Re-evaluating Extreme Multi-label Text Classification Methods in Tail Label Prediction

Anonymous ACL submission

Abstract

Extreme multi-label text classification (XMTC) is the task of tagging each document with the relevant labels in a very large set of predefined category labels. The most challenging part of the problem is due to a highly skewed label distribution where the majority of the categories (namely the tail labels) have very few training instances. Recent benchmark evaluations have focused on micro-averaging metrics, where the performance on tail labels can be easily overshadowed by that on the high-frequency labels (namely the head labels). This paper presents a re-evaluation of state-of-the-art (SOTA) methods based on the *binned macro-averaging F1* instead, revealing new insights into the strengths and weaknesses of representative methods, especially in tail label prediction.

1 Introduction

Extreme multi-label text classification (XMTC) is the task of tagging each document with the relevant labels in a very large set of predefined categories, in which the number of labels can be from a few thousands to more than a million. In the target space of XMTC problems, the label frequency often follows a power law. That is, a small portion of the labels have a dominating number of training instances, whereas the majority of the labels have very few training instances. The former is referred as the head labels and the latter is referred as the tail labels. The severe data sparse issue with the tail labels makes XMTC more challenging than other classification tasks where the number of categories are much smaller and the label distributions are not as skewed.

As an example, in the Wiki-31K benchmark dataset shown in figure 1, only 1% of labels has more than 100 training instances, but they cover more than 40% of all the training instances. On the other hand, 89% of labels only cover less than 30% of training instances. Other benchmark datasets show similar distribution (Appendix A).

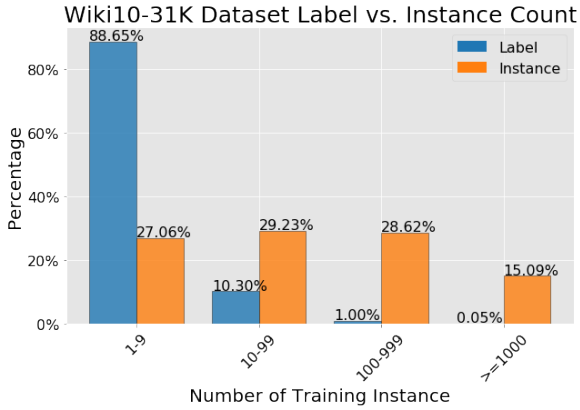

Figure 1: The percentage of labels vs. training instances in Wiki10-31K dataset that shows the label sparse issue in the skewed distribution.

It is worth pointing out that the performance of current SOTA models has been evaluated using the micro-averaging precision scores as the dominating metric (Liu et al., 2017; You et al., 2018; Ye et al., 2020; Chang et al., 2020; Jiang et al., 2021). This metric gives an equal weight to the score of each instance when computing the average performance on a test set. As a result, the global average is dominated by the system's performance on head labels. In other words, such a metric is not sufficiently sensitive for evaluating the performance of methods in tail label prediction, and possibly misleading in method comparison. Furthermore, metrics with a single score prohibit a fine-grained analysis of model performance with respect to the different label frequencies.

In order to assess the true success of the current SOTA methods in XMTC on tail label prediction, we present a re-examination of them by our proposed *binned macro-averaging F1 metric*. Specifically, the labels are binned according to their training frequencies and the performance score of each label is given an equal weight when calculating the average over all labels in the bin. By

comparing a set of SOTA neural models with a binary Support Vector Machines (SVM) as a baseline on several benchmark datasets, we reveal that the deep-layered pre-trained Transformer models (Ye et al., 2020; Jiang et al., 2021; Chang et al., 2020) perform worse than the simple SVM baseline in tail label prediction, while a label-word attention-based RNN model (You et al., 2018) outperformed the SVM baseline on two of those benchmark datasets.

2 Metrics for Re-examination

2.1 Preliminary

Let $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)_{i=1}^{N_{\text{train}}}\}$ be the training data where $\mathbf{x}_i$ is the input text and $\mathbf{y}_i \in \{-1, +1\}^L$ is the ground truth label list. The XMTC method can be formulated as learning a scoring function $f(\mathbf{x}, l) \in \mathbb{R}$ defined on an input and candidate label pair $(\boldsymbol{x}, l)$. The score should be larger for relevant labels than irrelevant labels. The objective function of a classification model is defined as:

$$\min_f \quad \frac{1}{N_{\text{train}} L} \sum_{i=1}^{N_{\text{train}}} \sum_{l=1}^{L} \mathcal{L}(y_{il}, f(\mathbf{x}_i, l))$$

where $\mathcal{L}(\cdot, \cdot)$ is an instance-wise loss, such as a hinge loss in a Linear SVM or a sigmoid loss in the final layer of a neural network.

2.2 Issues in Recent Evaluations

In recent years, SOTA methods have been competing on the metrics of micro-averaging precision at k (P@k) (Liu et al., 2017; You et al., 2018; Prabhu et al., 2018; Khandagale et al., 2019; Ye et al., 2020; Chang et al., 2020; Jiang et al., 2021), micro-averaging recall at k (R@k), the Normalized Discounted Cumulative Gain at k (NDCG@k) (Prabhu and Varma, 2014), and/or Propensity-scored Performance at k (PSP@k) (Jain et al., 2016), where k is the threshold on a ranked list. All of those metrics give an equal weight to the performance score on each test instance when computing the average performance, and thus are dominated by the system's performance on head labels and may not be sufficiently informative to evaluate the tail label prediction in highly skewed label distributions. Although macro-averaging metrics have been well understood for a long time, which can compensate the limitations of micro-averaged metrics (Yang and Liu, 1999; Gopal and Yang, 2010), a thorough evaluation on SOTA neural XMTC models is missing. As a result, the current understanding of the strengths and weaknesses of SOTA methods may not reflect their true performance in tail label prediction.

2.3 Proposed Metrics

We propose to re-examine a set of representative XMTC methods with both micro and macro-averaging metrics. Specifically, we focus on the F1@k (k omitted for simplicity), which combines precision (P) and recall (R) via a harmonic average:

$$\text{F1} = 2\frac{P \cdot R}{P + R} \tag{1}$$

The precision and recall for a predicted ranked list $\boldsymbol{p}$ are computed by $\text{P} = \frac{\text{TP}}{\text{TP+FP}}, \text{R} = \frac{\text{TP}}{\text{TP+FN}}$ according to the confusion matrix in table 1.

	l in $\boldsymbol{y}_i$	l not in $\boldsymbol{y}_i$
l in $\boldsymbol{p}_i$	True Positive(TP_l^i)	False Positive(FP_l^i)
l not in $\boldsymbol{p}_i$	False Negative(FN_l^i)	True Negative(TN_l^i)

Table 1: Confusion Matrix for instance i and label l given the ranked list $\boldsymbol{p}_i$.

The micro and macro-average combines the statistics in the confusion matrix in different ways. Specifically, given N_{test} instances and L labels, the micro-average aggregates the statistics in the $N_{\text{test}} \times L$ confusion matrices globally, while the macro-average computes the scores on individual category first (F1_l), and then take an average over all the categories ($\text{F1} = \frac{1}{|\mathbb{L}|} \sum_{i \in \mathbb{L}} \text{F1}_l$).

Binned Macro-averaging: Let $\mathbb{L}_b$ be a set of labels with similar training frequencies grouped in bin b, then the binned macro F1 is calculated by:

$$\frac{1}{|\mathbb{L}_b|} \sum_{l \in \mathbb{L}_b} \text{F1}_l \tag{2}$$

As a designed choice, we set $\text{F1}_l = 0$ if a label l is never predicted in any top k ranked list.

Relative Improvement over SVM Baseline: To have a clear view of model performance on head and tail labels, we report the relative improvement of the SOTA models over the SVM baseline by the macro-averaging F1 score on each bin of labels.

3 Methods for Comparison

We introduce the 4 SOTA deep learning models, which are compared with a tf-idf+SVM baseline:

X-Transformer (Chang et al., 2020) was proposed to tame large Transformer models to the XMTC task by a two stage training procedure: 1) a cluster-level classification with large Transformer models

such as BERT-large (Devlin et al., 2018), Roberta-large (Liu et al., 2019) and XLNet-large (Yang et al., 2019), and 2) a within cluster ranker implemented with one-vs-all SVM with both tf-idf and Transformer features as input.

APLC-XLNet (Ye et al., 2020) achieves an end-to-end training with the XLNet-base model (12 layers transformer) by decreasing sizes of document embedding to solve the scalability issue. Specifically, a pooler function was applied on top of the XLNet special [CLS] token to the decrease embedding size, whose scale is determined by the label frequency.

LightXML (Jiang et al., 2021) extracts the [CLS] embeddings from different layers of a pre-trained model to form a document embedding with richer semantic information. The same embedding is used for the cluster-level classifier and the within cluster ranker, which are trained alternatively.

AttentionXML (You et al., 2018) uses a label-word attention mechanism on top of a bi-directional LSTM to create label specific document embeddings, which are passed to a MLP to obtain the relevance scores of the labels.

Tf-idf+SVM (Cortes and Vapnik, 1995) is a simple baseline for XMTC. Although more complicated kernels such as the RBF kernel can be used, (Chang and Lin, 2011) shows that a Linear SVM achieves a similar performance with a RBF kernel SVM when the feature space is large (refer to Table 2 for the number of features in the benchmark datasets).

It is worth mentioning that the deep Transformer-based models encodes input text into a *fixed document embedding*, while the AttentionXML learns *label specific document embeddings* via label-word attention.

4 Datasets

We use three benchmark datasets: EURLex-4K (Loza Mencía and Fürnkranz, 2008), Wiki10-31K (Zubiaga, 2012) and AmazonCat-13K (McAuley and Leskovec, 2013). The statistics of the datasets are shown in Table 2.

We partition the bins according to the absolute training label frequency, which reflects the difficulty of optimization across datasets. For the bins with $1 \sim 9$ training instances, the 3 datasets cover 63.48%, 88.65% and 30% of training labels respectively. If a model cannot perform well on those bins, it means that a large portion of labels are not being predicted accurately.

The details of the experiment settings, descriptions of SOTA models and the training hyperparameters are discussed in Appendix C.

5 Evaluation Results

We present the evaluation results with both the micro and macro-averaging F1@k metrics. As a design choice, We pick $k = 19$ for Wiki10-31K dataset whose average number of labels are 18.64, and $k = 5$ for the rest of the datasets. More results are shown in Appendix D.

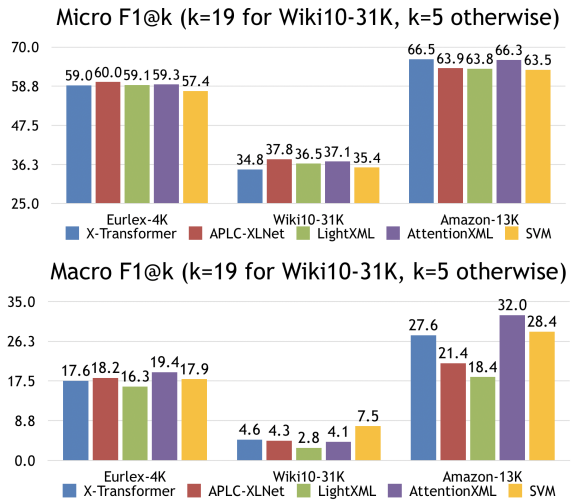

Figure 2: The result of micro and macro-averaging F1@k metrics. The two metrics gives different conclusion on the best performing model.

5.1 Performance in Micro-averaging Metrics

At the top of figure 2, we report the micro-averaging F1@k for the SOTA models on the benchmark datasets. In the micro-based evaluation, all the SOTA neural models outperform the SVM baseline on all 3 benchmark datasets. We found that the deep Transformer model achieves the best performs across the datasets. Specifically, APLC-XLNet is the best on the EUR-Lex and Wiki10-31K datasets, and the X-Transformer is the best on the AmazonCat-13K dataset.

5.2 Performance in Macro-averaging Metrics

At the bottom of figure 2, the macro-averaging F1@k is reported. Specifically, the SVM baseline performs the best in the Wiki10-31K, where the label space is both large and skewed. It also achieves competitive results on the other two datasets. Especially on the AmazonCat-13K dataset, the SVM beats all the deep Transformer-based models which perform better on the micro-averaging metric.

Dataset	N_{train}	N_{test}	F	$\bar{L}_d$	L	$\lvert\mathbb{L}_1\rvert$	$\lvert\mathbb{L}_2\rvert$	$\lvert\mathbb{L}_3\rvert$	$\lvert\mathbb{L}_4\rvert$
EURLex-4K	15,539	3,809	186,104	5.30	3,956	2,413	1,205	182	1
Wiki10-31K	14,146	6,616	101,938	18.64	30,938	26,545	3,084	300	16
AmazonCat-13K	1,186,239	306,782	203,882	5.04	13,330	3,936	5,813	2,862	719

Table 2: N_{train} and N_{test} are the number of training and testing instances respectively. F is the tf-idf feature size. $\bar{L}_d$ is the average number of labels per document. L is the number of labels. $|\mathbb{L}_k|$ refers to the number of labels in bin k. The labels in bin $[1, 2, 3, 4]$ have $[1 \sim 9, 10 \sim 99, 100 \sim 999, 1000 \sim]$ instances respectively. The bin partition is used in our binned macro-averaging F1 metric.

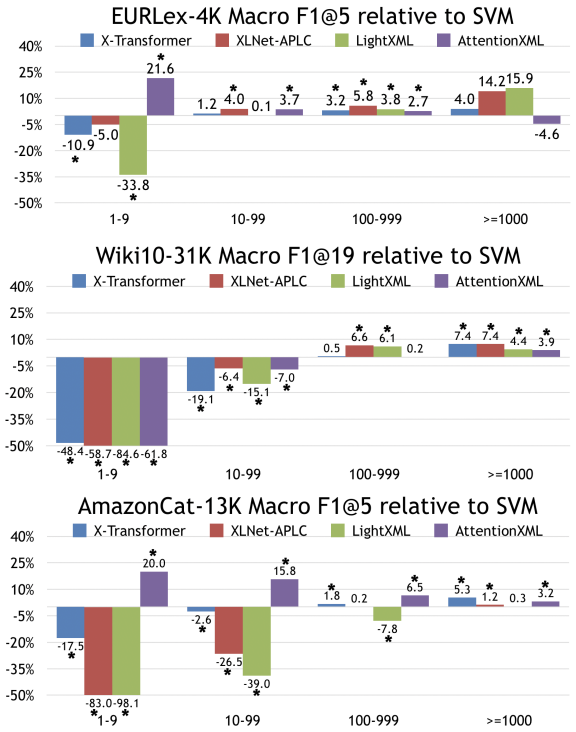

Figure 3: The relative improvement of SOTA deep learning models over tf-idf+SVM baseline on the binned macro-averaging F1@5 metrics. The labels are partitioned to bins whose labels have $[1 \sim 9, 10 \sim 99, 100 \sim 999, \geq 1000]$ training instances respectively. $*$ indicates the macro t-test is significant ($p < 0.01$).

We also observe that the AttentionXML model is the best on both the EURLex-4K and the AmazonCat-13K datasets, though it is not the best in any datasets when evaluated with the micro-averaging metric.

Binned Macro-averaging F1 Metric The relative improvement of the binned macro-averaging F1@k over the tf-idf+SVM baseline is reported in figure 3, which gives a fine-grained comparison of the model performance on the tail (or head) labels. We conduct the macro t-test to justify the significance of the results ($p < 0.01$) (Yang and Liu, 1999) on the bins with more than 10 labels. We have the following observations: 1) the deep learning models tend to perform better than SVM on middle and head labels. 3) all the deep Transformer-based models underperform the SVM baseline on the tail labels. 2) The AttentionXML model tends to outperform the SVM baseline on the tail labels on 2 datasets.

5.3 Analysis

Although the tf-idf+SVM model underperforms deep learning models on the micro-averaging metrics, it has an advantage over the deep Transformer-based models on tail labels. Our explanation is that since the tf-idf feature is computed unsupervisedly, it is irrelevant to the number of training examples. Therefore, the feature maintains separable in the low resource cases. To the contrary, the Transformer-based models rely on supervised learning to encode semantics into a fixed feature vector, which is more sensitive to the scarceness of training signals in the tail label.

The AttentionXML learns label specific word embeddings via the label-word attention, which avoids the information of head labels to overshadow that of tail labels in a fixed embedding. Additionally, the local label to word matching captures the semantic similarity between the two which can not be achieved by the tf-idf term frequency. Therefore, it has additional gains over SVM baseline on two datasets. Still, it underperforms SVM when label is extremely sparse in Wiki10-31K.

6 Conclusion

In this paper, we propose the binned macro-averaging F1 metric to re-evaluate the SOTA XMTC model performance especially on the tail labels. Our evaluation reveals that although the tf-idf+SVM model underperforms the SOTA deep learning models on the micro-averaging metrics, it has an advantage on the tail label prediction where the training data is scarce, shedding insights on model strength and weakness in feature learning.

References

Chih-Chung Chang and Chih-Jen Lin. 2011. Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27.

Wei-Cheng Chang, Hsiang-Fu Yu, Kai Zhong, Yiming Yang, and Inderjit S Dhillon. 2020. Taming pretrained transformers for extreme multi-label text classification. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3163–3171.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Siddharth Gopal and Yiming Yang. 2010. Multilabel classification with meta-level features. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 315–322.

Himanshu Jain, Yashoteja Prabhu, and Manik Varma. 2016. 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*.

Ting Jiang, Deqing Wang, Leilei Sun, Huayi Yang, Zhengyang Zhao, and Fuzhen Zhuang. 2021. Lightxml: Transformer with dynamic negative sampling for high-performance extreme multi-label text classification. *arXiv preprint arXiv:2101.03305*.

Sujay Khandagale, Han Xiao, and Rohit Babbar. 2019. Bonsai – diverse and shallow trees for extreme multi-label classification.

Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. 2017. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 115–124.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Eneldo Loza Mencía and Johannes Fürnkranz. 2008. Efficient pairwise multilabel classification for large-scale problems in the legal domain. In *Machine Learning and Knowledge Discovery in Databases*. Springer Berlin Heidelberg.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: Understanding rating dimensions with review text. page 165–172. Association for Computing Machinery.

Yashoteja Prabhu, Anil Kag, Shrutendra Harsola, Rahul Agrawal, and Manik Varma. 2018. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In *Proceedings of the 2018 World Wide Web Conference*, page 993–1002. International World Wide Web Conferences Steering Committee.

Yashoteja Prabhu and Manik Varma. 2014. 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*, page 263–272, New York, NY, USA. Association for Computing Machinery.

Yiming Yang and Xin Liu. 1999. A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42–49.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Hui Ye, Zhiyu Chen, Da-Han Wang, and Brian Davison. 2020. Pretrained generalized autoregressive model with adaptive probabilistic label clusters for extreme multi-label text classification. In *International Conference on Machine Learning*, pages 10809–10819. PMLR.

Ronghui You, Zihan Zhang, Ziye Wang, Suyang Dai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2018. Attentionxml: Label tree-based attention-aware deep model for high-performance extreme multi-label text classification. *arXiv preprint arXiv:1811.01727*.

Arkaitz Zubiaga. 2012. Enhancing navigation on wikipedia with social tags.

A Data Distribution

In Figure 4, we show the percentage of label vs. training instance in each bin in the benchmark datasets: EURLex-4K, Wiki10-31K and AmazonCat-13K dataset. In all the datasets, a small percentage of head labels cover most of the of training instances, while a large percentage of tail labels only cover a few training instances.

B Evaluation Metrics

We include more discussion and reference for the micro-averaging metrics and our proposed binned macro-averaging F1 metric.

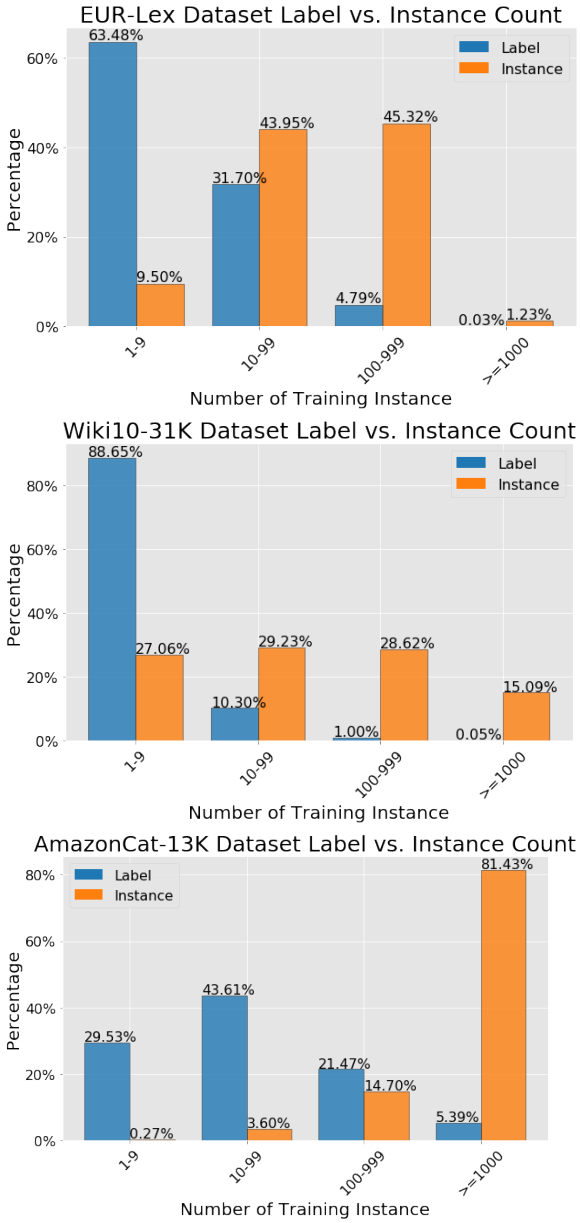

Figure 4: The percentage of label vs. training instance in EURLex-4K, Wiki10-31K and AmazonCat-13K dataset. A small percentage of head labels cover most of the of training instances, while a large percentage of tail labels only cover a few training instances.

B.1 Micro-averaging Metrics

In XMTC, there are multiple true labels for each instance, so it is important to present a ranked list of predicted labels for evaluation. The micro-averaging metrics calculate a score for each of the ranked list and then take an average over all the test instances.

$$\text{Micro@k} = \frac{1}{N}\sum_{i=1}^{N}\text{Metric}(\mathbf{p}_i, \mathbf{y}_i) \qquad (3)$$

where N is the number of test instances and $\mathbf{p}_i, \mathbf{y}_i$ are the predicted top k ranked list and ground truth labels list. Metric is a function to score the quality of the ranked list based on the ground truth labels list. In the following, we omit the index of instance i for clarity.

In most of the previous work, the Metric function is chosen to be the micro-averaging Precision at k (P@k) (Liu et al., 2017; You et al., 2018; Ye et al., 2020; Chang et al., 2020; Jiang et al., 2021). Specifically, this metric evaluates the quality of the top k of the prediction ranked list for a given test instance:

$$P@k = \frac{1}{k}\sum_{l=1}^{k}\mathbb{1}_{\mathbf{y}}(\mathbf{p}_l) \qquad (4)$$

where $\mathbf{p}_l$ is the l-th label in the predicted ranked list $\mathbf{p}$ and $\mathbb{1}_{\mathbf{y}}$ is the indicator function.

Other choices of the Metric function include N@k (normalized Discounted Cumulative Gain at k) or PSP@K (Propensity-scored Performance at k) (Jain et al., 2016). The N@k is defined as:

$$DCG@k = \sum_{l=1}^{k}\frac{\mathbb{1}_{\mathbf{y}}(\mathbf{p}_l)}{\log(l+1)}$$

$$iDCG@k = \sum_{l=1}^{\min(k,\|\mathbf{y}\|)}\frac{1}{\log(l+1)}$$

$$N@k = \frac{DCG@k}{iDCG@k}$$

Although N@k and P@k are calculated differently, You et al. shows that they are the same metrics when measuring the quality of ranked lists.

The PSP @ k is defined as:

$$PSP@k = \frac{1}{k}\sum_{l=1}^{k}\frac{\mathbb{1}_{\mathbf{y}}(\mathbf{p}_l)}{\text{prop}(\mathbf{p}_l)}$$

where $\text{prop}(\mathbf{p}_l)$ is the propensity score (Jain et al., 2016) of label $\mathbf{p}_l$, which gives higher weight for tail labels. Although PSP@k were used as evaluation metric for the long tail problem, it is still not informative enough to compare the performance of head labels vs. tail labels due to following: 1) It is still a micro-averaging metric that can be influenced by label with more instances. 2) Since it summarizes the performance into one number, we are not sure if the gain comes from the improvement of head labels or tail labels. Actually, an improvement in either type of labels will give an increment in the final score.

B.2 Macro-averaging Metrics

As we discussed above, the micro-averaging metrics are not informative enough to measure the performance of different types of labels, because they assign equal weights to each instance when taking the averaging. As a remedy, a label level evaluation should assign equal weights to each label when taking the average. Therefore, macro-averaging metrics should be applied for evaluating the label level performance, especially for the tail labels. To create groups for large label size, we split the labels in the scale of exponential of 10 as a design choice, e.g. $\{1 \sim 9, 10 \sim 99, 100 \sim 999, \dots\}$.

We used the F1 metric to balance the precision and recall. The tail labels tend to be under predicted due to the more frequent negative gradient penalty. As a result, tail labels tend to be predicted less often than the head labels, and thus the recall is low. If a model hardly predicts any tail labels, it should receive a low score for the corresponding bin. Therefore, we apply the F1 metric to balance the number of predictions (recall) and the accuracy of the predictions (prediction).

C Experiment Settings

C.1 Re-evaluation Experiment Settings

We choose the non-neural SVM as our baseline and investigate 4 SOTA deep learning models for XMTC: AttentionXML (You et al., 2018), X-Transformer (Chang et al., 2020), APLC (Ye et al., 2020) and LightXML (Jiang et al., 2021).

In the original papers, the experiments are conducted in different settings, e.g. (Chang et al., 2020; Jiang et al., 2021) reports the ensemble of multiple models and different works have their own data processing method. For fair comparison, we run our experiments with single model with the following training and testing data. We obtain the datasets from the Extreme classification Repository[1]. However, the repository only contains the stemmed version of EURLex-4K, which hurts the performance of pretrained Transformer models whose tokenizers are applied to unstem natural text. Therefore, we obtain the unstemmed version of the EURLex-4K from the APLC-XLNet github[2].

C.2 Training Settings and Hyperparameters

For the AttentionXML model, we use the same hyperparameter as in their code [3]. For the X-Transformer mode, the reuse the released pretrained model [4] for evaluation.

For the end-to-end Transformer-based models including APLC-XLNet and LightXML, we train the model using the same framework as APLC, including discriminative fine-tuning and slanted triangular learning rates (Ye et al., 2020). The discriminative fine-tuning decouples the learning rate of model into 3 parts: The base Transformer module, the pooler module and the linear classifier module, such that the Transformer module receives smallest learning rate and the linear module receives largest learning rate. The slanted triangular learning rates allows the model to warm up with a slowly increasing learning rate first, and then decrease the learning rate to let the model converge stably.

D More Experimental Results

In Table 3, we report the micro P@k, micro F1@k and the macro F1@k (k=1, 3, 5) for the SOTA models on the benchmark datasets. We observe that the micro P@5 and the F1@5 shows similar conclusions on which models perform the best, but the macro F1@5 gives different conclusions.

In Table 4, we report the N@k and PSP@k (k=1, 3, 5) for the SOTA models on the benchmark datasets. For N@k, it shows the same conclusions with P@k on which model performs the best (they are mathematically equivalent). Although the PSP metric were used to measure the performance of tail label prediction, we found that it gives higher score for the some of deep Transformer-base models than the SVM baseline, e.g. X-Transformer, XLNet-APLC on EURLex-4K, XLNet-APLC on Wiki10-31K and X-Transformer on AmazonCat-13K. However, those models actually underperform the SVM baseline on tail labels in our binned macro-averaging F1 metric designed for tail label evaluation. This shows that the PSP metric still scarifies from the micro-averaging when applied to evaluate the tail label performance.

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

[2] https://github.com/huiyegit/APLC_XLNet.git

[3] https://github.com/yourh/AttentionXML

[4] https://github.com/OctoberChang/X-Transformer

EURLex-4K									
Methods	Micro			Micro			Macro		
	P@1	P@3	P@5	F1@1	F1@3	F1@5	F1@1	F1@3	F1@5
SVM	83.44	70.62	59.08	26.50	51.06	57.37	5.64	13.28	17.89
X-Transformer	85.46	72.87	60.79	27.14	52.69	59.03	5.12	12.86	17.56
XLNet-APLC	86.83	74.34	61.94	27.43	53.55	59.96	6.66	14.51	18.19
LightXML	86.98	73.38	61.07	27.48	52.86	59.12	5.68	12.85	16.31
AttentionXML	85.12	72.80	61.01	27.03	52.64	59.25	7.69	15.80	19.41
Wiki10-31K									
Methods	Micro			Micro			Macro		
	P@1	P@3	P@5	F1@1	F1@3	F1@5	F1@1	F1@3	F1@5
SVM	84.61	74.64	65.89	8.45	20.33	27.42	0.18	0.97	1.99
X-Transformer	87.12	76.51	66.69	8.70	20.84	27.76	0.14	0.51	1.02
XLNet-APLC	88.59	78.30	68.87	8.85	21.33	28.67	0.31	0.93	1.59
LightXML	88.59	78.51	68.84	8.85	21.39	28.65	0.25	0.69	1.10
AttentionXML	86.46	77.22	67.98	8.63	21.03	28.30	0.39	1.05	1.67
AmazonCat-13K									
Methods	Micro			Micro			Macro		
	P@1	P@3	P@5	F1@1	F1@3	F1@5	F1@1	F1@3	F1@5
SVM	93.20	78.89	64.14	30.54	58.42	63.49	3.86	16.29	28.36
X-Transformer	95.75	82.46	67.22	31.38	61.06	66.54	1.90	12.50	27.55
XLNet-APLC	94.56	79.78	64.59	30.99	59.07	63.93	3.95	13.32	21.38
LightXML	94.61	79.83	64.45	31.00	59.11	63.79	1.60	8.38	18.36
AttentionXML	95.53	82.03	67.00	31.31	60.74	66.31	7.12	21.94	31.98

Table 3: SVM and SOTA Deep Transformer Models evaluated on benchmark datasets: EURLex-4K, Wiki10-31K and AmazonCat-13K. The metrics are micro P@k, micro F1@k and macro F1@k for k=$1, 3, 5$.

EURLex-4K						
Methods	N@1	N@3	N@5	PSP@1	PSP@3	PSP@5
SVM	83.44	73.58	65.70	38.76	46.71	51.17
X-Transformer	85.46	75.84	67.62	37.85	47.05	51.81
XLNet-APLC	86.83	77.29	68.79	42.21	49.83	52.88
LightXML	86.98	76.53	68.05	40.54	47.56	50.50
AttentionXML	85.12	75.74	67.71	44.20	50.85	53.87
Wiki10-31K						
Methods	N@1	N@3	N@5	PSP@1	PSP@3	PSP@5
SVM	84.61	76.99	70.34	11.89	14.23	15.96
X-Transformer	87.12	79.00	71.56	12.52	13.62	14.63
XLNet-APLC	88.59	80.72	73.58	14.43	15.38	16.47
LightXML	88.59	80.87	73.58	14.09	14.87	15.52
AttentionXML	86.46	79.41	72.47	14.49	15.65	16.54
AmazonCat-13K						
Methods	N@1	N@3	N@5	PSP@1	PSP@3	PSP@5
SVM	93.20	82.87	76.47	51.26	64.69	72.34
X-Transformer	95.75	86.26	79.80	51.42	66.14	75.57
XLNet-APLC	94.56	83.89	77.29	52.55	65.11	71.36
LightXML	94.61	83.95	77.21	50.70	63.14	70.13
AttentionXML	95.53	85.87	79.54	54.94	69.48	76.45

Table 4: SVM and SOTA Deep Transformer Models evaluated on benchmark datasets: EUR-Lex (4K), Wiki10-31K and AmazonCat-13K. The metrics are N@k and PSP@k for k=$1, 3, 5$.