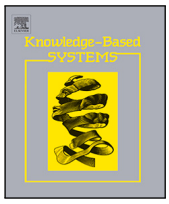
Knowledge-Based Systems 320 (2025) 113607

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys



Contrastive Transformer Network for Long Tail Classification

Johannes Melsbach *, Frederic Haase, Sven Stahlmann, Stefan Hirschmeier, Detlef Schoder

Cologne Institute for Information Systems University of Cologne, Pohligstr. 1, 50969, Cologne, NRW, Germany

ARTICLE INFO

Keywords:
Multi-label text classification
Long-tail classification
Contrastive learning
Natural Language Processing

ABSTRACT

In the context of big data, multi-label text classification presents considerable challenges, most notably the long-tail problem, wherein a small number of labels account for the majority of instances, while the vast majority of labels occur only rarely. This imbalance creates a critical bias in classification models, leading to suboptimal performance on tail labels that significantly impacts applications such as recommender systems and search engines. We present CTN-LT (Contrastive Transformer Network for Long Tail Classification), a novel dual-encoder architecture that combines adapted loss functions, contrastive learning and reframes the multi-label text classification as a semantic similarity task to specifically enhance tail label performance. Our method achieves state-of-the-art performance on tail labels while maintaining competitive performance on head labels across multiple benchmark datasets. The model demonstrates superior few-shot and zero-shot capabilities, making it particularly valuable for dynamic environments where new categories frequently emerge. We release our code at https://github.com/jmelsbach/CTN-LT.

1. Introduction

Numerous application domains generate massive volumes of textual content daily, spanning Web 2.0 platforms such as wikis [1,2], e-commerce platforms with extensive product descriptions [3,4] and information retrieval systems [5]. The increase in data production drives the growing need of text classification and tagging, which are essential for categorizing, searching, retrieving and recommending textual content effectively [6]. The number of labels for some scenarios can grow extremely large spanning from thousands to millions of labels [7]. Appropriately, these classification problems are called extreme multi-label text classification (XML) in the literature.

The overarching challenge within XML lies in the distribution of the frequency of these labels, which typically follows a long-tailed pattern as shown in Fig. 1. This leads to the long-tail problem: a small subset of labels (head labels) is associated with a majority of the samples, while a vast number of labels (tail labels) are linked to a small number of samples [8]. This imbalance results in machine learning systems that are biased towards predicting head labels more frequently than tail labels [9], thereby compromising the performance for tail labels due to insufficient training samples. Thus, addressing the long-tail problem by improving predictions on tail labels can ultimately contribute to better outcomes in real-world applications.

Addressing the performance challenges of tail labels remains a critical objective. Existing literature has made significant strides in addressing these challenges through the application of pre-trained language models [10–12], the development of innovative data sampling strategies [13], and the refinement of loss functions to mitigate category imbalance [14]. Furthermore, novel approaches such as decoupled training [15], dual-branch training [7], and contrastive learning methods [16] have been introduced to enhance the performance on tail labels.

However, despite these advancements in multi-label text classification, the improvements in tail label classification have not kept pace with the strong improvements on head labels that have been achieved in recent years. [17] note that the research community has primarily focused on improving overall accuracy, which often heavily favors head labels. One commonly used benchmark for evaluating multi-label classification models, the Extreme Classification Repository [18], reveals a recurring performance gap. Metrics that factor in propensity scores – a method used to adjust for label frequency – still show significantly lower performance for tail labels compared to head labels.

To tackle this long-tail performance gap, our study proposes a Contrastive Transformer Network for Long Tail Classification (CTN-LT). Our approach is based on three design choices: First, we reformulate the multi-label classification task as a semantic similarity problem by jointly embedding documents and textual label descriptions into a shared vector space. Here, embeddings refer to vector representations that capture semantic meaning, enabling direct comparison

* Corresponding author.
E-mail addresses: melsbach@wim.uni-koeln.de (J. Melsbach), haase@wim.uni-koeln.de (F. Haase), stahlmann@wim.uni-koeln.de (S. Stahlmann), hirschmeier@wim.uni-koeln.de (S. Hirschmeier), schoder@wim.uni-koeln.de (D. Schoder).

https://doi.org/10.1016/j.knosys.2025.113607
Received 17 November 2024; Received in revised form 8 April 2025; Accepted 19 April 2025
Available online 6 May 2025

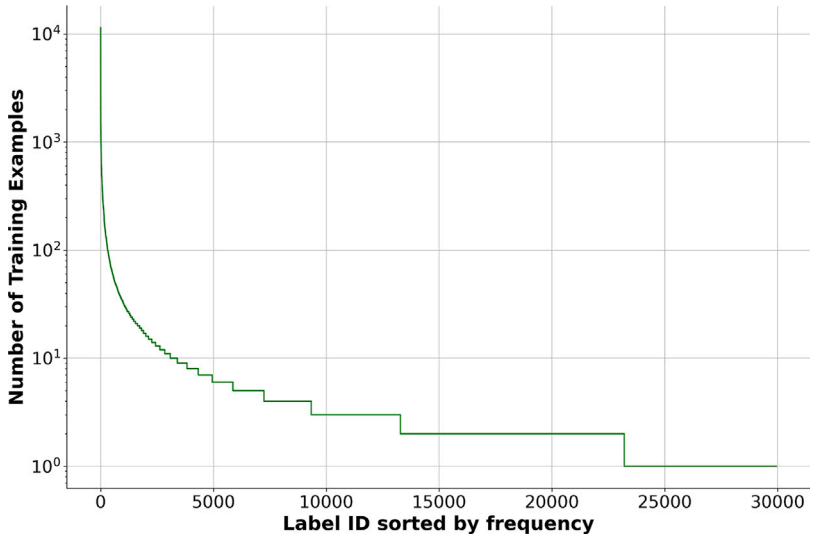


Fig. 1. Label frequency (log-scale) of the Wiki10-31K training dataset. The highly imbalanced nature, where only about 10% of labels appear more than 10 times, highlights the long-tail challenge in extreme multi-label classification.

between documents and labels. Second, we introduce a novel combination of two adapted loss functions—a masked Binary Cross Entropy loss (mBCE) and a adapted Cross Entropy loss (CE) that dynamically constructs softmax distributions for each positive label to improve tail label performance. Third, we incorporate contrastive learning with in-batch negative sampling through a dual-encoder architecture based on pre-trained transformers. This contrastive setup efficiently trains the model to distinguish relevant from irrelevant labels within each batch, further improving performance on infrequent and even zero-shot labels. Taken together, these design choices lead to new state-of-the-art performance on several benchmark datasets, particularly in propensity adjusted metrics that emphasize tail label performance. While our approach slightly compromises the performance for head labels, the trade-off results in a significant improvement for tail labels, thus addressing a critical challenge in the field. Additionally, our model achieves superior results on zero-shot labels, demonstrating its ability to generalize effectively by even predicting labels for which no training examples are available, which is common in real-world scenarios [19].

The paper is organized as follows: we begin with an overview of prior work on multi-label classification and contrastive learning, followed by a description of our method that includes a detailed description of our model architecture. The subsequent section describes our data and the evaluation metrics used, presents the results of our experiments, and discusses our findings. Finally, we conclude by summarizing our contributions and suggesting directions for future research.

2. Related work

2.1. Multi-label classification

Multi-label text classification is the task of assigning one or more labels from a predefined set to an input document. In the literature, problems where the set of possible labels is very large are denoted as Extreme Multi-label Classification (XMC). This type of classification problem presents several key challenges. The large label space in XMC problems can lead to significant computational difficulties, both in terms of computational and memory requirements [12]. Additionally, XMC problems often exhibit a long-tail distribution, where a few labels appear very frequently, while the majority of labels occur rarely. This distribution typically results in model bias, where models tend to perform worse on the tail labels due to the lack of training data, making it challenging for models to learn effective representations for these rare categories. To address these challenges, researchers have developed various approaches that can be categorized into label embedding techniques, tree-based methods, and more recently deep learning methods. Label embedding techniques, such as AnnexML [20] and SLEEC [21], learn low-dimensional embeddings for labels to capture label correlations. Tree-based methods, including FastXML [22], Parabel [23], and Bonsai [24], use tree structures to partition the label space, enabling efficient training and prediction. Current state-of-the-art models make use of deep learning approaches, incorporating pre-trained language models and novel architectures to improve XMC performance [10,25,26]. Some studies have focused on the challenges associated with long-tailed label distributions. Some researchers have examined the role of tail labels, proposing techniques like label trimming [27] and transferring knowledge from head labels to tail labels [8]. Data augmentation methods, such as TailMix, have also been explored to address data scarcity in tail labels [28]. Another body of work concentrates on developing specialized loss functions, including propensity-scored losses [6], distribution-balanced losses [14], and gradient-balanced loss, designed to adaptively suppress negative gradient accumulation [7]. Additionally, two-stage approaches that employ separate classifiers for head and tail labels have been proposed to improve performance across the label spectrum [29]. Despite these advancements, improving performance on tail labels while maintaining strong results on head labels remains an active area of research in multi-label text classification. The ongoing efforts in this field aim to develop more robust and efficient models capable of handling the complexities of extreme multi-label classification tasks.

2.2. Contrastive learning

Contrastive Learning (CL) is a paradigm widely used in the domains of Computer Vision [30–32] as well as Natural Language Processing (NLP) [33,34]. The purpose of CL is to learn meaningful representations of data by comparing similar (positive) and dissimilar (negative) pairs of datapoints. The goal is to maximize the similarity between positive pairs and simultaneously minimize the similarity between negative pairs. More recently, it has also been applied to the domain of text classification. [11] introduce a model named SiameseXML that combines siamese networks with high-capacity extreme classifiers to effectively utilize label metadata and make high-quality predictions for rare labels at the scale of millions of labels. [35] proposes a contrastive learning method for text classification that incorporates linguistic knowledge and an adaptive augmentation policy. The approach

constructs word-level positive and negative sample pairs using WordNet, injects linguistic knowledge through a novel contrastive learning function, and dynamically selects data augmentation policies to obtain better sentence representations. [36] develop a novel approach for few-shot text classification that uses a triplet contrast network and dynamic rate of change sampling to improve learning efficiency and focus on difficult-to-classify samples. [37] use a pre-trained transformer and fine-tune their model with a triplet loss to learn similarities between document and label embeddings.

3. Method

3.1. Problem description

We begin by formally describing the multi-label classification setting we address in this study. In multi-label document classification, datasets consist of document-label pairs $\{x_i, y_i\}$, where x_i is the text of a document and y_i is encoded as a binary target vector with $y_i \in \{0,1\}^N$, where N is the number of unique labels in the dataset. In this binary representation, a value of 1 indicates that the corresponding label is present or applicable to the document, while a value of 0 indicates that the label is absent or not applicable. The goal is to learn a function $f(x)$ that maps the input x_i to the binary output vector y_i. However, in many real-world datasets, labels are words or phrases that contain valuable information. This label information is discarded when the labels are encoded as a binary vector. Especially in long-tail scenarios, where many labels have very few training examples, ignoring label semantics can limit a model's generalization capabilities.

3.2. Solution approach

To address this limitation, we reformulate the multi-label classification problem as a semantic similarity task. Instead of treating labels as abstract indices, we leverage their textual descriptions and embed them in a shared semantic space alongside the document representations. In this formulation, the task becomes learning to measure similarity between document and label pairs. Therefore, we approach multi-label document classification by leveraging the label information t of the textual label descriptions. In this setting, each training pair consists of a document x_i and a set of textual descriptions of labels T_i, which is a subset of all existing labels $\mathcal{T} = \{t_1, t_2, \ldots, t_N\}$.

Our goal is to learn two functions, $f_\theta(x_i)$ and $g_\phi(t_j)$, parameterized by θ and ϕ, which map documents and labels in a joint vector space. In the context of NLP, these functions are often referred to as encoders, which we denote as $\mathcal{E}_\theta$ and $\mathcal{E}_\phi$ for the remainder of the paper. Let $\mathcal{E}_\theta(x_i) = d_i \in \mathbb{R}^n$ represent the embedding of a document and $\mathcal{E}_\phi(t_j) = l_j \in \mathbb{R}^n$ the embedding of a label, where n is the dimensionality of the embedding space and i and j are the indices of the document and label, respectively. In this space, the embedding of a label l_j should be similar to the embedding of a document d_i if $t_j \in T_i$, and dissimilar otherwise.

We measure similarity as the inner product, which takes into account the direction and magnitude of a vector. During inference, we classify a document by performing an inner product search with all label vectors.

3.3. Model architecture

Next, we present the architectural components of our approach. Our model is designed with dual text encoders: the document encoder $\mathcal{E}_\theta$ and the label encoder $\mathcal{E}_\phi$. We take advantage of pre-trained language models based on the transformer architecture [38] as the foundation for both encoders. The selection of this model, along with the hyperparameter settings for jointly training both encoders, is described in detail in Section 4.3. After each encoder, we deploy an embedding head that is illustrated in Fig. 2. It consists of a linear transformation followed by a gated mechanism. The output of the gated mechanism is then added to the original input embedding through a skip connection [39]. In addition, a dropout layer [40] is applied to the trainable branch to prevent overfitting. The gated embedding head functions similarly to adapter layers [41], enabling adaptation of pre-trained transformer representations while preserving the original embedding space. Although we fine-tune the full model, the gating mechanism allows the network to skip updates to the pre-trained model, which is especially valuable for avoiding overfitting and generalizing to few-shot or zero-shot labels.

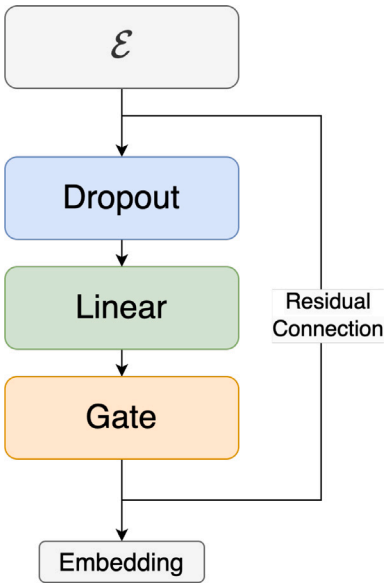


Fig. 2. Architecture of the gated embedding head applied after both document ($\mathcal{E}_\theta$) and label ($\mathcal{E}_\phi$) encoder outputs.

Our training employs contrastive learning with in-batch negative examples. Fig. 3 illustrates the encoding process for a batch of documents which is inspired by [42] and adapted to the multi-label scenario. Each document and label are passed through their respective encoders and heads to obtain d_i and l_j. Subsequently, we calculate the dot product for each pair of document and label embeddings $d_i \cdot l_j$ within the batch which results in a similarity Matrix $B \in \mathbb{R}^{b \times l_{batch}}$ where b is the batch size (*i.e.*, the number of documents in the batch) and l_{batch} is the number of unique labels in the batch.

While the number of documents remains constant across batches, the number of labels may fluctuate depending on set of correct labels in the batch. Fig. 3 shows one batch. As the label "sports" is present twice within the batch (for yellow document and purple document), the resulting count of label embeddings is nine, as duplicate labels are removed in advance. In practice, for larger batch sizes the number of unique labels in a batch can be in the thousands.

For each batch, we dynamically construct a target matrix $Y \in \{0,1\}^{b \times l_{batch}}$ that reflects the correct associations between documents and labels. For the given example, the target vector is represented as:

$$Y = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

3.4. Loss functions

Masked binary cross entropy loss

We apply the sigmoid function to the inner products of the embeddings to obtain the predictions as probabilities $\hat{y} = \sigma(d_i \cdot l_j) \in]0,1[$. These probabilities are then used in conjunction with the target vector to compute the loss.

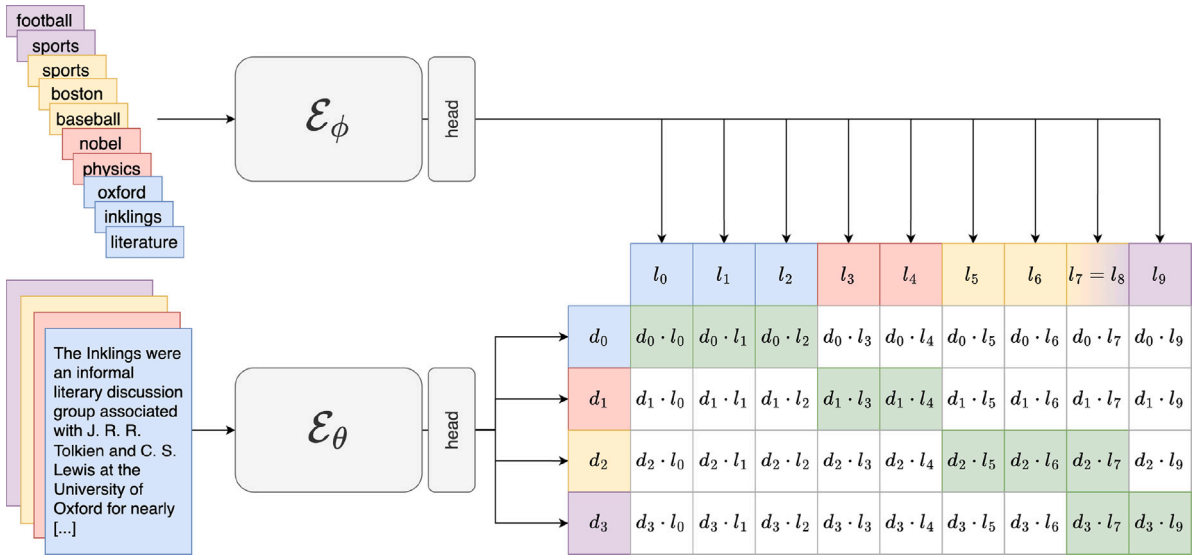


Fig. 3. CTN-LT Model Architecture: The texts of documents and labels are embedded by individual encoders $\mathcal{E}_\theta$ and $\mathcal{E}_\phi$ and the inner product is calculated between all labels and documents. During training the goal is to maximize the inner product of correct labels (marked green) while minimizing all negative labels in the batch. The colors denote single documents and their respective labels.

The Binary Cross Entropy loss (BCE) is frequently used in multi-label classification. Eq. (1) shows the loss ℓ_n for a single class n, where $y_n \in 0, 1$ is the binary label indicator, and $\hat{y}_n \in [0, 1]$ are the predicted probabilities. We can calculate the BCE loss for one training example as shown in (2).

$$\ell_n = -y_n \log(\hat{y}_n) + (1 - y_n) \log(1 - \hat{y}_n) \tag{1}$$

$$J_{\text{BCE}} = -\frac{1}{N} \sum_{n=1}^{N} \ell_n \tag{2}$$

The loss function calculates a loss value per class which is averaged for each training example. This property poses a challenge for classification problems where the number of possible labels is high: As only a handful labels are correct in those settings, the vast majority of labels y_i is zero. This circumstance might incentives the network to assign a very low probability to every class when trying to minimize the loss converging at a local minimum, ultimately leading to many false negative predictions. To address this problem, we propose a modified version of the Binary Cross Entropy Loss we denote as *masked Binary Cross Entropy loss* (mBCE). We add an additional mask parameter $m \in \mathbb{N}$ that masks all loss values except the highest m losses per example. We achieve this by calculating all per class loss values ℓ_n and sort them in descending order to identify the top m largest losses. Let A denote the set of indices corresponding to the top m losses. We then modify the BCE by only considering the highest m losses as seen in (3). This means only the most significant false negatives and false positives are penalized. We argue that this improves the training process in two ways. First, the process of masking a large proportion of classes for each training example directly tackles the class imbalance problem where networks converge to assigning very low probabilities across all classes. Second, at the same time only the hardest labels contribute to the training which leads to a desired effect of sampling hard negatives that is well known to improve contrastive learning approaches [30].

$$J_{\text{mBCE}} = \frac{1}{m} \sum_{n \in A} \ell_n \tag{3}$$

Adapted cross entropy loss

In addition to the *masked Cross Entropy loss* described in the previous chapter, we also apply a modified version of *Cross Entropy loss*, which is used in multi-class classification problems where the softmax function is applied to the output logits. The softmax function as seen in Eq. (4) is usually applied to the logits z in classification tasks.

$$softmax(\hat{y}_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{4}$$

After applying the softmax to the logits the Cross Entropy loss can be calculated as follows:

$$J_{CE}(y, \hat{y}) = -\sum_i y_i \log(\hat{y}_i) \tag{5}$$

Applying softmax and CE is ill-suited for multi-label classification due to its inherent design for exclusive class selection, as all probabilities $\hat{y}_i$ add up to 1. In addressing the challenge of applying Cross Entropy loss to multi-label text classification, we propose a simple adaptation. The adaptation involves a procedural iteration over each positive class label within a given batch. For each positive label, the dot product corresponding to that label and the all dot products of all negative labels are subjected to a softmax operation. This process essentially transforms the multi-label scenario into a series of binary-like classifications, allowing for the application of Cross Entropy in a context that traditionally precludes its use. The final loss is computed by averaging the Cross Entropy losses derived from each of these iterations.

CTN-LT loss

To effectively balance the benefits of the two previously introduced loss functions – the masked Binary Cross Entropy loss (mBCE) and the adapted Cross Entropy loss – we define the final training objective of our model as a weighted combination of both. We introduce a weighting parameter $\alpha \in [0, 1]$ to control the relative contribution of each loss. The final CTN-LT loss $J_{\text{CTN-LT}}$ is defined as:

$$J_{\text{CTN-LT}} = \alpha \cdot J_{\text{CE}} + (1 - \alpha) \cdot J_{\text{mBCE}} \tag{6}$$

3.5. Inference

Once training is complete, we turn to the inference procedure used to predict labels for unseen documents. We use the label encoder $\mathcal{E}_\phi$ once to create the label embeddings for each label in the dataset. The label encoder can then be discarded as it is not needed anymore, which reduces the model size by half and makes inference very fast. To predict the labels for an unseen document, we use the document encoder $\mathcal{E}_\theta$ to embed the document and calculate the inner product to all label embeddings created in the previous step and take the top k most similar labels as the prediction.

Table 1
Overview of datasets used in this study, including number of labels (L), training/test instances, label density, and average document length.

Dataset	L	N_{train}	N_{test}	$\bar{N}_{\text{label}}$	$\bar{L}_{\text{point}}$	N_{words}
EURLex-4K	3,993	15,539	3,809	25.73	5.31	1240.64
Wiki10-31K	30,938	14,146	6,616	8.52	18.64	2104.42
AmazonCat-13K	13,330	1,186,239	306,782	448.57	5.04	208.6

Table 2
Breakdown of label frequency categories across datasets. Labels are grouped into 'head' (frequent), 'few-shot', and 'zero-shot' categories following the thresholds in [45].

Dataset	L	Head labels	Tail labels	
		frequent (> 50)	few-shot (1–50)	zero-shot (0)
EURLex-4K	3,993	391	3,410	155
Wiki10-31K	30,938	628	29,319	991
AmazonCat-13K	13,330	5,106	8,224	0

4. Experiments

4.1. Data

To evaluate the effectiveness of our approach, we conduct a comprehensive set of experiments across three benchmark datasets, namely *Wiki10-31K* [43], *EURLex-4K* [44] and *AmazonCat-13K* [4]. Table 1 provides additional details about the datasets. It shows the total number of unique labels L, the number of training points in the training N_{train} and test set N_{test}, the average number of training points per label $\bar{N}_{\text{label}}$, the average number of labels per training point $\bar{L}_{\text{point}}$, and the average number of words per training point N_{words}. All dataset have several thousand unique labels whose frequencies resemble a long tail distribution similar to Fig. 1.

In this study, we adopt the framework introduced by [45] to categorize labels based on their frequency in the dataset. This framework defines three categories: frequent labels that occur more than 50 times, few-shot labels that occur less than 50 times, and zero-shot labels that do not occur in the training dataset but do occur in the test set. For the purposes of our analysis, we consider frequent labels as head labels, while few-shot and zero-shot labels are collectively referred to as tail labels. It is crucial to distinguish zero-shot labels as a separate category due to the unique challenges they present in prediction tasks. These labels, having no representation in the training data, are particularly difficult to predict accurately. Table 2 presents the distribution of label categories across the three datasets used in this study. The datasets exhibit varying proportions of head and tail labels. Notably, the EURLex-4K dataset is characterized by a predominance of few-shot labels, with a relatively small number of frequent and zero-shot labels. The Wiki10-31K dataset shows a similar pattern, with a large majority of few-shot labels, a moderate number of zero-shot labels, and a smaller proportion of frequent labels. In contrast, the AmazonCat-13K dataset presents a more balanced distribution between frequent and few-shot labels, with no zero-shot labels present.

4.2. Metrics

To evaluate the performance in comparison to other approaches, we use several commonly used metrics and their propensity scored counterparts. Propensity scoring addresses challenges in multi-label classification, where labels follow a power law distribution. The propensity model by [6] accounts for the likelihood of label annotation, especially for rare but informative tail labels, allowing for balanced performance evaluation across frequent and infrequent labels.

For our evaluation, we use precision at top k (P@k), Normalized Discounted Cumulative Gains at k (nDCG@K) and their propensity scored counterparts PSP@K and PSnDCG@k. Here, the parameter k defines the cutoff threshold at which the ranked list of predicted labels is evaluated (*e.g.*, top 1, 3, or 5 predictions). P@k is defined in Eq. (7).

$$\text{P@}k := \frac{1}{k} \sum_{l \in \text{rank}_k(\hat{\mathbf{y}})} \mathbf{y}_l \tag{7}$$

$\sum_{l \in \text{rank}_k(\hat{\mathbf{y}})}$ denotes a sum over the indices l that belong to the set of top k rank in the predicted scores $\hat{y}$. It iterates through the top k ranked labels, summing the binary relevance indicators $y_l \in \{0, 1\}$ for each of the labels. Eq. (8) shows the propensity weighted version of P@k, where each binary indicator is scaled by the propensity score for the corresponding label. This gives a rare label a higher weight and therefore a lower propensity score.

$$\text{PSP@}k := \frac{1}{k} \sum_{l \in \text{rank}_k(\hat{\mathbf{y}})} \frac{\mathbf{y}_l}{\boldsymbol{p}_l} \tag{8}$$

The DCG at a particular rank k accounts for the relevance of each item up to the kth position in the ranked list. Relevance indicators are discounted logarithmically, giving higher importance to items ranked higher in the list. The formula for DCG@k is given by:

$$\text{DCG@}k := \sum_{l \in \text{rank}_k(\hat{\mathbf{y}})} \frac{\mathbf{y}_l}{\log(l+1)} \tag{9}$$

Eq. (10) is the normalized version of the DCG where $\|y\|_0$ is the count of relevant items in the ground truth. This way, the formula captures the quality of the ranking up to position k, normalized by the best possible ranking of the same size, ensuring the score ranges between 0 (worst) and 1 (best), assuming all relevance scores are binary.

$$\text{nDCG@}k := \frac{\text{DCG@}k}{\sum_{l=1}^{\min(k, \|\mathbf{y}\|_0)} \frac{1}{\log(l+1)}} \tag{10}$$

Eqs. (11) and (12) represent the variant scored for propensity of the DCG and nDCG metrics, respectively. These versions incorporate the propensity scores of the labels to adjust for the label distribution's imbalance. By doing so, they aim to provide a fairer evaluation of the model's performance across different labels, especially in datasets where some labels are much more frequent than others.

The PSDCG@k metric, as shown in Eq. (11), modifies the standard DCG@k formula by dividing the relevance indicator $\mathbf{y}_l$ by the propensity score p_l for each label l, in addition to logarithmic discounting. This adjustment means that the relevance of rarer labels (with lower propensity scores) is amplified, reflecting their increased difficulty in being correctly predicted.

The PSnDCG@k metric, described in Eq. (12), further normalizes PSDCG@k by dividing it by the ideal PSDCG score, which is calculated assuming a perfect ranking of the items to position k. This normalization ensures that the PSnDCG@k score is bounded between 0 and 1, where 1 represents a perfect ranking with respect to both relevance and label distribution, and 0 represents the worst possible ranking.

$$\text{PSDCG@}k := \sum_{l \in \text{rank}_k(\hat{\mathbf{y}})} \frac{\mathbf{y}_l}{p_l \log(l+1)} \tag{11}$$

Table 3
Hyperparameter configurations for training CTN-LT across datasets.

Hyperparameter	EURLex-4K	Wiki10-31K	AmazonCat-13K
Tokens	256	256	128
Label Input Length	16	16	16
Batch Size	128	128	768
Training Epochs	30	30	20
mBCE (m)	30	50	50
α	0.8	0.8	0.8

$$\text{PSnDCG@}k := \frac{\text{PSDCG@}k}{\sum_{l=1}^{k} \frac{1}{\log(l+1)}} \tag{12}$$

In the remainder of the paper, we will refer to nDCG@k as N@k and to PSnDCG@k as PSN@k.

4.3. Training

We train our model on a single RTX 3090Ti GPU. We use the `distilbert-base-uncased`[1] [46] from Huggingface as the base encoder for both documents and labels. This model offers a favorable balance between computational efficiency and performance, making it well-suited for large-scale experiments on long-tail datasets. For each dataset, we use the predefined training and test splits provided with the datasets to ensure fair comparison with results reported in prior work. In addition, we set aside 10% of the training data for validation and hyperparameter tuning. We chose AdamW [47] as our optimizer and schedule the learning rate according to the 1cycle policy [48] with a maximum learning rate of 0.00005. Table 3 shows the remaining hyperparameters used for training. For EURLex-4K and Wiki10-31K datasets, we set the number of tokens that represent the documents to 256. For AmazonCat-13K, we use 128 as input length, because using more tokens did not make a difference regarding the performance but allows a much faster training due to the quadratic relation between input length and required computational cost of the attention mechanism used in transformer models. As the textual representations of labels in all datasets consist of words and short phrases, 16 tokens are enough to encode them. We choose the largest batch size possible for our hardware. During our experiments, we evaluated the model's performance on validation datasets while adjusting the hyperparameters m and α to different values. Through this analysis, we observed that setting m too low ($m < 20$) or too high ($m > 80$) degraded the performance on the datasets. We determined that the optimal value for m varied across datasets, while $\alpha = 0.8$ worked consistently well in most scenarios. Specifically, we found that $m = 30$ for the EURLex-4K dataset and $m = 50$ for both the Wiki10-31K and AmazonCat-13K datasets provided a practical balance between head and tail label performance.

4.4. Baselines

We compare the performance of our model against several state-of-the-art methods from the literature using the metrics described in 4.2. All reported metrics are sourced directly from the respective articles or from the Extreme Classification Repository website [18]. We did not tune or reimplement these models to avoid introducing subjectivity. For comparison, we selected approaches from the XML literature [10,20,22–24,49,50] as well as approaches specifically designed for long-tail classification problems [7,28].

To ensure a fair comparison, we consistently report results for single-model variants only, since our proposed CTN-LT is also a single model. While ensemble-based methods can yield stronger performance, they come with significant computational overhead and are less suited for low-latency applications. All baseline results are based on official model implementations as provided by the respective authors, and standard test splits as defined by the respective datasets.

[1] https://huggingface.co/distilbert-base-uncased

4.5. Results

4.5.1. Model performance

The results of the benchmark are summarized in Table 4. CTN-LT achieves competitive results on several metrics for the EURLex-4K dataset. Particularly noteworthy is its performance on propensity-scored metrics, where it outperforms all other methods.

While DBGB slightly edges out in precision metrics like P@3 and P@5, CTN-LT maintains competitive performance, ranking second or third in most precision-based measures. This suggests that our method strikes a good balance between overall precision and tail label prediction.

The results for the Wiki10-31K dataset further validate the effectiveness of CTN-LT. Although APLC-XLNet, A-XML, and DBGB show better performance in precision metrics, our method achieves the highest scores in all propensity-scored metrics, with a notable margin to the second best approach.

On the AmazonCat-13k dataset, CTN-LT demonstrates excellent overall performance. It achieves the highest scores in all propensity-scored precision except for PSP@1 where it comes second after AnnexML. These results surpass those of other state-of-the-art methods, including A-XML + TailMix and DBGB. While DBGB shows better results in some precision metrics, CTN-LT's performance is still highly competitive, ranking second or third in most of these measures. The consistent strong performance across different metrics and datasets underscores the effectiveness and versatility of our proposed method.

The results across all three datasets highlight the strong performance of our proposed CTN-LT model in document classification tasks. CTN-LT consistently achieved competitive results in terms of precision@k and nDCG@k metrics, often surpassing the state-of-the-art baselines. Moreover, its superior performance in the propensity-scored metrics demonstrates the ability of CTN-LT to predict labels that appear less frequently in the datasets.

4.5.2. Zero-shot and few-shot performance

To further investigate the performance on tail labels, we evaluate our model, CTN-LT, and compare it with APLC-XLNet, on frequent, few-shot, and zero-shot labels. We chose APLC-XLNet as a comparison as the authors provide model checkpoints allowing the exact replication of the results reported in the paper. For the evaluation, we create subsets of the training data that include only instances with frequent labels, few-shot labels, and zero-shot labels. We test both models under two different conditions: masked, where we only allow the model to predict labels within the corresponding category by masking all other labels, and unmasked, where we allow the model to predict any label without restrictions. We use Recall-Precision@k (RP@5) and Normalized Discounted Cumulative Gain@k (NDCG@5) as our evaluation metrics. The results are presented in Table 5. An analysis of the results reveals several key findings. In terms of overall performance, both models perform similarly on the "All Labels" category across all datasets, with APLC-XLNet slightly outperforming CTN-LT in most cases. For frequent labels, APLC-XLNet generally performs better in the masked condition, while CTN-LT shows competitive performance in some cases, particularly on the AmazonCat dataset.

CTN-LT demonstrates superior performance on few-shot labels, particularly in masked conditions across all datasets. This indicates that our model more effectively generalizes to labels with limited training examples. Furthermore, CTN-LT significantly outperforms APLC-XLNet on zero-shot labels, especially in masked scenarios, highlighting its enhanced ability to predict labels not encountered during training. In contrast, APLC-XLNet fails to predict any zero-shot labels in the unmasked case. CTN-LT successfully identifies previously unseen labels through its similarity-based architecture leveraging pre-trained language models. This advantage is most evident in masked zero-shot scenarios, where CTN-LT notably surpasses APLC-XLNet.

Table 4
Comparison of CTN-LT performance with other XML baselines on EURLex-4K, Wiki10-31K, and AmazonCat-13K Datasets. Metrics include precision@k (P@k), normalized DCG@k (N@k), and their propensity-scored counterparts (PSP@k and PSN@k). Best results in bold, second-best underlined.

EURLex-4K										
Model	P/N@1	P@3	P@5	N@3	N@5	PS{P/N}@1	PSP@3	PSP@5	PSN@3	PSN@5
A-XML + TailMix [28]	84.11	71.93	60.48	–	–	43.64	51.11	53.69	49.11	50.94
APLC-XLNet [10]	**87.72**	74.56	62.28	77.90	71.75	42.93	49.84	53.07	48.00	50.40
AnnexML [20]	79.26	64.30	52.33	68.13	61.60	34.25	39.83	42.76	38.35	40.30
DiSMEC [49]	82.40	68.50	57.70	72.50	66.70	41.20	45.40	49.30	44.30	46.90
DBGB [7]	87.61	**75.21**	**62.54**	**78.53**	**72.30**	41.19	51.75	55.75	–	–
FastXML [22]	76.37	63.36	52.03	66.63	60.61	33.17	39.68	41.99	37.92	39.55
Parabel [23]	82.25	68.71	57.53	72.17	66.54	36.44	44.08	48.46	41.99	44.91
Bonsai [24]	82.96	69.76	58.31	73.15	67.41	37.08	45.13	49.57	42.94	46.10
XT [50]	78.97	65.64	54.44	69.05	63.23	33.52	40.35	44.02	38.50	41.09
CTN-LT (ours)	87.68	73.37	60.49	77.01	70.33	**44.22**	**52.80**	**56.03**	**50.57**	**53.13**
Wiki10-31K										
Model	P/N@1	P@3	P@5	N@3	N@5	PS{P/N}@1	PSP@3	PSP@5	PSN@3	PSN@5
A-XML + TailMix [28]	84.05	76.87	67.91	–	–	13.47	17.09	18.44	16.22	17.26
APLC-XLNet [10]	**89.44**	**78.93**	**69.73**	**81.38**	**74.41**	14.84	15.85	17.04	15.58	16.40
AnnexML [20]	86.49	74.27	64.20	77.13	69.44	11.90	12.76	13.58	12.53	13.10
DiSMEC [49]	85.20	74.60	65.90	77.10	70.40	13.60	13.10	13.80	13.20	13.60
DBGB [7]	89.23	78.67	69.59	81.08	74.19	14.31	15.67	17.01	–	–
FastXML [22]	83.03	67.47	57.76	75.35	63.36	9.80	10.17	10.54	10.08	10.33
Parabel [23]	84.17	72.46	63.37	75.22	68.22	11.68	12.73	13.69	12.47	13.14
Bonsai [24]	84.69	73.69	64.39	76.25	69.17	11.78	13.27	14.28	12.89	13.61
XT [50]	86.15	75.18	65.41	77.76	70.35	11.87	13.08	13.89	12.78	13.36
CTN-LT (ours)	89.24	76.99	66.89	79.79	72.10	**18.28**	**20.64**	**21.88**	**20.64**	**20.93**
AmazonCat-13K										
Model	P/N@1	P@3	P@5	N@3	N@5	PS{P/N}@1	PSP@3	PSP@5	PSN@3	PSN@5
A-XML + TailMix [28]	95.23	80.95	65.70	–	–	51.48	65.44	72.62	61.62	66.42
APLC-XLNet [10]	94.56	79.82	64.61	88.74	86.66	52.22	65.08	71.40	62.57	67.92
AnnexML [20]	93.54	78.37	63.30	87.29	85.10	49.04	61.13	69.64	58.83	65.47
DiSMEC [49]	93.40	79.10	64.10	87.70	85.80	**59.10**	67.10	71.20	65.20	68.80
DBGB [7]	**96.59**	**83.61**	**68.25**	**92.48**	**90.57**	53.95	69.15	77.52	–	–
FastXML [22]	93.11	78.20	63.41	87.07	85.16	48.31	60.26	69.30	56.90	62.75
Parabel [23]	93.03	79.16	64.52	87.72	86.00	50.93	64.00	72.08	60.37	65.68
Bonsai [24]	92.98	79.13	64.46	87.68	85.92	51.30	64.60	72.48	–	–
XT [50]	92.59	78.24	63.58	86.90	85.03	49.61	62.22	70.24	59.71	66.04
CTN-LT (ours)	95.01	80.64	65.52	89.73	88.03	57.39	**71.87**	**78.28**	**68.79**	**74.06**

In summary, while APLC-XLNet shows slightly better performance on frequent labels, our CTN-LT model demonstrates superior capabilities in handling few-shot and zero-shot scenarios. These results highlight CTN-LT's potential for more challenging and diverse label prediction tasks, particularly in situations where training data for certain labels is limited or non-existent.

4.6. Ablation study

To prove the effectiveness of our design choices for our architecture and the different loss functions, we conduct an ablation study that aims to isolate the effects of the introduced loss functions of the model. Table 6 shows the results of this study.

The first architecture, denoted as BCE, utilizes the unmodified version of the Binary Cross Entropy loss. The second variant, masked BCE (mBCE), implements our modified version of the Binary Cross Entropy loss, as presented in Section 3.4. The results indicate a significant improvement of the mBCE over the standard BCE loss function across almost all metrics, with the exceptions of PSP@1, PSN@1, and PSN@3. The model variant labeled CE relies solely on the modified version of the Cross Entropy Loss. This approach demonstrates superior performance on P@k and N@k values but performs notably worse on the propensity-weighted metrics. Our final model, CTN-LT, combines the mBCE and CE losses by averaging them. This combination yields excellent results on propensity-weighted metrics while only marginally compromising performance on other metrics. This approach allows us to leverage the strengths of both loss functions, resulting in a more robust and balanced model performance across various evaluation criteria.

4.7. Discussion

The results of our experiments demonstrate the effectiveness of CTN-LT in addressing the long-tail problem in multi-label text classification. Our model shows significant improvements, particularly in propensity-scored metrics, which are designed to give more weight to tail labels. This improvement indicates that CTN-LT is particularly adept at handling infrequent labels, a crucial aspect in many real-world applications where the distribution of labels is often highly skewed.

The performance of CTN-LT on head labels is competitive with state-of-the-art models, while it significantly outperforms existing approaches on tail labels. This balance is crucial, as it addresses a common trade-off in multi-label classification where improvements in tail label prediction often come at the cost of reduced performance on head labels. Our model manages to achieve superior tail label performance while maintaining strong results on head labels, indicating a more balanced and robust approach to multi-label classification.

The ablation study provides insights into the effectiveness of our design choices. The combination of masked Binary Cross Entropy loss and adapted Cross Entropy loss proves to be crucial in balancing the model's performance across different label frequencies. This hybrid approach allows the model to learn from frequent and infrequent labels effectively, contributing to its robust performance across various metrics.

Of course, our work is not without limitations. Obviously, our approach relies on textual representations of labels and is therefore not usable if those are not available. The computational cost during inference might be a limitation for datasets with several hundred thousand labels, since the inner search is performed on all labels. As always, our findings might be dataset-specific, which we tried to ameliorate by evaluating three different datasets.

Table 5
Performance comparison of APLC-XLNet and CTN-LT models across head (frequent), few-shot, and zero-shot label categories. Both masked and unmasked settings are shown.

	APLC-XLNet				CTN-LT			
	masked		unmasked		masked		unmasked	
	RP@5	N@5	RP@5	N@5	RP@5	N@5	RP@5	N@5
				EURLex-4K				
All Labels	66.13	71.75	66.13	71.75	59.34	69.41	59.34	69.41
Frequent	57.42	63.67	48.75	52.75	46.38	70.79	42.22	60.74
Few Shot	23.93	30.27	17.38	18.82	25.87	58.22	22.47	39.68
Zero Shot	0.93	2.08	0	0	5.45	21.59	4.39	10.64
				Wiki10-31K				
All Labels	69.74	74.42	69.74	74.42	66.96	72.16	66.96	72.16
Frequent	66.93	73.69	60.82	67.38	66.05	72.79	50.69	55.75
Few Shot	27.66	32.92	9.00	8.65	28.87	35.89	16.52	17.79
Zero Shot	0.17	0.38	0	0	5.44	19.35	0.84	1.89
				AmazonCat-13K				
All Labels	64.61	86.66	64.61	86.66	63.93	88.03	63.93	88.03
Frequent	64.19	87.16	63.96	86.86	64.45	87.64	62.98	86.21
Few Shot	17.00	63.11	6.05	13.53	17.31	64.80	8.91	19.45
Zero Shot	–	–	–	–	–	–	–	–

Table 6
Ablation study using different loss configurations: standard binary cross entropy (BCE), masked BCE (mBCE), modified Cross Entropy (CE), and the combined CTN-LT loss.

	AmazonCat-13k											
Model Variant	P@1	P@3	P@5	N@1	N@3	N@5	PSP@1	PSP@3	PSP@5	PSN@1	PSN@3	PSN@5
BCE (m = 0)	92.38	76.41	61.74	92.38	85.46	83.45	59.45	70.77	74.42	59.45	68.43	71.89
mBCE (m = 50)	94.56	79.32	64.37	94.56	88.33	86.44	57.08	71.47	76.78	57.08	68.34	72.94
CE	95.48	81.01	65.43	95.48	89.99	87.74	51.81	67.98	75.59	51.81	64.47	70.60
CTN-LT (m = 50)	95.01	80.64	65.52	95.01	89.73	88.03	57.39	71.87	78.28	57.39	68.79	74.06

5. Conclusion

This study presents CTN-LT, a novel deep learning model designed to mitigate the long-tail problem in multi-label text classification. By prioritizing accurate prediction of tail labels while maintaining robust performance on head labels, CTN-LT achieves state-of-the-art results across multiple benchmark datasets. The incorporation of few-shot and zero-shot learning capabilities further equips the model to handle dynamic and evolving label spaces, making it particularly well-suited for real-world applications such as search engines and recommender systems. Our empirical evaluations demonstrate that CTN-LT effectively balances between head and tail label performance, addressing a critical gap in existing multi-label classification approaches. The propensity-scored metrics underscore the model's strength in handling infrequent labels, thereby enhancing the overall utility and reliability of text classification systems. In conclusion, CTN-LT not only advances the methodological landscape of multi-label text classification, but also offers tangible benefits for industries reliant on comprehensive and nuanced text categorization. Future research will aim to refine the model's scalability and generalizability, ensuring its applicability across an even wider array of domains and datasets.

CRediT authorship contribution statement

Johannes Melsbach: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Project administration, Methodology, Data curation, Conceptualization. **Frederic Haase:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Data curation, Conceptualization. **Sven Stahlmann:** Writing – review & editing, Software, Methodology, Conceptualization. **Stefan Hirschmeier:** Writing – review & editing, Writing – original draft, Supervision. **Detlef Schoder:** Writing – review & editing, Supervision.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used *Grammarly* in order to improve the readability and language of the manuscript. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

All used datasets are publicly available.

References

[1] A. Sun, E.-P. Lim, Y. Liu, On strategies for imbalanced text classification using SVM: A comparative study, Decis. Support Syst. 48 (1) (2009) 191–201, http://dx.doi.org/10.1016/j.dss.2009.07.011.

[2] I.-D. Borlea, R.-E. Precup, F. Dragan, A.-B. Borlea, Centroid update approach to K-means clustering, Adv. Electr. Comput. Eng. 17 (4) (2017) 3–10, http://dx.doi.org/10.4316/AECE.2017.04001.

[3] S. Chen, S. Ke, S. Han, S. Gupta, U. Sivarajah, Which product description phrases affect sales forecasting? An explainable AI framework by integrating WaveNet neural network models with multiple regression, Decis. Support Syst. 176 (2024) 114065, http://dx.doi.org/10.1016/j.dss.2023.114065.

[4] J. McAuley, J. Leskovec, Hidden factors and hidden topics: understanding rating dimensions with review text, in: Proceedings of the 7th ACM Conference on Recommender Systems, ACM, Hong Kong China, 2013, pp. 165–172, http://dx.doi.org/10.1145/2507157.2507163.

[5] C. Sansone, G. Sperlí, Legal information retrieval systems: State-of-the-art and open issues, Inf. Syst. 106 (2022) 101967, http://dx.doi.org/10.1016/j.is.2021.101967.

[6] H. Jain, Y. Prabhu, 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, ACM, San Francisco California USA, 2016, pp. 935–944, http://dx.doi.org/10.1145/2939672.2939756.

[7] Y. Yao, J. Zhang, P. Zhang, Y. Sun, A dual-branch learning model with gradient-balanced loss for long-tailed multi-label text classification, ACM Trans. Inf. Syst. 42 (2) (2024) 1–24, http://dx.doi.org/10.1145/3597416.

[8] L. Xiao, X. Zhang, L. Jing, C. Huang, M. Song, Does head label help for long-tailed multi-label text classification, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, No. 16, 2021, pp. 14103–14111, http://dx.doi.org/10.1609/aaai.v35i16.17660.

[9] Z. Ge, X. Li, To be or not to be, tail labels in extreme multi-label learning, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, CIKM '21, Association for Computing Machinery, New York, NY, USA, 2021, pp. 555–564, http://dx.doi.org/10.1145/3459637.3482303.

[10] H. Ye, Z. Chen, D.-H. Wang, B.D. Davison, Pretrained generalized autoregressive model with adaptive probabilistic label clusters for extreme multi-label text classification, in: Proceedings of the 37th International Conference on Machine Learning, PMLR, 2020, pp. 10809–10819.

[11] K. Dahiya, A. Agarwal, D. Saini, G. K, J. Jiao, A. Singh, S. Agarwal, P. Kar, M. Varma, SiameseXML: Siamese networks meet extreme classifiers with 100M labels, in: Proceedings of the 38th International Conference on Machine Learning, PMLR, 2021, pp. 2330–2340.

[12] W.-C. Chang, H.-F. Yu, K. Zhong, Y. Yang, 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, KDD '20, Association for Computing Machinery, New York, NY, USA, 2020, pp. 3163–3171, http://dx.doi.org/10.1145/3394486.3403368.

[13] K. Dahiya, N. Gupta, D. Saini, A. Soni, Y. Wang, K. Dave, J. Jiao, G. K, P. Dey, A. Singh, D. Hada, V. Jain, B. Paliwal, A. Mittal, S. Mehta, R. Ramjee, S. Agarwal, P. Kar, M. Varma, NGAME: Negative mining-aware mini-batching for extreme classification, in: Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, WSDM '23, Association for Computing Machinery, New York, NY, USA, 2023, pp. 258–266, http://dx.doi.org/10.1145/3539597.3570392.

[14] Y. Huang, B. Giledereli, A. Köksal, A. Özgür, E. Ozkirimli, Balancing methods for multi-label text classification with long-tailed class distribution, 2021, http://dx.doi.org/10.48550/arXiv.2109.04712, arXiv:2109.04712 [cs].

[15] T. Wei, W.-W. Tu, Y.-F. Li, G.-P. Yang, Towards robust prediction on tail labels, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, KDD '21, Association for Computing Machinery, New York, NY, USA, 2021, pp. 1812–1820, http://dx.doi.org/10.1145/3447548.3467223.

[16] P. Wang, K. Han, X.-S. Wei, L. Zhang, L. Wang, Contrastive learning based hybrid networks for long-tailed image classification, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 943–952.

[17] R. Zhang, Y.-S. Wang, Y. Yang, D. Yu, T. Vu, L. Lei, Long-tailed extreme multi-label text classification by the retrieval of generated pseudo label descriptions, in: A. Vlachos, I. Augenstein (Eds.), Findings of the Association for Computational Linguistics: EACL 2023, Association for Computational Linguistics, Dubrovnik, Croatia, 2023, pp. 1092–1106, http://dx.doi.org/10.18653/v1/2023.findings-eacl.81.

[18] K. Bhatia, K. Dahiya, H. Jain, P. Kar, A. Mittal, Y. Prabhu, M. Varma, The extreme classification repository: Multi-label datasets and code, 2016.

[19] L. Yan, T. Zhao, X. Xie, R.-E. Precup, OSSEFS: An online semi-supervised ensemble fuzzy system for data streams learning with missing values, Expert Syst. Appl. 255 (2024) 124695, http://dx.doi.org/10.1016/j.eswa.2024.124695.

[20] Y. Tagami, AnnexML: Approximate nearest neighbor search for extreme multi-label classification, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, Halifax NS Canada, 2017, pp. 455–464, http://dx.doi.org/10.1145/3097983.3097987.

[21] K. Bhatia, H. Jain, P. Kar, M. Varma, P. Jain, Sparse local embeddings for extreme multi-label classification, in: Advances in Neural Information Processing Systems, 28, Curran Associates, Inc., 2015.

[22] Y. Prabhu, 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, ACM, New York New York USA, 2014, pp. 263–272, http://dx.doi.org/10.1145/2623330.2623651.

[23] Y. Prabhu, A. Kag, S. Harsola, R. Agrawal, M. Varma, Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising, in: Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW '18, ACM Press, Lyon, France, 2018, pp. 993–1002, http://dx.doi.org/10.1145/3178876.3185998.

[24] S. Khandagale, H. Xiao, R. Babbar, Bonsai: diverse and shallow trees for extreme multi-label classification, Mach. Learn. 109 (11) (2020) 2099–2119, http://dx.doi.org/10.1007/s10994-020-05888-2.

[25] T. Jiang, D. Wang, L. Sun, H. Yang, Z. Zhao, F. Zhuang, LightXML: Transformer with dynamic negative sampling for high-performance extreme multi-label text classification, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, No. 9, 2021, pp. 7987–7994, http://dx.doi.org/10.1609/aaai.v35i9.16974.

[26] W.-C. Chang, H.-F. Yu, K. Zhong, Y. Yang, I. Dhillon, X-BERT: extreme multi-label text classification with using bidirectional encoder representations from transformers, in: NeurIPS 2019 Workshop on Science Meets Engineering of Deep Learning, 2019.

[27] T. Wei, Y.-F. Li, Does tail label help for large-scale multi-label learning, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18, in: International Joint Conferences on Artificial Intelligence Organization, 2018, pp. 2847–2853.

[28] S. Han, E. Choi, C. Lim, H. Shim, J. Lee, Long-tail mixup for extreme multi-label classification, in: Proceedings of the 31st ACM International Conference on Information & Knowledge Management, ACM, Atlanta GA USA, 2022, pp. 3998–4002, http://dx.doi.org/10.1145/3511808.3557632.

[29] M. Yuan, J. Xu, Z. Li, Long tail multi-label learning, in: 2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering, AIKE, 2019, pp. 28–31, http://dx.doi.org/10.1109/AIKE.2019.00013.

[30] F. Schroff, D. Kalenichenko, J. Philbin, FaceNet: A unified embedding for face recognition and clustering, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE, Boston, MA, USA, 2015, pp. 815–823, http://dx.doi.org/10.1109/CVPR.2015.7298682.

[31] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, D. Krishnan, Supervised contrastive learning, in: Advances in Neural Information Processing Systems, Vol. 33, Curran Associates, Inc., 2020, pp. 18661–18673.

[32] K. He, H. Fan, Y. Wu, S. Xie, R. Girshick, Momentum contrast for unsupervised visual representation learning, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Seattle, WA, USA, 2020, pp. 9726–9735, http://dx.doi.org/10.1109/CVPR42600.2020.00975.

[33] N. Reimers, I. Gurevych, Sentence-BERT: Sentence embeddings using siamese BERT-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP, Association for Computational Linguistics, Hong Kong, China, 2019, pp. 3980–3990, http://dx.doi.org/10.18653/v1/D19-1410.

[34] T. Gao, X. Yao, D. Chen, SimCSE: Simple contrastive learning of sentence embeddings, in: M.-F. Moens, X. Huang, L. Specia, S.W. t. Yih (Eds.), Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2021, pp. 6894–6910, http://dx.doi.org/10.18653/v1/2021.emnlp-main.552.

[35] S. Zhang, N. Ran, Contrastive learning based on linguistic knowledge and adaptive augmentation for text classification, Knowl.-Based Syst. 300 (2024) 112189, http://dx.doi.org/10.1016/j.knosys.2024.112189.

[36] K. Dong, B. Jiang, H. Li, Z. Zhu, P. Liu, Meta-learning triplet contrast network for few-shot text classification, Knowl.-Based Syst. 303 (2024) 112440, http://dx.doi.org/10.1016/j.knosys.2024.112440.

[37] J. Melsbach, S. Stahlmann, S. Hirschmeier, D. Schoder, Triplet transformer network for multi-label document classification, in: Proceedings of the 22nd ACM Symposium on Document Engineering, ACM, San Jose California, 2022, pp. 1–4, http://dx.doi.org/10.1145/3558100.3563843.

[38] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, Curran Associates Inc., Red Hook, NY, USA, 2017, pp. 6000–6010.

[39] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2015, pp. 770–778.

[40] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.

[41] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, S. Gelly, Parameter-efficient transfer learning for NLP, in: K. Chaudhuri, R. Salakhutdinov (Eds.), Proceedings of the 36th International Conference on Machine Learning, in: Proceedings of Machine Learning Research, vol. 97, PMLR, 2019, pp. 2790–2799.

[42] A. Radford, J.W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, I. Sutskever, Learning transferable visual models from natural language supervision, in: Proceedings of the 38th International Conference on Machine Learning, PMLR, (ISSN: 2640-3498) 2021, pp. 8748–8763.

[43] A. Zubiaga, Enhancing navigation on wikipedia with social tags, 2012, arXiv:1202.5469 [cs].

[44] E. Loza Mencía, J. Fürnkranz, Efficient pairwise multilabel classification for large-scale problems in the legal domain, in: W. Daelemans, B. Goethals, K. Morik (Eds.), Machine Learning and Knowledge Discovery in Databases, Vol. 5212, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 50–65.

[45] I. Chalkidis, E. Fergadiotis, P. Malakasiotis, I. Androutsopoulos, Large-scale multi-label text classification on EU legislation, in: A. Korhonen, D. Traum, L. Màrquez (Eds.), Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Florence, Italy, 2019, pp. 6314–6322, http://dx.doi.org/10.18653/v1/P19-1636.

[46] V. Sanh, L. Debut, J. Chaumond, T. Wolf, DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, 2019, http://dx.doi.org/10.48550/ARXIV.1910.01108.

[47] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, 2017, arXiv preprint arXiv:1711.05101.

[48] L.N. Smith, N. Topin, Super-convergence: very fast training of neural networks using large learning rates, in: Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications, vol. 11006, SPIE, 2019, pp. 369–386, http://dx.doi.org/10.1117/12.2520589.

[49] R. Babbar, 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, ACM, Cambridge United Kingdom, 2017, pp. 721–729, http://dx.doi.org/10.1145/3018661.3018741.

[50] M. Wydmuch, K. Jasinska, M. Kuznetsov, R. Busa-Fekete, K. Dembczyński, A no-regret generalization of hierarchical softmax to extreme multi-label classification, in: Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS'18, Curran Associates Inc., Red Hook, NY, USA, 2018, pp. 6358–6368.