# Towards Extreme Multi-label Text Classification Through Group-wise Label Ranking

Jie Xiong*, Huiyuan Li*, Qiyuan Duan†, Qihan Du*
*Renmin University of China, Beijing, 100872, China, {xiongjiezk, hyli18, duqihan}@ruc.edu.cn
†Zhihu Inc., {duanqiyuan}@zhihu.com

***Abstract*—Extreme Multi-label text Classification (XMC) aims to find the most relevant labels (i.e., the positives) for a document from an extremely large label set. The remaining labels are regarded as the negatives. Recently, the deep learning-based methods have been widely used to solve XMC, most of which use *sigmoid* as the activation function of output layer and use *binary cross entropy* loss as the learning objective. However, the existing methods suffer from the following limitations. First, the score of each label is predicted independently, where the label rank-missing problem is ignored. Second, the cardinalities of the positives and the negatives are extremely unbalanced in XMC, which makes the classifier more biased towards the majority one. In this paper, we use *label group* to denote the positive and the negative labels and propose a novel XMC model leveraging Group-wise label Ranking (X-GRank) to address those limitations. Specifically, X-GRank uses the newly proposed GRank loss to rank the label groups. Then, X-GRank solves the label imbalance problem by constraining the backward gradient amplitude between label groups. Extensive experiments show that X-GRank outperforms the state-of-the-art methods on five widely used datasets.**

***Index Terms*—Extreme Multi-label Text Classification, Deep Learning, Group-wise Ranking**

## I. Introduction

Text classification is a key task in the field of natural language processing (NLP), which can be divided into multi-class and multi-label classification problems based on whether the classes/labels are mutually exclusive. Extreme Multi-label text Classification (XMC) aims to tag a document with relevant labels from an extremely large label set. XMC becomes increasingly important due to the fast-growing of online contents and the urgent need for the better organization of the messy big data. For example, Wikipedia needs to tag a new article from more than one million labels. Compared with the traditional multi-label classification methods [1], XMC focuses on how to extract uncertain positive labels from numerous candidate labels, which leaves researchers with open challenges.

In this paper, we introduce a concept of *label group* and view text classification from a novel unified perspective. For a given document, its associated labels are regarded as a *positive group*, denoted as $G_p$. Other irrelevant labels are treated as a corresponding *negative group*, denoted as $G_n$. $G_p$ and $G_n$ satisfy $|G_p| + |G_n| = L$, where $|G_p|$ and $|G_n|$ are the cardinalities of $G_p$ and $G_n$, and $L$ is the number of labels of the corpus. In this perspective, multi-class classification means that there is a constraint of $|G_p| = 1$, and multi-label

DOI reference number: 10.18293/SEKE23-028

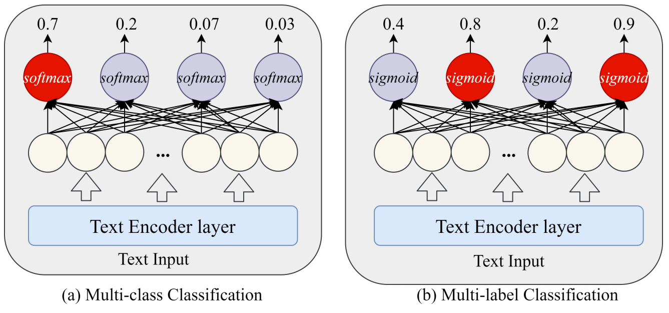

Fig. 1. Comparison of multi-class classification and multi-label classification in current deep learning methods.

classification allows $|G_p| >= 1$. For XMC, there may be an $L$ as large as millions, and $|G_p|$ is far less than $|G_n|$ and $L$.

Nowadays, the superiority of deep learning-based text classification methods has been frequently reported [1]–[5]. Most of the existing models stack a full connection layer on the text encoder to output the scores of each label. For multi-class classification, as shown in Fig. 1 (a), *softmax* and *cross entropy* (CE) loss always act as the activation function of the output layer and the learning objective. The scores of softmax can be viewed as a probability distribution due to the sum of all scores is 1.0, which results in maximizing the positive labels is equivalent to minimizing the negative labels and vice versa. For multi-label classification problem, to adapt $|G_p| >= 1$, *softmax* and CE is replaced by *sigmoid* and *binary cross entropy* (BCE) [1], [3], so that the model can predict all labels independently, as shown in Fig. 1 (b). Sigmoid and BCE loss is the currently most widely used combination in multi-label classification [1], [2]. However, this combination brings the following problems in XMC:

- The ranking learning between labels is missing, but we hope all scores in $G_p$ are greater than those in $G_n$.
- The sparse positive labels lead to serious label imbalance in XMC, which may hinder the model convergence.

To address the problems above, we explore the label ranking learning mechanism and label balancing mechanism for XMC. Specifically, we use $\boldsymbol{g_p}$ and $\boldsymbol{g_n}$ denote the scores of labels in $G_p$ and $G_n$, respectively. First, from the perspective of label ranking, it needs to compare the scores between $\boldsymbol{g_p}$ and $\boldsymbol{g_n}$, and then optimize $Obj_{raw} : g_p^i > g_n^j, i = 1, \cdots, |G_p|, j = 1, \cdots, |G_n|$. However, $Obj_{raw}$ can be hardly used as a loss function due to the non-differentiability and slow convergence. In this paper, we made a series of smooth transformations on

$Obj_{raw}$ and proposed a new efficient differentiable **G**roup-wise label **Rank**ing (GRank) loss function, which offers the comparison between label groups with linear time complexity. Subsequently, the problem of label imbalance can be addressed by GRank loss through the equivalent backward gradients between groups involved.

In addition, due to the huge label set of XMC, it is an inevitable challenge for solving the XMC with the limited computational resources. To reduce the calculation cost, dividing the training process into multiple cascaded stages, usually including a retrieving stage and a ranking stage, has become a prevalent method [2]–[4], [6]. The retrieving stage gathers a shortlist of candidate labels from the original large label set, and the ranking stage boosts the positive labels from retrieved labels. However, most of the existing methods train those stages not in an end-to-end way, such as AttentionXML [2] and X-Transformer [4], which brings unavoidable cascaded errors. In this paper, we designed a new XMC model with GRank loss, named X-GRank, which includes a retrieving task and a ranking task. We trained X-GRank in end-to-end way to eliminate the cascaded errors. Moreover, we used label features to enhance the label representation, which further improved the performance of the X-GRank. We summarize three major contributions of this paper as follows:

- We proposed a novel Group-wise label Ranking loss based on the perspective of positive and negative label groups. GRank loss addressed the rank-missing and label imbalance problems of the existing XMC models.
- We proposed a new XMC model with GRank loss. We trained X-GRank end-to-end to eliminate the cascaded errors in existing methods.
- We conducted extensive experiments on five widely datasets of XMC. The experimental results show our model achieves remarkable results on those datasets.

## II. Related work

We summarized the existing work into traditional machine learning methods and deep learning methods.

The traditional methods can be divided into three classes: one-vs-all, tree-based, and embedding-based methods. One-vs-all methods, such as DiSMEC [7], have good classification results, but with heavy computational overhead. Tree-based methods generate a tree hierarchy through recursive partitioning and the representative models, including Parabel [8] and BonSai [9]. However, the cascaded errors of tree-based methods cannot be ignored. Embedding-based methods attempt to address the data sparsity issue through matrix compression and decompression, which is unfriendly to tail labels [1].

Deep learning-based methods show their strong learning ability in XMC. Many novel network structures have been proposed, including dynamic pooling technology in XML-CNN [1], attention mechanism in AttentionXML [2], negative sampling strategy in LightXML [3], and label structure learning [10]. Those methods continuously refresh the leaderboard of XMC.

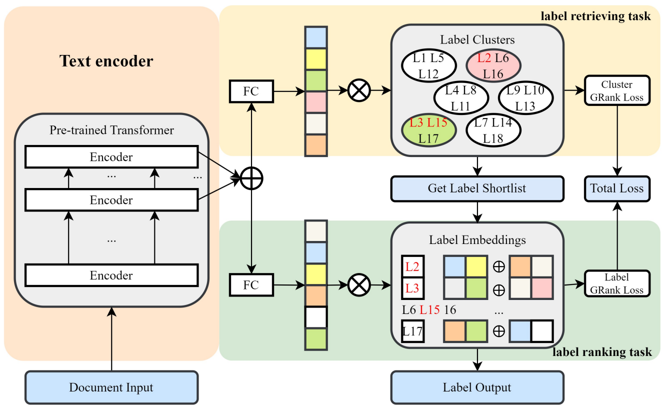

Fig. 2. The structure of X-GRank.

In addition, the problem of tail labels is a great challenge in XMC. Tail labels are referred to the infrequently occurred labels, which have limited training samples and are harder to predict than the frequently occurred ones (referred to as head labels). Many existing XMC approaches [1], [7], [9] treat all labels with equal importance, which may lead XMC to favor head labels. Propensity score [11] is proposed to measure the model performance on tail labels. Label features are important to label learning. Effective utilization of label features in X-Transformer [4], ECLARE [5] significantly improves precision on accessible benchmark datasets.

## III. Proposed method

### A. Research formulation

Given a document corpus, $D = \left\{(d_i : G_p^i, G_n^i)\right\}_{i=1}^{N}$, where $d_i$ is the $i$-th document, $N$ is document size of corpus. $G_p^i$, $G_n^i$ are positive and negative label group of $d_i$. Our goal is to learn a function $f(d_i) \in R^L$, which outputs the correlation scores of all labels for the given document. The function $f$ should be optimized to make the scores of labels in $G_p^i$ to be higher than that in $G_n^i$.

### B. The structure of X-GRank

We proposed a novel XMC model, named X-GRank. It consists of three components, as shown in Fig. 2: 1) a text encoder, 2) a label retriever, 3) a label ranker.

Text encoder aims to get an appropriate vector to represent a document. In this paper, we use pre-trained transformer models such as *bert-base-uncased* [12] as the text encoder. To make full utilize of implicit representations between each transformer layer, we concatenate the hidden states of "[CLS]" token of last five layers as the document representation, which is the same as LightXML [3].

Both label retriever and label ranker use the newly proposed loss function, GRank. So, we will first introduce the rationales of GRank, and then explain the application of GRank.

### C. GRank: Group-wise ranking loss

*1) Design of GRank loss:* Generally, XMC uses the scored top-k labels as the prediction result. Consequently, we hope all positive labels have higher scores than the negative labels, which can be formalized as a loss function:

$$Loss = \sum_{i=1}^{|G_p|} \sum_{j=1}^{|G_n|} max(g_n^j - g_p^i, 0) \tag{1}$$

However, Equation (1) has three disadvantages: 1) It is not globally differentiable, resulting in slow convergence. 2) It is not robust during training, where all label pairs are optimized independently, lacking mutual constraint of gradient amplitudes between them. 3) It has a high computational complexity, which is $O(|G_p| \times |G_n|)$. In particular, XMC has numerous labels, which will produce a huge number of label pairs.

In order to solve the above problems, we note that Equation (1) is equivalent to making the smallest positive label greater than the largest negative label, namely, $min(\boldsymbol{g_p}) > max(\boldsymbol{g_n})$. Therefore, the reformed loss function is:

$$Loss = max(0, max(\boldsymbol{g_n}) + max(-\boldsymbol{g_p})) \tag{2}$$

Due to Equation (2) only includes the $max$ function and is not differentiable, we use the softened smooth maximum function *log-sum-exp* (LSE) [13] to approximate the $max$ function. LSE is defined as:

$$LSE(x_1, \cdots, x_n) = log(exp(x_1) + \cdots + exp(x_n)) \tag{3}$$

In particular, $max(0, x)$ can be approximated as $LSE(0, x) = log(1 + exp(x))$. Therefore, we propose the following smoothing loss based on Equation (2):

$$Loss = log(1 + \sum_{j=1}^{|G_n|} exp(g_n^j) \sum_{i=1}^{|G_p|} exp(-g_p^i)) \tag{4}$$

Equation (4) is essentially designed to optimize the ranking relationship between positive and negative label groups, so we call it **G**roup-wise **Rank**ing (GRank) loss.

*2) Analysis on GRank loss:* GRank loss can effectively solve the disadvantages mentioned in Equation (1). First, GRank loss is differentiable, which uses $LSE(\boldsymbol{x})$ approximates $max(\boldsymbol{x})$ with the limited bounds:

$$max(\boldsymbol{x}) \leq LSE(\boldsymbol{x}) \leq max(\boldsymbol{x}) + log(n) \tag{5}$$

where $n$ is the element size of $\boldsymbol{x}$, and the first inequality is strict unless $n = 1$ and the second inequality is strict unless all arguments are equal.

Second, GRank loss has solved the label imbalance problem of XMC. GRank loss is robust in training and has a balanced optimization for positive and negative groups. We denote $s = \sum_{j=1}^{|G_n|} exp(g_n^j) \sum_{i=1}^{|G_p|} exp(-g_p^i)$, and the derivative of Equation (4) is:

$$\sum_{j=1}^{|G_n|} \frac{\partial Loss}{\partial g_n^j} = \frac{s}{(1+s)}, \sum_{i=1}^{|G_p|} \frac{\partial \text{Loss}}{\partial g_p^i} = \frac{-s}{(1+s)} \tag{6}$$

Obviously, there is a mutual constraint gradient between the two groups, which solves the label imbalance problem and can improve the robustness of the model training.

Third, GRank loss has low computational complexity, which is line time complexity with $O(L)$.

## *D. The retrieving stage*

In this stage, we aim to obtain a shortlist of candidate labels from the original labels to reduce the computational cost in the ranking stage. A common method is to divide the original labels into some clusters, and each label belongs to one cluster. Clusters with positive labels are viewed as positive clusters, and clusters without positive labels are negative clusters. So, the cluster prediction is a small-scale XMC task, as shown in Fig. 2. The labels in the positive clusters will generate the shortlist of candidate labels.

*1) Label clustering:* To handle large-scale label sets, we use a Probabilistic Label Tree (PLT) to partition labels, which is widely used in LightXML [3], Bonsai [9] and AttentionXML [2]. PLT is constructed through a top-down hierarchical clustering until the termination condition is met. Each original label is the leaf node of PLT, and the parent node of the leaf node is the label clustering we need. Since a deeper PLT is more prone to error propagation, we use a two-layer PLT with the same building ways in LightXML and AttentionXML.

*2) Label retrieving task:* The key part of label retrieving task is to predict the positive clusters, which can be formalized as: $D = \{(d_i : G_{cp}^i, G_{cn}^i)\}_{i=1}^N$, where $G_{cp}^i$, $G_{cn}^i$ are the positive and the negative clusters, respectively. We denote $\boldsymbol{g_{cp}}$, $\boldsymbol{g_{cn}}$ as the scores of $G_{cp}$ and $G_{cn}$. So, the GRank loss of retrieving task can be formalized as:

$$Loss_{retr} = log(1 + \sum_{j=1}^{|G_{cn}|} exp(g_{cn}^j) \sum_{i=1}^{|G_{cp}|} exp(-g_{cp}^i)) \tag{7}$$

## *E. The ranking stage*

In this stage, we need to learn the label representation first. Label representation is an appropriate vector in a hyperspace to represent a label, which includes two parts in this paper. First, considering that a label may have textual features, such as title and description, we utilize a text embedding block to embed and average those text features through bag-of-words representation. Then, we add an identity embedding layer for each label, which takes the input of the unique identity number of a label in the label set. The label identity embedding is initialized randomly, and then concatenated with label text embedding as the final label representation. The overall structure of label representation learning is shown in right part of Fig. 2.

The label ranking task is to predict the scores of retrieved labels. We divide the retrieved labels into positive and negative groups, denoted as $G_p^*$ and $G_n^*$, respectively. The label scores, $g_p$ and $g_n$, are performed by the dot product of the document and label representations. We apply GRank loss again in ranking task, namely:

$$Loss_{rank} = log(1 + \sum_{j=1}^{|G_n^*|} exp(g_n^j) \sum_{i=1}^{|G_p^*|} exp(-g_p^i)) \tag{8}$$

## *F. Model training*

The retrieving task produces the candidate labels. The ranking task extracts the positive label group from the candidate labels. Therefore, those two tasks have cascading dependencies. In order to reduce error propagation between tasks, we trained X-GRank through end-to-end learning. The intensity of model learning for those two tasks is different during model training. At the beginning of training, X-GRank should pay more attention to the retrieving task, and the ranking task should be prominent at the end of training. As a result, We designed a dynamic task weighting mechanism with linear schedule, which can be formulated as:

$$Loss = (1-\lambda) * Loss_{retr} + \lambda * Loss_{rank} \tag{9}$$

where $\lambda = 0.5 * e/E$, $e = 1, \cdots, E$, $e$ is the number of current epochs, $E$ is the number of total epochs.

## IV. Experiments

### A. Datasets and experiment setup

We use five widely used XMC datasets[1], including Eurlex-4K, AmazonCat-13K, Wiki10-31K, Wiki-500K and Amazon-670K. Those datasets follow the same processing as X-Transformer[2] [4] and AttentionXML[3] [2]. The detailed summary statistics of each dataset are shown in Table I, where $N_{trn}, N_{tst}$ are the numbers of training and test samples, respectively. $L$ is the number of labels, $\bar{L}$ is the average number of labels per sample, $\hat{L}$ is the average number of samples per label, $L_p$ is the percentage of tail labels (sample number per label $< 5$) in all labels, and $W_{trn}$ and $W_{tst}$ are the average number of words per-training and test sample respectively. $L_f$ denotes whether the dataset has label features.

TABLE I
Detailed statistics of datasets.

| Datasets | $N_{trn}$ | $N_{tst}$ | $L$ | $\bar{L}$ | $\hat{L}$ | $L_p$ | $W_{trn}$ | $W_{tst}$ | $L_f$ |
|---|---|---|---|---|---|---|---|---|---|
| Eurlex-4K | 15,449 | 3,865 | 3,956 | 5.30 | 20.79 | 46.07 | 1248.58 | 1230.40 | Yes |
| AmazonCat-13K | 1,186,239 | 306,782 | 13,330 | 5.04 | 448.57 | 17.95 | 246.61 | 245.98 | Yes |
| Wiki10-31K | 14,146 | 6,616 | 30,938 | 18.64 | 8.52 | 75.84 | 2484.30 | 2425.45 | Yes |
| Wiki-500K | 1,779,881 | 769,421 | 501,070 | 4.75 | 16.86 | 43.42 | 808.66 | 808.56 | Yes |
| Amazon-670K | 490,449 | 153,025 | 670,091 | 5.45 | 3.99 | 76.31 | 247.33 | 241.22 | No |

For each dataset, we use raw texts as input, which are truncated or padded to the maximum length. For datasets with small-scale labels like Eurlex-4K, AmazonCat-13K and Wiki10-31K, the number of label cluster is set to 1 and all labels are participated in ranking. In order to make a fair comparison with LightXML, we adopt the same model integration strategy as LightXML, that is, we use three different pre-trained transformer models, including *bert-base-uncased* [12] *roberta-base* [14] and *xlnet-base-cased* [15], to train three sub-models to integrate. For large-scale datasets like Wiki-500K and Amazon-670K, three different label clusters are used for model training. Our model is implemented in PyTorch DistributedDataParallel[4] module with NCCL as the backend. Our Experiments are conducted on eight NVIDIA Tesla V100 GPUs (32 GB memory each). X-GRank is trained by AdamW

[1]http://manikvarma.org/downloads/XC/XMLRepository.html
[2]https://github.com/OctoberChang/X-Transformer
[3]https://github.com/yourh/AttentionXML
[4]https://pytorch.org/docs/stable/distributed.html

TABLE II
Hyperparameters of all datasets.

| Datasets | $E$ | $B$ | $d$ | $C$ | $K_c$ | $L_t$ |
|---|---|---|---|---|---|---|
| Eurlex-4K | 50 | 32 | 512 | 1 | 3,956 | 512 |
| AmazonCat-13K | 20 | 32 | 512 | 1 | 13,330 | 512 |
| Wiki10-31K | 50 | 32 | 512 | 1 | 30,938 | 512 |
| Wiki-500K | 20 | 128 | 128 | 60 | 32 | 128 |
| Amazon-670K | 50 | 64 | 128 | 80 | 32 | 256 |

[16] optimizer. The initial learning rate is $1E-4$ with a warm-up proportion as 0.1. Other hyperparameters are given in Table II, where $E$ is the number of epochs, $B$ is the batch size, $d$ is the dimension of label embedding, $C$ is cluster size which is the number of labels in a cluster, $K_c$ is the top-k retrieved cluster, $L_t$ is the maximum length of input tokens of transformer model.

### B. Metrics

We employ the widely used Recall ($R@k$) and Precision ($P@k$) [1], [3], [4] as the evaluation metrics in retrieving and ranking stages, respectively. In addition, Propensity Scored Precision ($PSP@k$) [11] is used to examine the performance on tail labels. $R@k$, $P@k$ and $PSP@k$ are defined as:

$$\begin{aligned} R@k &= \frac{\sum_{j \in rank_k(\hat{y})} y_j}{\sum_j y_j} \\ P@k &= \frac{1}{k} \sum_{j \in rank_k(\hat{y})} y_j \\ PSP@k &= \frac{1}{k} \sum_{j \in rank_k(\hat{y})} \frac{y_j}{p_j} \end{aligned} \tag{10}$$

where $y_j \in \{0,1\}^L$ is the true binary vector, $rank_k(\hat{y})$ is the indices of the $k$, and $p_j$ is the propensity score of label $j$.

### C. Experimental results

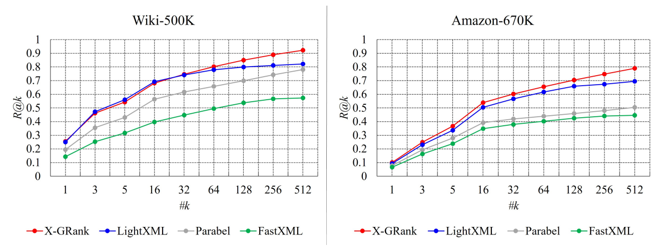

Fig. 3. Changes of recall results with varying $k$ values, where $k$ is the retrieved label size.

*1) Results of the retrieving stage:* In retrieving stage, we evaluate the label recall results on Wiki-500K and Amazon-670K. The comparison methods include LightXML, Parabel [8] and FastXML [17]. LightXML is one of the best deep learning-based method. Parabel and FastXML are classic tree-based methods. As shown in Fig. 3, the recall of X-GRank substantially outperformed Parabel and FastXML. For example, on the Wiki-500K dataset, $R@512$ of Parabel is 59.11%, while $R@512$ of X-GRank achieves 92.18%, with an increase of 33.07%. Although LightXML performed better than Parabel and FastXML, X-GRank outperformed LightXML when the retrieved label size $k$ is greater than 64 on the Wiki-500K and Amazon-670K.

TABLE III
COMPARING X-GRANK AGAINST THE STATE-OF-THE-ART XMC METHODS ON $P@k$

| Datasets | Metrics | DiSMEC | Parabel | Bonsai | XT | XML-CNN | AttentionXML | X-Transformer | LightXML | X-GRank |
|---|---|---|---|---|---|---|---|---|---|---|
| Eurlex-4K | $P@1$ | 83.21 | 82.12 | 82.30 | 79.17 | 75.32 | 87.12 | 87.22 | 87.63 | **88.05** |
| | $P@3$ | 70.39 | 68.91 | 69.55 | 66.80 | 60.14 | 73.99 | 75.12 | 75.89 | **76.67** |
| | $P@5$ | 58.73 | 57.89 | 58.35 | 56.09 | 49.21 | 61.92 | 62.90 | 63.36 | **63.50** |
| AmazonCat-13K | $P@1$ | 93.81 | 92.98 | 92.98 | 92.50 | 93.26 | 95.92 | 96.70 | 96.77 | **97.35** |
| | $P@3$ | 79.08 | 79.14 | 79.13 | 78.12 | 77.06 | 82.41 | 83.85 | 84.02 | **84.08** |
| | $P@5$ | 64.06 | 64.51 | 64.46 | 63.51 | 61.40 | 67.31 | 68.58 | 68.70 | **68.97** |
| Wiki10-31K | $P@1$ | 84.13 | 84.19 | 84.52 | 83.66 | 81.41 | 87.47 | 88.51 | 89.45 | **89.92** |
| | $P@3$ | 74.72 | 72.46 | 73.76 | 73.28 | 66.23 | 78.48 | 78.71 | **78.96** | 78.64 |
| | $P@5$ | 65.94 | 63.37 | 64.69 | 64.51 | 56.11 | 69.37 | 69.62 | **69.85** | 69.67 |
| Wiki-500K | $P@1$ | 70.21 | 68.70 | 69.26 | 65.17 | 59.85 | 76.95 | 77.28 | 77.78 | **78.26** |
| | $P@3$ | 50.57 | 49.57 | 49.80 | 46.32 | 39.28 | 58.42 | 57.47 | 58.85 | **59.54** |
| | $P@5$ | 39.68 | 38.64 | 38.83 | 36.15 | 29.81 | **46.14** | 45.31 | 45.57 | 45.94 |
| Amazon-670K | $P@1$ | 44.78 | 44.91 | 45.58 | 42.54 | 33.41 | 47.58 | - | 49.10 | **49.36** |
| | $P@3$ | 39.72 | 39.77 | 40.39 | 37.93 | 30.00 | 42.61 | - | 43.83 | **44.73** |
| | $P@5$ | 36.17 | 35.98 | 36.60 | 34.63 | 27.42 | 38.92 | - | 39.85 | **40.37** |

*Note: The results of all baselines are cited from Table 3 in LightXML [3].*

*2) Results of the ranking stage:* We compared X-GRank with those state-of-the-art traditional and deep learning-based methods of XMC, including DiSMEC [7], Parabel [8], Bonsai [9], XT [18], XMC-CNN [1], AttentionXML [2], X-Transformer [4], and LightXML [3]. Table III shows the results of $P@k$ on the five datasets. It is worth noting that the result of X-Transformer on Amazon-670K is not reported because of the hardware limits. In order to compare with baselines under the same conditions, we followed the metrics used in baselines to focus on top prediction by varying $k$ at 1, 3, and 5.

Compared with the traditional methods, the deep learning methods have made great progress on XMC. X-GRank, X-Transformer [4] and LightXML [3] are transformer-based approaches using a pre-trained transformer model. LightXML is the current state-of-the-art method of XMC on those five datasets. X-GRank outperform LightXML on all datasets except the metrics $P@3$ and $P@5$ on Wiki10-31K, which demonstrate the effectiveness of our X-GRank.

## D. Ablation analysis

We conducted the following ablation analysis: 1) the comparisons between GRank loss and BCE loss; 2) the contributions of label text features. All those experiments were conducted on the single model, using "*bert-base-uncased*" to initialize the parameters of the text encoder. We conducted the ablation study on Eurlex-4K and Wiki10-31K datasets. The results are shown in Table IV.

- **X-BCE-1 w/o LF** uses the combination of sigmoid and BCE loss without label text features.
- **X-GRank-1 w/o LF** uses GRank loss without label text features.
- **X-GRank-1 w/ LF** uses GRank loss and label text features.

TABLE IV
RESULTS OF THE ABLATION STUDY OF X-GRANK

| Datasets | Metrics | X-BCE-1 w/o LF | X-GRank-1 w/o LF | X-GRank-1 w/ LF |
|---|---|---|---|---|
| Eurlex-4K | $P@1$ | 84.27 | 86.30 | **86.65** |
| | $P@3$ | 71.81 | 73.82 | **74.93** |
| | $P@5$ | 59.59 | 61.99 | **62.11** |
| Wiki10-31K | $P@1$ | 87.71 | 88.63 | **89.63** |
| | $P@3$ | 76.99 | 77.72 | **77.89** |
| | $P@5$ | 67.63 | **68.75** | 68.10 |

TABLE V
COMPARISON OF X-GRANK AGAINST THE BASELINES ON $PSP@k$

| Datasets | Metrics | DiSMEC | Parabel | Bonsai | XML-CNN | Attention XML | X-GRank |
|---|---|---|---|---|---|---|---|
| Eurlex-4K | $PSP@1$ | 38.45 | 37.20 | 37.33 | 32.41 | 44.97 | **45.04** |
| | $PSP@3$ | 46.20 | 44.74 | 45.40 | 36.95 | 51.91 | **53.28** |
| | $PSP@5$ | 50.25 | 49.17 | 49.92 | 39.45 | 54.86 | **57.12** |
| AmazonCat-13K | $PSP@1$ | 51.41 | 50.92 | 51.30 | 52.42 | 53.76 | **55.63** |
| | $PSP@3$ | 61.02 | 64.00 | 64.60 | 62.83 | 68.72 | **69.27** |
| | $PSP@5$ | 65.86 | 72.10 | 72.48 | 67.10 | **76.38** | 75.78 |
| Wiki10-31K | $PSP@1$ | 10.60 | 11.69 | 11.85 | 9.39 | **15.57** | 15.27 |
| | $PSP@3$ | 12.37 | 12.47 | 13.44 | 10.00 | 16.80 | **16.89** |
| | $PSP@5$ | 13.61 | 13.14 | 14.75 | 10.20 | 17.82 | **18.01** |
| Wiki-500K | $PSP@1$ | 27.42 | 26.88 | 27.46 | - | 30.85 | **33.50** |
| | $PSP@3$ | 32.95 | 31.96 | 32.25 | - | 39.23 | **40.73** |
| | $PSP@5$ | 36.95 | 35.26 | 35.48 | - | **44.34** | 42.81 |
| Amazon-670K | $PSP@1$ | 26.26 | 26.36 | 27.08 | 17.43 | 30.29 | **32.21** |
| | $PSP@3$ | 30.14 | 29.95 | 30.79 | 21.66 | 33.85 | **34.60** |
| | $PSP@5$ | 33.89 | 33.17 | 34.11 | 24.42 | 37.13 | **39.02** |

*Note: The results of all baselines are cited from Table 6 in AttentionXML [2].*

*1) Effects of the GRank loss:* X-BCE-1 w/o LF and X-GRank-1 w/o LF use the same input. Because of the contribution of ranking between labels and the avoidance of the imbalance between positive and negative label groups, GRank loss is superior to BCE loss in multi-label classification. The ablation results have strongly verified this conclusion. As shown in Table IV, $P@1$, $P@3$, and $P@5$ of X-GRank-1 w/o LF increased by 2.03%, 2.01% and 2.40% on Eurlex-4K than that of X-BCE-1 w/o LF.

*2) Effects of the label text features:* Label text features are a piece of text that is potentially associated with the content of the document. Specifically, we observed that label text features are always mentioned in document on Eurlex-4K and Wiki10-31K datasets. Therefore, label text features may become a strong signal to characterize the relevance between the given document and a label. As shown in Table IV, compared with X-GRank-1 w/o LF, $P@1$ and $P@3$ of X-GRank-1 w/ LF increased by 0.35% and 1.11% on Eurlex-4K, which verified the positive effects of label features. In addition, when the label text feature is added to X-GRank-1 on Wiki10-31K, $P@1$ increased by 1.0%. Therefore, we added label text features as one of the inputs of X-GRank.

## E. Performance on tail labels

We verified the performance of X-GRank on tail labels. The comparison results are shown in Table V. X-GRank almost completely outperformed AttentionXML on small-scale datasets. In addition, the large-scale datasets, Wiki-500K and Amazon-670K, contains many tail labels. The average sample numbers per label of those datasets are only 16.86 and 3.99, and the tail labels account for 43.42% and 76.31% of all labels, respectively. Compared with AttentionXML, X-GRank has achieved better performance on all metrics except $PSP@5$ on Wiki-500K. The experimental results verified the superiority of effectiveness of X-GRank on tail labels.

### *F. Evaluations of efficiency*

TABLE VI
COMPARISONS OF EFFICIENCY WITH DIFFERENT METHODS

| Datasets | Metrics | Attention XML-1 | Light XML-1 | X-GRank-1 |
|---|---|---|---|---|
| Wiki-500K | Train Time (ms/sample) | 7.5 | **6.9** | 8.71 |
| | Inference Time (ms/sample) | 4.20 | 3.75 | **3.36** |
| | Model Size (GB) | 3.11 | 1.47 | **0.714** |
| Amazon-670K | Train Time (ms/sample) | 12.8 | 14.1 | **8.44** |
| | Inference Time (ms/sample) | 8.59 | 4.09 | **3.12** |
| | Model Size (GB) | 5.52 | 1.53 | **0.797** |

We compare the efficiency of X-GRank with AttentionXML and LightXML of single model, marked with suffix of "-1". The experiment is carried out under the same hardware with one Tesla V100 GPU. The input token lengths of AttentionXML, LightXML and X-GRank are set to 128. The results on the Wiki-500K and Amazon-670K datasets are shown in Table VI. Specifically, for the Amazon-670K dataset, the training time of X-GRank-1 is 8.44ms per sample, which is lower than that of AttentionXML-1 and LightXML-1 with a remarkable reduction range of 34.06% and 40.14%. The reason why X-GRank-1 is faster than LightXML-1 is that we optimized the implementation of cluster retrieving of X-GRank-1, which reduces the data exchange between the CPU and GPU. The model size of X-GRank-1 is the smallest among the compared models, because the dimension of label embedding of X-GRank-1 is 128 while that of LightXML-1 is 400, which cuts down 182 million parameters. The experimental results verified the superiority of X-GRank in efficiency.

## V. Conclusion and future work

In this paper, we proposed a novel GRank loss to solve the rank-missing and label imbalance problems in the existing work. We apply GRank loss to XMC problem and proposed a new model named X-GRank, which includes a label retrieving task and a label ranking task. The former is a cluster-granularity multi-label classification to generate the label candidate shortlist. The latter aims to rank the retrieved labels. Both tasks used the GRank loss to boost the results. X-GRank is trained in an end-to-end way. With extensive experiments, X-GRank shows high efficiency with the best precision on widely used large scale datasets, such as Wiki-500K and Amazon-670K, compared to the current state-of-the-art methods. In the future, we plan to apply the idea of *group ranking* to more scenarios, such as multi-label image classification, image and text matching, dense passage retrieval, which opens a new perspective for researchers in a wide range of applications.

## References

[1] J. Liu, W.-C. Chang, Y. Wu, and Y. Yang, "Deep learning for extreme multi-label text classification," in *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, 2017, pp. 115–124.

[2] R. You, Z. Zhang, Z. Wang, S. Dai, H. Mamitsuka, and S. Zhu, "Attentionxml: Label tree-based attention-aware deep model for high-performance extreme multi-label text classification," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[3] T. Jiang, D. Wang, L. Sun, H. Yang, Z. Zhao, and F. Zhuang, "Lightxml: Transformer with dynamic negative sampling for high-performance extreme multi-label text classification," *arXiv preprint arXiv:2101.03305*, 2021.

[4] W.-C. Chang, H.-F. Yu, K. Zhong, Y. Yang, and I. S. Dhillon, "Taming pretrained transformers for extreme multi-label text classification," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 3163–3171.

[5] A. Mittal, N. Sachdeva, S. Agrawal, S. Agarwal, P. Kar, and M. Varma, "Eclare: Extreme classification with label graph correlations," in *Proceedings of the Web Conference 2021*, 2021, pp. 3721–3732.

[6] J. Xiong, L. Yu, X. Niu, and Y. Leng, "Xrr: Extreme multi-label text classification with candidate retrieving and deep ranking," *Information Sciences*, vol. 622, pp. 115–132, 2023.

[7] R. Babbar and B. Schölkopf, "Dismec: Distributed sparse machines for extreme multi-label classification," in *Proceedings of the tenth ACM international conference on web search and data mining*, 2017, pp. 721–729.

[8] Y. Prabhu, A. Kag, S. Harsola, R. Agrawal, and M. Varma, "Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 993–1002.

[9] S. Khandagale, H. Xiao, and R. Babbar, "Bonsai: diverse and shallow trees for extreme multi-label classification," *Machine Learning*, vol. 109, pp. 2099–2119, 11 2020.

[10] D. Saini, A. K. Jain, K. Dave, J. Jiao, A. Singh, R. Zhang, and M. Varma, "Galaxc: Graph neural networks with labelwise attention for extreme classification," in *Proceedings of the Web Conference 2021*, 2021, pp. 3733–3744.

[11] 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*, 2016, pp. 935–944.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Jun. 2019, pp. 4171–4186.

[13] F. Nielsen and G. Hadjeres, "Monte carlo information geometry: The dually flat case," *arXiv preprint arXiv:1803.07225*, 2018.

[14] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[15] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *Advances in neural information processing systems*, vol. 32, 2019.

[16] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

[17] 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*, 2014, pp. 263–272.

[18] M. Wydmuch, K. Jasinska, M. Kuznetsov, R. Busa-Fekete, and K. Dembczynski, "A no-regret generalization of hierarchical softmax to extreme multi-label classification," *Advances in neural information processing systems*, vol. 31, pp. 6358–6368, 2018.