Long-tailed Extreme Multi-label Text Classification by the Retrieval of Generated Pseudo Label Descriptions

Ruohong Zhang and **Yau-Shian Wang**
ruohongz,yaushiaw@andrew.cmu.edu

Yiming Yang
yiming@cs.cmu.edu

Donghan Yu
dyu2@cs.cmu.edu

Tom Vu
tom.m.vu@gmail.com

Likun Lei
llei@flexport.com

Abstract

Extreme Multi-label Text Classification (XMTC) has been a tough challenge in machine learning research and applications due to the sheer sizes of the label spaces and the severe data scarcity problem associated with the long tail of rare labels in highly skewed distributions. This paper addresses the challenge of tail label prediction by leveraging the power of dense neural retrieval model in mapping input documents (as queries) to relevant label descriptions. To further enhance the quality of label descriptions, we propose to generate pseudo label descriptions from a trained bag-of-words (BoW) classifier, which demonstrates better classification performance under severe scarce data conditions. The proposed approach achieves the state-of-the-art (SOTA) performance of overall label prediction on XMTC benchmark datasets and especially outperforms the SOTA models in the tail label prediction. We also provide a theoretical analysis for relating the BoW and neural models w.r.t. performance lower bound.

1 Introduction

Extreme multi-label text classification (XMTC) is the task of tagging documents with relevant labels in a very large and often skewed candidate space. It has a wide range of applications, such as assigning subject topics to news or Wikipedia articles, tagging keywords for online shopping items, classifying industrial products for tax purposes, etc.

The most difficult part in solving the XMTC problem is to train classification models effectively for the rare labels in the long tail of highly skewed distributions, which suffers severely from the lack of sufficient training instances. Efforts addressing this challenge by the text classification community include Bayesian modeling of graphical dependencies among labels (Gopal and Yang, 2010; Gopal et al., 2012), novel loss or regularization of label embeddings (Babbar and Schölkopf, 2019a; Wei et al., 2021), clustering-based algorithms (Chang et al., 2020; Khandagale et al., 2019; Prabhu et al., 2018), and so on. Despite the remarkable progresses made so far, the problem is still very far from being well solved. Figure 1 shows the performance of X-Transformer (Chang et al., 2020), one of the state-of-the-art (SOTA) XMTC models, on the Wiki10-31K benchmark dataset (with over 31k labels). The horizontal axis is the ranks of the labels sorted from rare to common and the vertical axis is the text classification performance measured in macro-averaged $F1@19$ (higher the better) for binned labels (100 labels per bin). The blue curve is the result of X-Transformer, which has the scores close to 0 (worst possible score) for nearly half of the total labels. In other words, SOTA methods in XMTC still perform poorly in tail label prediction.

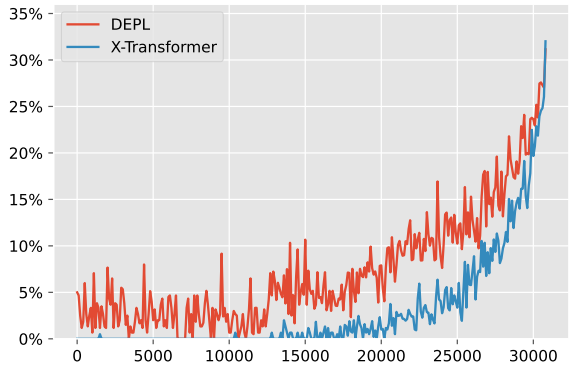

Figure 1: The classification performance of X-Transformer and DEPL (ours) measured in macro-averaged $F1@19$ on the Wiki10-31K dataset.

In this paper, we seek solutions for tail label prediction from a new angle: we introduce a novel framework, namely the Dual Encoder with Pseudo Label (DEPL). It treats each input document as a query and uses a neural network model to retrieve relevant labels from the candidate space based on the textual descriptions of the labels. The underlying assumption is, if the label descriptions are

Findings of the Association for Computational Linguistics: EACL 2023, pages 1092–1106

highly informative for text-based matching, then the retrieval system should be able to find relevant labels. The system would be particularly helpful for tail label prediction as the retrieval effectiveness does not necessarily rely on the availability of a large number of training instances, which is what the tail labels are lacking.

The next research question that we tackle is how to obtain highly informative descriptions for each label without human annotation. In reality, class names are often available but they are typically one or two words, which cannot be sufficient for retrieval-based label prediction. Therefore, we propose to augment the label description with statistical learning algorithms. Specifically, we train linear support vector machine (SVM) model with the bag-of-words (BoW) features, such as tf-idf, to automatically generate informative keywords for each label, which we call the *pseudo description* of the label. Since the learned label embeddings of the BoW classifier encode token importance information, it is natural and efficient to leverage them for keywords extraction. In sections 4 and 6, we further provide theoretical motivations and empirical evidence to show the advantage of unsupervised statistical features for classification under extreme scarce data conditions.

The result of our approach (DEPL) is shown as the red curve in Figure 1, which significantly outperforms the blue curve of X-Transformer not only in the tail-label region but also in all other regions. We also observed similar improvements by DEPL over strong baselines on other benchmark datasets (see section 6). Our main contributions are summarized as the following:

1. We propose DEPL, a retrieval-based model to alleviate the difficulty in tail label prediction by matching the semantics between documents and augmented label descriptions which are generated automatically by a statistical model with BoW features.

2. We provide theoretical analyses to motivate the usage of BoW feature for classification under scarce data setting, and prove a performance lower bound of the neural model.

3. We did extensive experiments with different tail label evaluation metrics to show that our method significantly and consistently outperforms strong baselines on multiple challenging benchmark datasets.

2 Related Work

XMTC Classifier Traditional BoW classifiers rely on the bag-of-words features such as one-hot vector with tf-idf weights, which capture the word importance in a document. Examples include one-vs-all SVM models such as DiSMEC (Babbar and Schölkopf, 2017), ProXML (Babbar and Schölkopf, 2019b), PPDSparse (Yen et al., 2017), tree-based models such as Parabel (Prabhu et al., 2018) and Bonsai (Khandagale et al., 2019).

To compensate for the lack of semantics in BoW features, deep learning models were proposed for XMTC. Examples include CNN-based models such as XML-CNN (Liu et al., 2017) and SLICE (Jain et al., 2019), RNN-based models such as AttentionXML (You et al., 2018) and Transformer-based models such as X-Transformer (Chang et al., 2020), LightXML (Jiang et al., 2021) and APLC-XLNet (Ye et al., 2020).

Label Description The SiameseXML (Dahiya et al., 2021) for XMTC encodes both input documents and label descriptions with pretrained word embeddings with shallow networks and leverages the embedding matching. The SOTA pretrained Transformer-based models (Chang et al., 2020; Jiang et al., 2021) leverage the label descriptions to build label clusters. To generate label descriptions, Chai et al. (2020) adopt reinforcement learning to produce extended label descriptions from predefined label descriptions. However, the algorithm can not scale to the extreme label space and relies on the availability of sufficient training data.

3 Proposed Method

3.1 Preliminaries

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 $\mathbf{y}_i \in \{0,1\}^L$ are the binary ground truth labels of size L. Given an instance $\boldsymbol{x}$ and a label l, a classification system produces a matching score of the text and label:

$$f(\boldsymbol{x}, l) = \langle \phi(\boldsymbol{x}), \boldsymbol{w}_l \rangle$$

where $\phi(\boldsymbol{x})$ represent the document feature vector and $\boldsymbol{w}_l$ represents the label embedding of l. The dot product $\langle \cdot, \cdot \rangle$ is used as the similarity function.

Typically, the label embedding $\boldsymbol{w}_l$ is randomly initialized and trained from the supervised signal. While learning the embedding as free parameters is expressive when data is abundant, it could be difficult to be optimized under the scarce data situation.

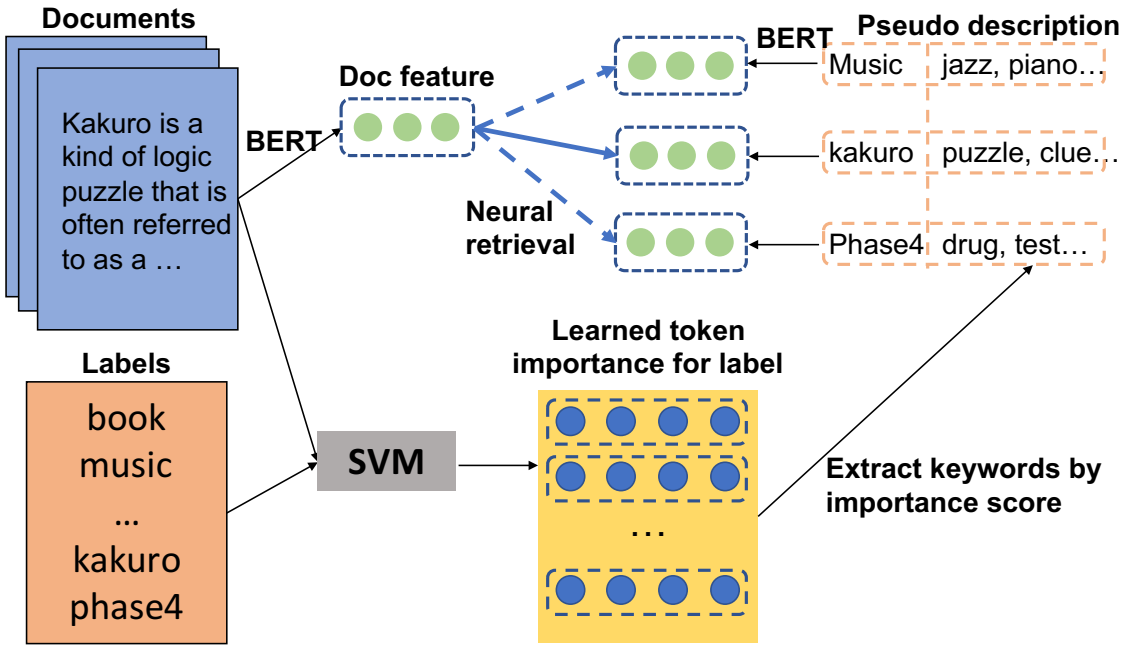

Figure 2: The proposed DEPL framework. First, we train a BoW classifier (SVM) and extract the top keywords from the label embeddings according to the learned token importance. Then, we concatenate the keywords with the original label names to form pseudo descriptions. Finally, we leverage the neural retrieval model to rank the labels according to semantic matching between document text and label descriptions.

Sketch of Method DEPL tackles the long-tailed XMTC by neural retrieval with generated pseudo label descriptions, as shown in figure 2. Instead of learning the label embedding from scratch, the retrieval module directly leverages the **semantic matching** between the document and label text, providing a strong inductive bias on tail label prediction. Next, we introduce the components of our system in details.

3.2 Generated Pseudo Label Description

As the provided label names are usually short and noisy, we augment it with generated pseudo label description from a SVM model. As the tf-idf features $\phi_t(\boldsymbol{x})$ used by SVM are sparse, we also call the statistical model a *sparse* model:

$$f_{\text{sparse}}(\boldsymbol{x}, l) = \langle \phi_t(\boldsymbol{x}), \boldsymbol{w}_l^{\text{svm}} \rangle$$

The label embedding weight $\boldsymbol{w}_l^{\text{svm}}$ is optimized with the hinge loss:

$$\mathcal{L}_{\text{hinge}} = \frac{1}{LB} \sum_{i=1}^{B} \sum_{l=1}^{L} \max(0, 1 - \tilde{y}_l \cdot f_{\text{sparse}}(\boldsymbol{x}_i, l))$$

where $\tilde{y}_l = 2y_l - 1 \in \{-1, 1\}$, B is the batch size.

For a trained SVM model, $\boldsymbol{w}_l^{\text{svm}}$ has the dimension equal to the vocabulary size and each value w_{li}^{svm} of the label embedding denotes the learned importance of the token i w.r.t label l. We select the top k most important tokens (ranked according to the importance score) as keywords, which are appended to the original label name to form the pseudo label description:

$$\text{pseudo\_label}(l) = \text{label\_name}(l) \oplus \text{keywords}(l)$$

where $\oplus$ is the append operation.

3.3 Retrieval Model with Label Text

DEPL leverages the semantic matching of document and label texts via a dual encoder model (Gao and Callan, 2021; Xiong et al., 2020; Luan et al., 2020; Karpukhin et al., 2020). We use the BERT (Devlin et al., 2018) model as the backbone of our neural encoder, which is shared for both the document and label text encoding. Since a neural model encodes textual inputs into condensed vector representations, we call them *dense* models.

The similarity between text and label representation is measured by:

$$f_{\text{dual}}(\boldsymbol{x}, l) = \langle \phi_{\text{doc}}(\boldsymbol{x}), \phi_{\text{label}}(\text{text}(l)) \rangle$$

where $\text{text}(l)$ is the textual information of the label l. When the textual information only includes the label name given in the dataset, we call the model

DE-ret. Otherwise, when the textual information includes the generated pseudo label description, we call the model **DEPL**.

The document embedding $\phi_{\text{doc}}(\boldsymbol{x})$ is obtained from the CLS embedding of the BERT model followed by a linear pooling layer:

$$\phi_{\text{doc}}(\boldsymbol{x}) = \boldsymbol{W}_{doc} \cdot \text{BERT}(\boldsymbol{x}, \texttt{CLS}) + \boldsymbol{b}_{doc}$$

where $\text{BERT}(\boldsymbol{x}, \texttt{CLS})$ represents the contextualized embedding of the special CLS token. $\boldsymbol{W}_{doc}$ and $\boldsymbol{b}_{doc}$ are the weights and biases for the document pooler layer.

For the label embedding $\phi_{\text{label}}(\text{text}(l))$, we take an average of the last hidden layer of BERT followed by a linear pooler layer:

$$\phi_{\text{label}}(\text{text}(l)) = \boldsymbol{W}_{label} \cdot \psi_{bert}(\text{text}(l)) + \boldsymbol{b}_{label} \quad (1)$$

$$\psi_{bert}(\text{text}(l)) = \frac{1}{|\,\text{text}(l)|} \sum_{j=1}^{|\,\text{text}(l)|} \text{BERT}(\text{text}(l), j) \quad (2)$$

where $\text{BERT}(\text{text}(l), j)$ represents the contextualized embedding of the j-th token in $\text{text}(l)$ obtained from the last hidden layer of the BERT model. $\boldsymbol{W}_{label}$ and $\boldsymbol{b}_{label}$ are the weights and biases for the label pooler layer. In the equation 2, the average embedding of label tokens yields better performance empirically than the CLS embedding possibly because the keywords are not natural language, and BERT may not effectively aggregate such type of information into CLS.

Learning with Negative Sampling Since calculating all the label embeddings for each batch is both expensive and prohibitive by the memory limit, we resort to negative sampling strategies for in-batch optimization. Specifically, we sample a fixed-sized subset of labels for each batch containing: 1) all the positive labels of the instances in the batch, 2) the top negative predictions by the sparse classifier as the hard negatives, and 3) the rest of the batch is filled with uniformly random sampled negatives labels.

Let $\mathbb{S}_b$ be the subset of labels sampled for a batch. The objective for the dual encoder is:

$$\mathcal{L}_{\text{dual}} = -\frac{1}{B|\mathbb{S}_b|} \sum_{i=1}^{B} \left(\sum_{p \in \boldsymbol{y}_i^+} \log \sigma(f_{\text{dual}}(\boldsymbol{x}_i, p)) + \sum_{n \in \mathbb{S}_b \setminus \boldsymbol{y}_i^+} \log \sigma((1 - f_{\text{dual}}(\boldsymbol{x}_i, n))) \right)$$

where B is the batch size, $\boldsymbol{y}_i^+$ is the positive labels for instance i, and σ is the sigmoid function.

3.4 Connection of Sparse and Dense Model

Complementary features: the sparse model uses the tf-idf feature based on corpus-level token statistics, while the dense model relies on the knowledge of the language learned during pretraining. The two types of features focus on different aspects of the text corpus and the combination of the two brings gains in performance.

Difference from ensemble: utilizing the augmented text for retrieval is better than a pure ensemble of sparse and dense methods such as in X-Transformer. In the ensemble method, the semantic meaning of important tokens in a label embedding learned from sparse classifier is not leveraged. By extracting the keywords from the sparse label embedding and presenting them as pseudo label descriptions, our model can additionally exploit the value of those key token semantics.

3.5 Enhance Classification with Retrieval

Our introduced retrieval model can be combined with a neural classifier for a performance boost on overall label classification (since our retrieval model is primarily targeted on improving tail label performance). In a neural classification system, the label embedding is treated as free parameters to be learned from supervised data, which is more expressive for labels with abundant training instances. The neural classifier learns the function:

$$f_{\text{cls}}(\boldsymbol{x}, l) = \langle \phi_{\text{doc}}(\boldsymbol{x}), \boldsymbol{w}_l^{\text{cls}} \rangle \quad (3)$$

We propose to enhance the classification model with the retrieval mechanism by jointly fine-tuning:

$$f_{\text{dual-cls}}(\boldsymbol{x}, l) = \frac{\sigma(f_{\text{dual}}(\boldsymbol{x}, l)) + \sigma(f_{\text{cls}}(\boldsymbol{x}, l))}{2} \quad (4)$$

The classification and retrieval modules share the same BERT encoder. We refer to the system as **DEPL+cls**. The object function $\mathcal{L}_{\text{dual-cls}}$ is similar to $\mathcal{L}_{\text{dual}}$ except for replacing f_{dual} with $f_{\text{dual-cls}}$.

The **DEPL+cls** model looks like an ensemble of the two systems at first sight, but there are two major differences: 1) As the BERT encoder is shared between the classification and retrieval modules, it doesn't significantly increase the number of parameters as in (Chang et al., 2020; Jiang et al., 2021); and 2) when the two modules are optimized together, the system can take advantages of both units according to the situation of head or tail label predictions.

4 Theoretical Analyses of DEPL

4.1 Rethinking Dense and Sparse Model for Imbalanced Text Classification

We analyze dense and sparse models from a gradient perspective for classification problems with skewed label distribution.

Preliminary: The predicted probability optimized by the binary cross entropy (BCE) loss is:

$$\mathcal{L}_{\text{BCE}} = -\sum_{l=1}^{L} y_l \log p_l + (1 - y_l) \log(1 - p_l)$$

The derivative of $\mathcal{L}_{\text{BCE}}$ w.r.t the logits s_l is:

$$\frac{\partial \mathcal{L}_{\text{BCE}}}{\partial s_l} = \begin{cases} p_l - 1 & \text{if } y_l = 1 \\ p_l & \text{otherwise} \end{cases} \tag{5}$$

Q1: *Why would sparse model with BOW feature benefit tail label prediction?*

Applying the chain rule to equation 5, the gradient of $\mathcal{L}_{\text{BCE}}$ w.r.t the document feature $\phi_n(\boldsymbol{x})$ is:

$$\frac{\partial \mathcal{L}_{\text{BCE}}(y_l, p_l)}{\partial \phi_n(\boldsymbol{x})} = \begin{cases} (p_l - 1)\boldsymbol{w}_l & \text{if } y_l = 1 \\ p_l \boldsymbol{w}_l & \text{otherwise} \end{cases}$$

By optimizing parameters θ of feature extractor, the document representation is encourage to move away from the negative label representation, that is:

$$\phi_n(\boldsymbol{x}; \theta') \leftarrow \phi_n(\boldsymbol{x}; \theta) - \eta p_l \boldsymbol{w}_l$$

where η is the learning rate.

For a dense model, the parameter θ of the feature extractor (such as BERT) is shared for all the data, so the optimization of the feature extractor is affected by the distribution of labels in the training data. Since a tail label appears more often as a negative target, the feature extractor is likely to under-represent the tail label information, making a tail label more difficult to be predicted. In comparison, the sparse feature like tf-idf is derived in an unsupervised manner from corpus statistics, which is independent of training label distribution. Therefore, the sparse feature may maintain better representation power to separate the tail labels.

Q2: *What is the advantage of a retrieval system on tail label prediction?*

In a typical classification system, labels are treated as indices whose embeddings are randomly initialized and learned from supervised signals. The gradients of $\mathcal{L}_{\text{BCE}}$ w.r.t the label feature is:

$$\frac{\partial \mathcal{L}_{\text{BCE}}(y_l, p_l)}{\partial \boldsymbol{w}_l} = \begin{cases} (p_l - 1)\phi_n(\boldsymbol{x}) & \text{if } y_l = 1 \\ p_l \phi_n(\boldsymbol{x}) & \text{otherwise} \end{cases}$$

The label embedding is updated by:

$$\boldsymbol{w}_l' = \boldsymbol{w}_l + \frac{\eta}{N_{\text{train}}} \sum_{i:y_{il}=1} (1 - p_{il})\phi_n(\boldsymbol{x}_i) - \frac{\eta}{N_{\text{train}}} \sum_{i:y_{il}=0} p_{il}\phi_n(\boldsymbol{x}_i)$$

As most of the instances are negative for a tail label, the update of tail label embedding is inundated with the aggregation of negative features, making it hard to encode distinctive feature reflecting its identity. Therefore, learning the tail label embedding from supervised signals alone can be distracting. Although previous works leverage negative sampling to alleviate the problem (Jiang et al., 2021; Chang et al., 2020), we argue that a fundamental solution is to inject the label information into the embedding. Our proposed retrieval system presents a natural way to incorporate label text for enhanced performance of tail label prediction.

4.2 Analysis on Performance Lower Bound

We will show the connection between DEPL and a sparse SVM classifier (for pseudo label extraction) by a performance lower bound. Specifically, DEPL outperforms a sparse model with high probability given that the selected keywords are important and the sparse classifier can separate the positive from the negative instances with non-trivial margin.

Notation: Let $\phi_t(\boldsymbol{x})$ be the normalized tf-idf feature vector of text with $\|\phi_t(\boldsymbol{x})\|_2 = 1$. The sparse label embeddings $\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_L\}$ satisfies $\|\boldsymbol{w}_l\|_2 \leq 1, w_{li} > 0$. In fact, label embeddings can be transformed to satisfy the condition without affecting the prediction rank. Let $\boldsymbol{z}_l$ be the top selected keywords from the sparse classifier, which is treated as the pseudo label. Define the sparse keyword embedding $\boldsymbol{v}_l$ with $v_{li} = w_{li}$ if i is an index of selected keywords and 0 otherwise.

In the following, we define the keyword importance and the classification error margin.

Definition 1. *For label l and $\delta \geq 0$, the sparse keyword embedding $\boldsymbol{v}_l$ is δ-bounded if $\langle \phi_t(\boldsymbol{x}), \boldsymbol{v}_l \rangle \geq \langle \phi_t(\boldsymbol{x}), \boldsymbol{w}_l \rangle - \delta$.*

Definition 2. *For two labels p and n, the error margin μ is the difference between the predicted scores $\mu(\phi(\boldsymbol{x}), \boldsymbol{w}_p, \boldsymbol{w}_n) = \langle \phi(\boldsymbol{x}), \boldsymbol{w}_p - \boldsymbol{w}_n \rangle$.*

The main theorem is stated as below:

Theorem 3. *Let $\phi_t(\boldsymbol{x})$ and $\phi_n(\boldsymbol{x})$ be the sparse and dense (dimension d) document feature, $\boldsymbol{w}_l$ be the label embedding and $\boldsymbol{z}_l$ be the δ-bounded keywords. For a positive label p, let $\mathbb{N}_p = \{n_1, \ldots, n_{M_p}\}$ be a set of negative labels ranked lower than p. The error margin $\epsilon_i = \mu(\phi_t(\boldsymbol{x}), \boldsymbol{w}_p, \boldsymbol{w}_{n_i})$ and $\epsilon = \min(\{\epsilon_1, \ldots, \epsilon_{M_p}\})$. An error $\mathcal{E}_i$ of the neural classifier occurs when*

$$\mu(\phi_n(\boldsymbol{x}), \phi_n(\boldsymbol{z}_p), \phi_n(\boldsymbol{z}_{n_i})) \leq 0 \qquad (6)$$

The probability of any such error happening satisfies

$$P(\mathcal{E}_1 \cup \ldots \cup \mathcal{E}_{M_p}) \leq 4M_p \exp(-\frac{(\epsilon-\delta)^2 d}{50})$$

When $(\epsilon - \delta) \geq 10\sqrt{\frac{\log M_p}{d}}$, the probability is bounded by $\frac{1}{M_p}$.

Discussion: An error event occurs when the sparse model makes a correct prediction but the neural model doesn't. If the neural model avoids all such errors, the performance should be at least as good as the sparse model, and Theorem 3 gives a bound of that probability.

The term δ measures the importance of selected keywords (smaller the more important), the error margin ϵ measures the difficulty the correctly predicted positive and negative pairs by the sparse model. The theorem states that the model achieves a lower bound performance as sparse classifier if the keywords are informative and error margin is non-trivial. Proofs are in section A.2 for interested readers and limitations are discussed in section 8.

5 Evaluation Design

Dataset	N_{train}	N_{test}	$\bar{L}_d$	L	$\lvert\mathbb{L}_{\text{tail}}\rvert$
EURLex-4K	15,539	3,809	5.30	3,956	2,413
AmazonCat-13K	1,186,239	306,782	5.04	13,330	3,936
Wiki10-31K	14,146	6,616	18.64	30,938	26,545
Wiki-500K	1,779,881	769,421	4.75	501,070	338,719

Table 1: Corpus Statistics: N_{train} and N_{test} are the number of training and testing instances respectively; $\bar{L}_d$ is the average number of labels per document, and L is the number of unique labels. $|\mathbb{L}_{\text{tail}}|$ is the number of tail labels with $1 \sim 9$ positive training instances.

5.1 Datasets

We conduct our experiments on 4 benchmark datasets: EURLex-4K, AmazonCat-13K, Wiki10-31K and Wiki-500K. The statistics of the datasets are shown in Table 1. An unstemmed version of EURLex-4K is obtained from the APLC-XLNet github[1] and the rest are from the Extreme classification Repository[2].

For comparative evaluation of methods in tail label prediction, we consider the subset of labels with $1 \sim 9$ positive training instances. Those tail-label subsets correspond to 63.48%, 29.53%, 88.65% and 67.60% of the total labels in the 4 datasets respectively. With mostly more than half of the labels as tail labels, the distributions are indeed highly skewed.

5.2 Tail Label Evaluation Metrics

Micro-averaged PSP@k: The PSP (Jain et al., 2016a) metric re-weights the score of each instance according to the label frequency:

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

where the propensity score $\text{prop}(\mathbf{p}_l)$ in the denominator gives higher weights to tail labels.

Since the micro-averaged metric gives an equal weight to the per-instance scores, it can still be dominated by the system's performance on the head labels but not the tail labels. As an alternative, we adopt a macro-averaged metric to evaluate tail label performance.

Macro-averaged F1@k: The macro-averaged metric (Yang and Liu, 1999) gives an equal weight to all the labels (we apply it to tail labels specifically). It is defined as the average of the label-specific $F1@k$ values, calculated based on a contingency table for each label, as shown in table 2. The precision, recall and $F1$ for a predicted ranked list of length k are computed as $\text{P} = \frac{\text{TP}}{\text{TP+FP}}, \text{R} = \frac{\text{TP}}{\text{TP+FN}}$, and $\text{F1} = 2\frac{P \cdot R}{P+R}$.

Table 2: Contingency table for label l.

	l is true label	l is not true label
l predicted	True Positive (TP_l)	False Positive (FP_l)
l not predicted	False Negative (FN_l)	True Negative (TN_l)

For micro-averaged PSP@k, we choose $k = 1, 3, 5$ as in previous works. For macro-averaged F1@k, we choose $k = 19$ for Wiki10-31K because it has an average of 18.64 labels and $k = 5$ for the rest datasets.

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

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

5.3 Baselines

For the tail label evaluation, our method is compared with the SOTA deep learning models including X-Transformer (Chang et al., 2020), XLNet-APLC (Ye et al., 2020), LightXML (Jiang et al., 2021), and AttentionXML (You et al., 2018). X-Transformer, LightXML, and XLNet-APLC employ pre-trained Transformers for document representation. We reproduced the results of single model (given in their implementation) predictions with BERT as the base model for LightXML, BERT-large for X-Transformer, XLNet for XLNet-APLC, and LSTM for AttentionXML. The AttentionXML utilizes label-word attention to generate label-aware document embeddings, while the other models generate fixed document embedding.

We use the SVM model with tf-idf feature as our choice of sparse classifier and BERT-base as our dense model for neural retrival and classification. Implementation details, more baselines and settings are discussed in appendix A.1.

6 Evaluation Results

Our experiments reveal the effectiveness of our model on the tail label prediction and we also include and discuss the performance on the overall prediction in appendix A.1.4.

6.1 Results in Tail Label Prediction

SVM on Tail Label Prediction The results evaluated with the F1 metric averaged on the tail labels are shown in figure 3. Surprisingly, a simple statistical SVM baseline achieves competitive results on the tail label predictions. We observe that SVM model can outperform most of the pretrained Transformer-based models on the tail label prediction, and outperform the AttentionXML on the Wiki10-31K dataset. This provides an empirical evidence for the robust performance of a sparse model on tail label prediction. As we analyzed in section 4, the SVM model utilizes the unsupervised statistical feature as document representation, which potentially suffers less from the data scarcity issue. The empirical result serves as an evidence for our theoretical analysis that the joint optimization of feature extractor and label embedding is difficult when data is limited.

Neural Classifier on Tail Label Prediction The neural classifiers include LightXML, X-Transformer, XLNet-APLC and AttentionXML. Specifically, the AttentionXML model leverages a label-word attention to calculate a label specific document representation. As we observe in figure 3, among the baseline models, the AttentionXML performs the best on the tail label predictions, beating the other baselines on 3 out of the 4 benchmark datasets. The superior performance could come from the local word and label matching which benefits the tail label prediction.

As mentioned in section 3, X-Transformer model ensembles a neural classifier and a SVM model by directly summing the prediction scores. Although X-Transformer outperforms SVM on the overall label prediction, it underperforms SVM on 3 out of 4 benchmark datasets. This shows that model performance on tail label is dragged down by the neural model prediction, and a simple ensemble does fully exploit the advantage of the sparse model. Compared with the X-Transformer, our model achieves better performance on both macro-F1 and micro-PSP metrics, showing the advantage of leveraging the retrieval of augmented label descriptions rather than a pure ensemble.

DEPL Performance On the 3 smaller scale benchmark datasets, EURLex-4K, AmazonCat-13K and Wiki10-31K, our model directly ranks all the labels. On the large Wiki-500K dataset, our model leverages the prediction of cluster-based algorithm in X-Transformer and replaces the reranker with our retrieval model.

Our proposed models perform the best on the Macro-F1 metric with the DEPL model consistently and significantly showing the best performance on all the benchmark datasets. A macro t-test (Yang and Liu, 1999) is conducted to justify the significance of improvement over the SVM and previous best neural model. The significant performance gains over the SVM model shows that our retrieval framework can outperform the sparse model which serves as label keywords extractor. We attribute the success of model on tail label prediction to the retrieval module that focuses on the semantic matching between the document and label text. The DEPL performs better than the DEPL+cls as it is less affected by the large amount of training instances for head labels and thus more biased on the tail label prediction.

According to the evaluation with the PSP metric shown in table 3, it also confirms that our proposed models DEPL and DEPL+cls improves over the previous SOTA neural models on all the benchmark

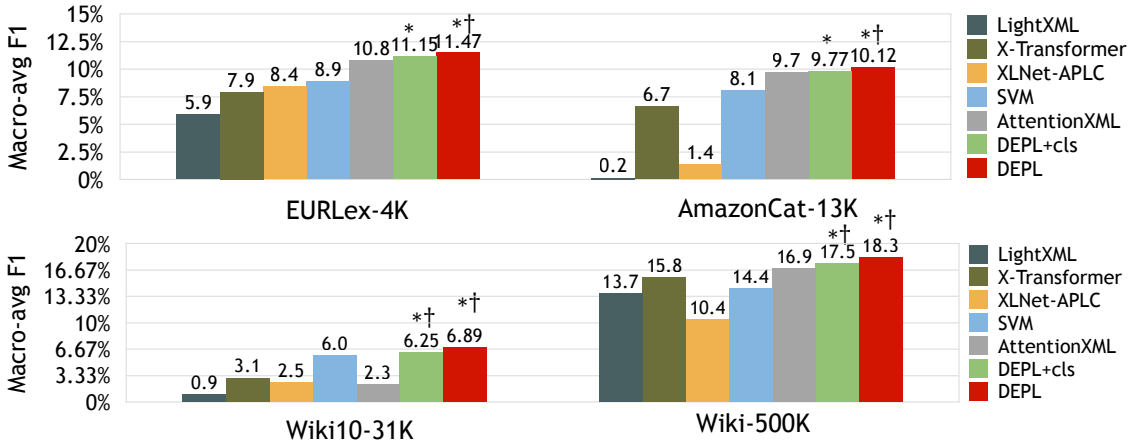

Figure 3: Tail-label prediction results in $F1@k$ on the labels with $1 \sim 9$ positive training instances, with $k = 19$ for the Wiki10-31K dataset and $k = 5$ for the rest. $*$ and † indicates the macro t-test is significant ($p < 0.05$) over SVM and previous best neural model respectively.

Table 3: Tail label prediction results of methods in $PSP@k$, with $*$ indicating significant improvement ($p < 0.05$) over the previous best model on the micro sign test.

	EURLex-4K			Wiki10-31K			AmazonCat-13K			Wiki-500K		
Methods	PSP@1	PSP@3	PSP@5	PSP@1	PSP@3	PSP@5	PSP@1	PSP@3	PSP@5	PSP@1	PSP@3	PSP@5
X-Transformer	37.85	47.05	51.81	13.52	14.62	15.63	51.42	66.14	75.57	31.20	36.78	40.21
XLNet-APLC	42.21	49.83	52.88	14.43	15.38	16.47	52.55	65.11	71.36	29.73	30.26	30.59
LightXML	40.54	47.56	50.50	14.09	14.87	15.52	50.70	63.14	70.13	31.01	37.10	39.28
AttentionXML	44.20	50.85	53.87	14.49	15.65	16.54	53.94	68.48	76.43	30.05	37.31	41.74
SVM	39.18	48.31	53.37	11.84	14.00	15.81	51.83	65.41	72.82	32.12	32.75	35.20
DEPL	**45.60***	52.28*	53.52	**17.20***	**16.90***	**16.95**	**55.94***	**70.01***	**76.87***	32.07	**40.60***	**43.74***
DEPL+cls	44.60	**52.74***	**54.64**	16.73*	16.84*	16.67	55.21*	69.73*	75.94	**32.18**	39.89*	41.46

Table 4: Examples of SVM generated keywords from Wiki10-31K. The classifier is trained with only 1 positive training instance per label. The top 20 keywords are shown. with meaningful words highlighted in red manually.

Label Text	**#training instance**	**Top Keywords**
phase4	1	trials clinical protection personal directive processed data trial drug phase eu processing patients sponsor controller legislation regulation art investigator study
ensemble	1	boosting kurtz ferrell weak algorithms learners misclassified learner kearns ensemble charges bioterrorism indictment doj indict cae correlated 2004 reweighted boost
kakuro	1	nikoli kakuro puzzles crossword clues entries entry values sums cells cross digits dell solvers racehorse guineas aa3aa digit clue kaji

datasets, with $*$ indicates significant improvement ($p < 0.05$) over the previous best model on the micro sign test (Yang and Liu, 1999). The Wiki10-31K dataset has the most skewed distribution as the most frequent label covers more than 85% of the training instances, resulting in a low PSP score. Since DEPL relies on the semantic matching between the document and label text, it is less affected by the dominating training pairs, and thus the PSP@1, PSP@3 beats the SOTA models by a larger margin. The DEPL+cls achieves worse performance on this dataset, because the classification counterpart of the model would benefit more on the head label predictions and tend to rank the head labels at the top.

Metric Comparison Although the PSP metric gives higher weight to the tail labels, it is a micro-averaged metric over the scores of each instance, which can still be affected by the performance on the more common categories that cover most of the instances. For example, SVM model doesn't stand out under the PSP metric, which has lower overall label performance. Since the F1 metric is calculated specifically on the set of tail labels, we argue that it provides a more accurate and fine-

grained evaluation on tail label prediction, which better reveals the success of XMTC models on predicting rare categories.

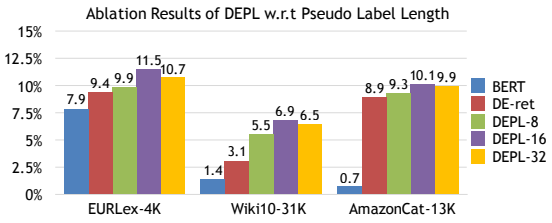

Figure 4: The ablation-test results of DEPL in Macro-averaged F1@k metric with varying length of pseudo label descriptions.

6.2 Ablation on Generated Pseudo Label

Table 4 shows examples of the SVM generated keywords trained on the Wiki10-31K dataset for labels with only 1 training example. We manually highlight the meaningful terms related to the label meaning. For example, the label name *phase4* is ambiguous, whose meaning needs to be inferred from the corresponding document. From the keywords *trial, clinical, drug, etc*, we deduce that the topic is about medical testing phase. In another example, *kakuro* is a Japanese logic puzzle known as a mathematical crossword and the game play involves in adding number in the cells. Generating a description for *kakuro* requires the background knowledge, but the keywords automatically learned from the sparse classifier provide the key concepts. Although not all the keywords can provide rich semantics to complement the original label name, they may serve as a context for the label to make it more distinguishable from others.

In figure 4, we conduct an ablation test on the length of the pseudo label and the performance is measured by Macro-avg F1@k. The BERT classifier is included as a baseline with no label text information. As we observe that the longer description of length 16 performs the better, but when length is 32, the performance doesn't increase as the text may become noisy with more unrelated keywords.

The DE-ret model is a pure retrieval baseline (avg length 3) with only the label name. While it achieves good performance on the EURLex-4K and AmazonCat-13K datasets, it still performs poorly on the Wiki10-31K dataset. This shows that generating the keywords from the sparse classifier can enhance the text quality. Furthermore, the generated text allows DEPL to use the semantic information of the label keywords, which is ignored in the SVM model. This could be another reason why our model performs better than the SVM baseline on the Wiki10-31K dataset.

7 Conclusion

In this paper, we propose a novel neural retrieval framework (DEPL) for the open challenge of tail-label prediction in XMTC. By formulating the problem as to capture the semantic mapping between input documents and system-enhanced label descriptions, DEPL combines the strengths of neural embedding based retrieval and the effectiveness of a large-marge BoW classifier in generating informative label descriptions under severe data sparse conditions. Our extensive experiments on very large benchmark datasets show significant performance improvements by DEPL over strong baseline methods, especially in tail label description.

8 Limitations

Our paper mainly focuses on the evaluation and improvement over the pretrained Transformer-based models such as X-Transformer, LightXML and APLC-XLNet by leveraging the recent advances in dense retrieval with BERT model. However, there are other works such as proposing reranking losses (Wei et al., 2021), regularization (Babbar and Schölkopf, 2019a) with other architectures are not included for comparison.

As pointed out by the reviewers, the performance bound analysis in section 4 adopts a strong assumption that the neural embeddings are random matrices. This could be very different in real application because the random matrices do not encode any semantic information. We acknowledge this limitation and provide more references on that. We rely on the mathematical tool based on random matrix theory, namely the *Johnson-Lindenstrauss* (JL) lemma. This tool was also adopted by Luan et al. (2020) under information retrieval setting, which provides the connection between dense and sparse retrievers. The bound is on its loose end because embeddings from BERT are more meaningful than random matrices (also verified from their empirical study). In our work, we study use the JL lemma to connect sparse and dense classifiers. The bound is reasonable considering that it is on its loose end, but, still, there is no guarantee when applied with real BERT embeddings.

References

Rohit Babbar and Bernhard Schölkopf. 2017. Dismec: Distributed sparse machines for extreme multi-label classification. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 721–729.

Rohit Babbar and Bernhard Schölkopf. 2019a. Data scarcity, robustness and extreme multi-label classification. *Machine Learning*, 108(8):1329–1351.

Rohit Babbar and Bernhard Schölkopf. 2019b. Data scarcity, robustness and extreme multi-label classification. *Machine Learning*, 108(8):1329–1351.

Shai Ben-David, Nadav Eiron, and Hans Ulrich Simon. 2002. Limitations of learning via embeddings in euclidean half spaces. *Journal of Machine Learning Research*, 3(Nov):441–461.

Duo Chai, Wei Wu, Qinghong Han, Fei Wu, and Jiwei Li. 2020. Description based text classification with reinforcement learning. In *International Conference on Machine Learning*, pages 1371–1382. PMLR.

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.

Kunal Dahiya, Ananye Agarwal, Deepak Saini, K Gururaj, Jian Jiao, Amit Singh, Sumeet Agarwal, Purushottam Kar, and Manik Varma. 2021. Siamesexml: Siamese networks meet extreme classifiers with 100m labels. In *International Conference on Machine Learning*, pages 2330–2340. PMLR.

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*.

Luyu Gao and Jamie Callan. 2021. Unsupervised corpus aware language model pre-training for dense passage retrieval. *arXiv preprint arXiv:2108.05540*.

Siddarth Gopal, Yiming Yang, Bing Bai, and Alexandru Niculescu-Mizil. 2012. Bayesian models for large-scale hierarchical classification.

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.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Himanshu Jain, Venkatesh Balasubramanian, Bhanu Chunduri, and Manik Varma. 2019. Slice: Scalable linear extreme classifiers trained on 100 million labels for related searches. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 528–536.

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

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

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*.

William B Johnson and Joram Lindenstrauss. 1984. Extensions of lipschitz mappings into a hilbert space 26. *Contemporary mathematics*, 26.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

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.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2020. Sparse, dense, and attentional representations for text retrieval. *arXiv preprint arXiv:2005.00181*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

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*, pages 993–1002.

Tong Wei, Wei-Wei Tu, Yu-Feng Li, and Guo-Ping Yang. 2021. Towards robust prediction on tail labels. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1812–1820.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *arXiv preprint arXiv:2007.00808*.

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.

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.

Ian EH Yen, Xiangru Huang, Wei Dai, Pradeep Ravikumar, Inderjit Dhillon, and Eric Xing. 2017. Ppdsparse: A parallel primal-dual sparse method for extreme classification. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 545–553.

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*.

A Appendix

A.1 Experiments

A.1.1 All-label Evaluation Metric

We introduce the micro-averaged $P@k$ as the metric for all-label prediction. Given a ranked list of the predicted labels for each test document, the **micro-averaged P@k** is:

$$P@k = \frac{1}{k} \sum_{i=1}^{k} \mathbb{1}_{\boldsymbol{y}_i^+}(p_i) \tag{7}$$

where p_i is the i-th label in the list $\boldsymbol{p}$ and $\mathbb{1}_{\boldsymbol{y}_i^+}$ is the indicator function.

A.1.2 More Baseline

For the overall prediction of all labels, we also include the baselines of sparse classifiers: DisMEC (Babbar and Schölkopf, 2017), PfastreXML (Jain et al., 2016b), Parabel (Prabhu et al., 2018), Bonsai (Khandagale et al., 2019), and we use the published results for comparison. We provide an implementation of linear SVM model with our extracted tf-idf features as another sparse baseline, and a BERT-base classifier as another dense classifier (used to initialize DEPL).

A.1.3 Implementation Details

For the sparse model, since the public available BoW feature doesn't have a vocabulary dictionary, we generate the tf-idf feature by ourselves. We tokenize and lemmatize the raw text with the spaCy (Honnibal and Montani, 2017) library and extract the tf-idf feature with the Sklearn (Pedregosa et al., 2011) library, with unigram whose df count is $>= 2$ and df frequency $<= 70\%$ of the total documents.

We use the BERT model as the contextualize function for our retrieval model, which is initialized with a pretrained dense classifier. Specifically, we fine-tune a 12 layer BERT-base model with different learning rates for the BERT encoder, BERT pooler and the classifier. The learning rates are $(1e-5,\ 1e-4,\ 1e-3)$ for Wiki10-31K and $(5e-5,\ 1e-4,\ 2e-3)$ for the rest datasets. For the negative sampling, we sample batch of 500 instances for Wiki10-31K, and 300 for EURLex-4K and AmazonCat-13K. For Wiki-500K dataset, we leverage the cluster-based algorithm in X-Transformer, and perform label re-ranking using our DEPL model to replace the linear model in X-Transformer. We use a negative batch size of 500 for to train the re-ranker.

We include 10 hard negatives predicted by the SVM model for each instances. We used learning rate $1e-5$ for fine-tuning the BERT of our retrieval model and $1e-4$ for the pooler and label embeddings. For the pseudo label descriptions, we concatenate the provided label description with the generated the top 20 keywords. The final length is truncated up to 32 tokens after BERT tokenization. We use length 16 of pseudo label description as the default setting for DEPL.

A.1.4 Results in All-label Prediction

The performance of our models evaluated on the all-label prediction by the micro-averaged P@k metric is reported in table 5. Our model is compared against the SOTA sparse and dense classifiers. DEPL+c achieves the best or second best performance on all the 4 benchmark datasets, achieving comparable results to the previous best SOTA models.

	EURLex-4K			Wiki10-31K			AmazonCat-13K			Wiki-500K		
Methods	P@1	P@3	P@5	P@1	P@3	P@5	P@1	P@3	P@5	P@1	P@3	P@5
DisMEC	83.21	70.39	58.73	84.13	74.72	65.94	93.81	79.08	64.06	70.21	50.57	39.68
PfastreXML	73.14	60.16	50.54	83.57	68.61	59.10	91.75	77.97	63.68	56.25	37.32	28.16
eXtremeText	79.17	66.80	56.09	83.66	73.28	64.51	92.50	78.12	63.51	65.17	46.32	36.15
Parabel	82.12	68.91	57.89	84.19	72.46	63.37	93.02	79.14	64.51	68.70	49.57	38.64
Bonsai	82.30	69.55	58.35	84.52	73.76	64.69	92.98	79.13	64.46	69.26	46.72	36.46
AttentionXML	85.12	72.80	61.01	86.46	77.22	67.98	95.53	82.03	67.00	75.20	56.42	44.10
X-Transformer	85.46	72.87	60.79	87.12	76.51	66.69	95.75	**82.46**	67.22	75.28	55.46	42.75
XLNet-APLC	**86.83**	**74.34**	61.94	**88.99**	**78.79**	**69.79**	94.56	79.78	64.59	72.95	51.23	38.64
LightXML	86.12	73.87	61.67	87.39	77.02	68.21	94.61	79.83	64.45	75.96	56.55	44.22
SVM	83.44	70.62	59.08	84.61	74.64	65.89	93.20	78.89	64.14	69.92	49.35	38.8
DEPL	85.38	71.86	59.91	84.63	74.80	65.96	94.86	80.85	64.55	74.69	55.72	42.71
DEPL+c	86.43	73.77	**62.19**	88.57	78.04	68.75	**96.16**	82.23	**67.65**	**76.83**	**57.15**	**45.07**

Table 5: The all-label prediction results of representative classification systems evaluated in the micro-avg P@k metric. The bold phase and underscore highlight the best and second best model performance.

We argue that though our models perform significantly better on the tail label prediction, the improvement is not announced in the overall label prediction. One of the problem is on the choice of evaluation metric: the micro-averaged precision metric is averaged over instances and can be dominated by the common categories with more test instances. Therefore, the metric is incapable of reflecting the tail label performance. We want to emphasis that over $26,545$ (88.65%) labels in the Wiki10-31K dataset belong to the tail labels with less than 10 training instances, constituting a majority of the label space. The overall classification precision (P@k) only reflects a part of the success of a classification system, and the tail label evaluation is yet another part. The results also shows while our model improves on the tail label prediction, the overall label prediction comparable to the other dense and sparse SOTA models.

When our model is compared on the Wiki-500K dataset, our backbone is the same as X-Transformer. DEPL achieve on par performance with Wiki-500k showing that the quality of overall ranking is similar. However, the DEPL+c achieves better performance, demonstrating the enhanced performance by combining retrieval with classification.

By comparing the DEPL+c and its retrieval-based counterpart DEPL , we uncover a trade-off between the head label and tail label prediction. We observe that the DEPL outperforms the DEPL+c on the tail label prediction, but not on the all-label prediction. This shows that incorporating a classifier with label embeddings trained from supervised signal can boost performance on a high data regime. The dense classifier could learn more expressive label representation from the frequent co-occurrence of document and label pairs when the training instances are abundant, while the retrieval system is better at matching the semantic of document and label texts when data is scarce. Each of the modules captures a certain aspect of the data heuristic for text classification and a combination of them by sharing the BERT encoder yields better performance.

Lastly, the sparse classifiers generally underperform the neural models and are comparable to our implement of SVM. We observe that DEPL can outperform the sparse models, which agrees with our theoretical analysis. Although the pseudo labels are extracted from the SVM classifier, the neural retrieval model can additionally leverage the keyword semantic information and correlation of them, which is ignored in the SVM classifier. The pseudo label descriptions encode both the term importance and key semantics of labels.

A.1.5 More Ablation Tests

Model Pre-training We fine-tune our retrieval model on a pre-trained neural classifier (BERT) and table 6 shows that without using the pre-trained model, there is a significant drop in the precision and PSP metrics.

Negative Sampling We used the top negative predictions by the SVM model as the choice of hard negative labels. By default, we use 10 hard negatives for each instance in the batch. In table 6, we

Table 6: Ablation-test results of DEPL under different training conditions.

Methods	P@1	P@3	P@5	PSP@1	PSP@3	PSP@5
EUR-Lex						
DE-ret	-1.34	-1.18	-1.16	-3.2	-2.11	+0.18
w/o pre-train	-6.81	-6.72	-6.16	-6.87	-7.09	-5.87
w/o neg	-2.52	-2.63	-2.2	-3.59	-3.66	-1.9
5 hard negative	-1.55	-1.19	-1.12	-3.57	-2.98	-1.32
Wiki10-31K						
DE-ret	-3.40	-7.71	-10.74	-7.63	-5.09	-1.20
w/o pre-train	-4.81	-8.66	-12.1	-2.46	-2.00	-1.92
w/o hard negative	-2.01	-5.89	-4.93	-1.15	-1.17	-1.37
5 hard negative	-0.84	-3.03	-4.16	-1.22	-0.99	-0.92

Table 7: Ablation-test results of DEPL with `CLS` and mean-pooling.

Methods	PSP@1	PSP@3	PSP@5
	EUR-Lex		
DE-ret	44.87	52.17	53.40
DEPL with cls	42.32	47.26	47.53
DEPL with mean-pooling	45.60	52.28	53.52
	Wiki10-31K		
DE-ret	16.71	15.76	16.35
DEPL with cls	14.71	14.58	15.33
DEPL with mean-pooling	17.20	16.90	16.95

observe a performance drop when no hard negatives or only 5 hard negatives are used for training.

CLS vs. Mean-pooling Table 7 shows an ablation test for the design of label description encoder. We observe that using a mean-pooling over the last layer of label keyword embeddings outperforms that using the `CLS` embedding by a large margin. This could be because the label keywords are not natural language the optimization using `CLS` embedding is more difficult.

A.2 Proof

We include the assumptions and proofs of Theorem 3.

Assumptions Similar to Luan et al. (2020), we treat neural embedding as fixed dense vector $\boldsymbol{E} \in \mathbb{R}^{d \times v}$ with each entry sampled from a random Gaussian $N(0, d^{-1/2})$. $\phi_n(\boldsymbol{x}) = \boldsymbol{E}\phi_t(\boldsymbol{x})$ is weighted average of word embeddings by the sparse vector representation of text. According to the *Johnson-Lindenstrauss (JL) Lemma* (Johnson and Lindenstrauss, 1984; Ben-David et al., 2002), even if the entries of $\boldsymbol{E}$ are sampled from a random normal distribution, with large probability, $\langle \phi_t(\boldsymbol{x}), \boldsymbol{v} \rangle$ and $\langle \boldsymbol{E}\phi_t(\boldsymbol{x}), \boldsymbol{E}\boldsymbol{v} \rangle$ are close.

Lemma 4. *Let $\boldsymbol{v}$ be the δ-bounded keyword-selected label embedding of $\boldsymbol{w}$. For two labels p, n, the error margins satisfy:*

$$|\mu(\phi_t(\boldsymbol{x}), \boldsymbol{w}_p, \boldsymbol{w}_n) - \mu(\phi_t(\boldsymbol{x}), \boldsymbol{v}_p, \boldsymbol{v}_n)| \leq \delta$$

Proof. By the definition of δ-bounded keywords,

$$\langle \phi_t(\boldsymbol{x}), \boldsymbol{w}_p \rangle - \delta \leq \langle \phi_t(\boldsymbol{x}), \boldsymbol{v}_p \rangle \leq \langle \phi_t(\boldsymbol{x}), \boldsymbol{w}_p \rangle \tag{8}$$

$$-\langle \phi_t(\boldsymbol{x}), \boldsymbol{w}_n \rangle \leq -\langle \phi_t(\boldsymbol{x}), \boldsymbol{v}_n \rangle \leq -\langle \phi_t(\boldsymbol{x}), \boldsymbol{w}_n \rangle + \delta \tag{9}$$

Adding equation 8 and equation 9 finishes the proof:

$$\langle \phi_t(\boldsymbol{x}), \boldsymbol{w}_p - \boldsymbol{w}_n \rangle - \delta \leq \langle \phi_t(\boldsymbol{x}), \boldsymbol{v}_p - \boldsymbol{v}_n \rangle \leq \langle \phi_t(\boldsymbol{x}), \boldsymbol{w}_p - \boldsymbol{w}_n \rangle + \delta \tag{10}$$

□

Lemma 5. *Let $\phi_t(\boldsymbol{x})$ and $\phi_n(\boldsymbol{x})$ be the sparse and dense (dimension d) document feature, $\boldsymbol{w}_l$ be the label embedding and $\boldsymbol{z}_l$ be the δ-bounded keywords. Let p be a positive label and n be a negative label ranked below p be the sparse classifier. The error margin is $\epsilon = \mu(\phi_t(\boldsymbol{x}), \boldsymbol{w}_p, \boldsymbol{w}_n)$. An error $\mathcal{E}$ of neural classification occurs when $\mu(\phi_n(\boldsymbol{x}), \phi_n(\boldsymbol{z}_p), \phi_n(\boldsymbol{z}_n)) \leq 0$. The probability $P(\mathcal{E}) \leq 4\exp(-\frac{(\epsilon-\delta)^2 d}{50})$.*

Proof. By the JL Lemma (Ben-David et al., 2002): For any two vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^v$, let $\boldsymbol{E} \in \mathbb{R}^{d \times v}$ be a random matrix such that the entries are sampled from a random Gaussian. Then for every constant $\gamma > 0$:

$$P\left(|\langle \boldsymbol{E}\boldsymbol{a}, \boldsymbol{E}\boldsymbol{b} \rangle - \langle \boldsymbol{a}, \boldsymbol{b} \rangle| \geq \frac{\gamma}{2}\left(\|\boldsymbol{a}\|^2 + \|\boldsymbol{b}\|^2\right)\right) \leq 4\exp\left(-\frac{\gamma^2 d}{8}\right) \tag{11}$$

Let $\gamma = \frac{2}{5}(\epsilon - \delta)$, $\boldsymbol{a} = \phi_t(\boldsymbol{x})$ and $\boldsymbol{b} = \boldsymbol{v}_p - \boldsymbol{v}_n$. Since $\|\boldsymbol{a}\|_2 = 1$ and $\|\boldsymbol{b}\|_2 \leq (\|\boldsymbol{v}_p\|_2 + \|\boldsymbol{v}_n\|_2)^2 \leq 4$, the JL Lemma gives

$$P\left(|\langle \boldsymbol{E}\phi_t(\boldsymbol{x}), \boldsymbol{E}(\boldsymbol{v}_p - \boldsymbol{v}_n) \rangle - \langle \phi_t(\boldsymbol{x}), \boldsymbol{v}_p - \boldsymbol{v}_n \rangle| \geq \epsilon - \delta\right) \tag{12}$$

$$\leq 4\exp(-\frac{(\epsilon-\delta)^2 d}{50}) \tag{13}$$

To complete the proof, we need to show $P(\mathcal{E}) \leq Eq.12$:

$$\mathcal{E} \implies |\langle \boldsymbol{E}\phi_t(\boldsymbol{x}), \boldsymbol{E}(\boldsymbol{v}_p - \boldsymbol{v}_n) \rangle - \langle \phi_t(\boldsymbol{x}), \boldsymbol{w}_p - \boldsymbol{w}_n \rangle| \geq \epsilon \tag{14}$$

$$\implies |\langle \boldsymbol{E}\phi_t(\boldsymbol{x}), \boldsymbol{E}(\boldsymbol{v}_p - \boldsymbol{v}_n) \rangle - \langle \phi_t(\boldsymbol{x}), \boldsymbol{v}_p - \boldsymbol{v}_n \rangle| \geq \epsilon - \delta \tag{15}$$

where the equation 15 is derived by Lemma 4:

$$|\langle \boldsymbol{E}\phi_t(\boldsymbol{x}), \boldsymbol{E}(\boldsymbol{v}_p - \boldsymbol{v}_n) \rangle - \langle \phi_t(\boldsymbol{x}), \boldsymbol{v}_p - \boldsymbol{v}_n \rangle| \tag{16}$$

$$\geq |\langle \boldsymbol{E}\phi_t(\boldsymbol{x}), \boldsymbol{E}(\boldsymbol{v}_p - \boldsymbol{v}_n) \rangle - \langle \phi_t(\boldsymbol{x}), \boldsymbol{w}_p - \boldsymbol{w}_n \rangle| - \tag{17}$$

$$|\langle \phi_t(\boldsymbol{x}), \boldsymbol{w}_p - \boldsymbol{w}_n \rangle - \langle \phi_t(\boldsymbol{x}), \boldsymbol{v}_p - \boldsymbol{v}_n \rangle| \tag{18}$$

$$\geq \epsilon - \delta \tag{19}$$

Therefore $P(\mathcal{E}) \leq Eq.12$, which completes the proof. □

Proof of **Theorem** 3

Proof. The Lemma 2 shows that

$$P(\mathcal{E}_i) \leq 4\exp(-\frac{(\epsilon_i - \delta)^2 d}{50}) \leq 4\exp(-\frac{(\epsilon - \delta)^2 d}{50}) \tag{20}$$

By an union bound on the error events $\{\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_{M_p}\}$,

$$P(\mathcal{E}_1 \cup \ldots \cup \mathcal{E}_{M_p}) \leq \sum_{i=1}^{M_p} 4\exp(-\frac{(\epsilon_i - \delta)^2 d}{50}) \tag{21}$$

$$= 4M_p \exp(-\frac{(\epsilon - \delta)^2 d}{50}) \tag{22}$$

□

When $(\epsilon - \delta)^2 \geq 10\sqrt{\frac{\log M_p}{d}}$, we have $\exp(-\frac{(\epsilon-\delta)^2 d}{50}) \leq \frac{1}{4{M_p}^2}$ and therefore $P(\mathcal{E}_1 \cup \ldots \cup \mathcal{E}_{M_p}) \leq \frac{1}{M_p}$.