectiveness of trimming tail labels in terms of prediction time and model size reduction, and promising predictive performance. The contribution of this work is that we provide a different aspect for large-scale multi-label learning, revealing that significant attention should be paid to the design of performance metrics, to fully exploit the great merit of tail labels.

## Acknowledgments

This research was supported by the National Key R&D Program of China (2018YFB1004301), the National Natural Science Foundation of China (61772262) and the Fundamental Research Funds for the Central Universities (020214380044).

## References

[Babbar and Schölkopf, 2017] R. Babbar and B. Schölkopf. Dismec: Distributed sparse machines for extreme multi-label classification. In *Proceedings of the 10th ACM In-*

*ternational Conference on Web Search and Data Mining*, pages 721–729, Cambridge, UK, 2017.

[Babbar and Schölkopf, 2018] R. Babbar and B. Schölkopf. Adversarial extreme multi-label classification. *arXiv preprint arXiv:1803.01570*, 2018.

[Bartlett, 1998] P. L. Bartlett. The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. *IEEE Transactions on Information Theory*, 44(2):525–536, 1998.

[Bhatia *et al.*, 2015] K. Bhatia, H. Jain, P. Kar, M. Varma, and P. Jain. Sparse local embeddings for extreme multi-label classification. In *Advances in Neural Information Processing Systems*, pages 730–738, Montreal, Canada, 2015.

[Bi and Kwok, 2013] W. Bi and J. T. Kwok. Efficient multi-label classification with many labels. In *Proceedings of the 30th International Conference on Machine Learning*, pages 405–413, Atlanta, GA, 2013.

[Brent, 2013] R. P. Brent. *Algorithms for minimization without derivatives*. Courier Corporation, 2013.

[Chen and Lin, 2012] Y.-N. Chen and H.-T. Lin. Feature-aware label space dimension reduction for multi-label classification. In *Advances in Neural Information Processing Systems*, pages 1529–1537, Lake Tahoe, NV, 2012.

[Choromanska and Langford, 2015] A. E. Choromanska and J. Langford. Logarithmic time online multiclass prediction. In *Advances in Neural Information Processing Systems*, pages 55–63, Montreal, Canada, 2015.

[Cisse *et al.*, 2013] M. M. Cisse, N. Usunier, T. Artieres, and P. Gallinari. Robust bloom filters for large multilabel classification tasks. In *Advances in Neural Information Processing Systems*, pages 1851–1859, Lake Tahoe, NV, 2013.

[Daume III *et al.*, 2016] H. Daume III, N. Karampatziakis, J. Langford, and P. Mineiro. Logarithmic time one-against-some. *arXiv preprint arXiv:1606.04988*, 2016.

[Forsythe *et al.*, 1977] G. E. Forsythe, C. B. Moler, and M. A. Malcolm. Computer methods for mathematical computations. 1977.

[Hsu *et al.*, 2009] D. J. Hsu, S. M. Kakade, J. Langford, and T. Zhang. Multi-label prediction via compressed sensing. In *Advances in Neural Information Processing Systems*, pages 772–780, Vancouver, Canada, 2009.

[Jain *et al.*, 2016] H. Jain, Y. Prabhu, and M. Varma. Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 935–944, San Francisco, CA, 2016.

[Li *et al.*, 2017] J.-J. Li, K. Lu, Z. Huang, and H.-T. Shen. Two birds one stone: On both cold-start and long-tail recommendation. In *Proceedings of the 2017 ACM on Multimedia Conference*, pages 898–906. ACM, 2017.

[Lim *et al.*, 2015] D. Lim, J. McAuley, and G. Lanckriet. Top-n recommendation with missing implicit feedback. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 309–312. ACM, 2015.

[Lin *et al.*, 2014] Z.-J. Lin, G.-G. Ding, M.-Q. Hu, and J.-M. Wang. Multi-label classification via feature-aware implicit label space encoding. In *Proceedings of the 31st International Conference on Machine Learning*, pages 325–333, Beijing, China, 2014.

[Prabhu and Varma, 2014] Y. Prabhu and M. Varma. Fastxml: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 263–272, New York City, NY, 2014.

[Schapire, 1990] R. E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.

[Tai and Lin, 2012] F. Tai and H.-T. Lin. Multilabel classification with principal label space transformation. *Neural Computation*, 24(9):2508–2542, 2012.

[Wang *et al.*, 2017] Y.-X. Wang, D. Ramanan, and M. Hebert. Learning to model the tail. In *Advances in Neural Information Processing Systems*, pages 7032–7042, 2017.

[Xu *et al.*, 2016] C. Xu, D.-C. Tao, and C. Xu. Robust extreme multi-label learning. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1275–1284, San Francisco, CA, 2016.

[Yeh *et al.*, 2017] C.-K. Yeh, W.-C. Wu, W.-J. Ko, and Y.-C. F. Wang. Learning deep latent space for multi-label classification. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 2838–2844, San Francisco, CA, 2017.

[Yen *et al.*, 2016] I. E.-H. Yen, X.-R. Huang, P. Ravikumar, K. Zhong, and I. S. Dhillon. Pd-sparse: A primal and dual sparse approach to extreme multiclass and multilabel classification. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 3069–3077, New York City, NY, 2016.

[Yu *et al.*, 2014] H.-F. Yu, P. Jain, P. Kar, and I. S. Dhillon. Large-scale multi-label learning with missing labels. In *Proceedings of the 31st International Conference on Machine Learning*, pages 593–601, Beijing, China, 2014.

[Zhang and Schneider, 2011] Y. Zhang and J. Schneider. Multi-label output codes using canonical correlation analysis. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, pages 873–882, Ft. Lauderdale, FL, 2011.

[Zhang and Zhou, 2014] M.-L. Zhang and Z.-H. Zhou. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837, 2014.