LEARNING TOP-K CLASSIFICATION WITH LABEL RANKING

Anonymous authors
Paper under double-blind review

ABSTRACT

Class confusability and multi-label nature of examples inevitably arise in classification tasks with the increasing number of classes, which poses a huge challenge to classification. To mitigate this problem, top-k classification is proposed, where the classifier is allowed to predict k label candidates and the prediction result is considered correct as long as the ground truth label is included in the k labels. However, existing top-k classification methods neglect the ranking of the ground truth label among the predicted k labels, which has high application value. In this paper, we propose a novel three-stage approach to learn top-k classification with label ranking. We first propose an ensemble based relabeling method and relabel the training data with k labels, which is used to train the top-k classifier. We then propose a novel top-k classification loss function that aims to improve the ranking of the ground truth label. Finally, we have conducted extensive experiments on four text datasets and four image datasets, and the experimental results show that our method could significantly improve the performance of existing methods.

1 INTRODUCTION

Multi-class classification aims to classify examples into one of more than two classes, and as the number of classes increases to a large extent, e.g., thousands of classes, training a multi-class classifier will become extremely challenging due to the presence of multi-label nature of the examples and class confusability (Gupta et al., 2014; Lapin et al., 2015; Chang et al., 2017). To mitigate this problem, the task of top-k classification is proposed (Berrada et al., 2018; Petersen et al., 2022), where the classifier is allowed to predict k label candidates and the prediction result is considered correct as long as the ground truth label is included in the k labels. This evaluation measure is commonly referred to as the top-k error (Lapin et al., 2016), i.e., the loss function will not penalize $k-1$ mistakes. Though state-of-the-art models directly trained with cross-entropy can also yield remarkable results in terms of top-k error, the data used for training must be both large and clean (Berrada et al., 2018), which cannot be guaranteed in real scenarios. Moreover, traditional top-1 error loss function like cross-entropy may have over-fitting problem when noisy label exists (Berrada et al., 2018). Hence, loss functions tailored for top-k error minimization are needed.

However, existing top-k classification loss functions (Lapin et al., 2015; Chang et al., 2017; Berrada et al., 2018) only consider whether the ground truth label is reported in the predicted k labels and neglect the ranking of the ground truth label within the top-k candidates. In fact, ranking is crucial for tasks of top-k classification. For example, in a classic human-in-the-loop (Zanzotto, 2019) data annotation scenario, a classification model trained with a small amount of labeled data is used to predict for each unlabeled example, and humans are required to check the prediction result of a classification model and relabel those with low confidence. However, manually selecting the correct label from a large label set is time-consuming and inefficient. In this case, the model is allowed to predict the k most likely labels so that humans could easily find the ground truth label from those k labels, i.e., top-k classification. Meanwhile, improving the ranking of the ground truth label in the k labels will allow humans to get the ground truth label at the first time, which could effectively improve the efficiency of humans checking. The ranking motivation behind above scenario actually aligns with applications like recommendation system and search engine (Oosterhuis & de Rijke, 2020).

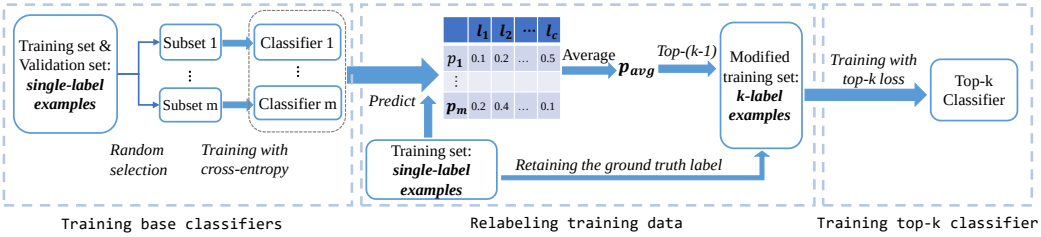


Figure 1: The proposed three-stage approach for top-k classification.

Therefore, in this paper, we aim to design a top-k classification method that can predict k most likely labels where the ground truth label is not only included but also its ranking should be as high as possible. Driven by the idea of multi-label classification (MLC) (Tsoumakas & Katakis, 2007), we propose a novel three-stage approach for top-k classification. As shown in Figure 1, in the first stage, we use the existing training and validation sets to train the base classifier which will be used to predict k-labels for each training example in next stage. Considering that the correct k labels are crucial for top-k classification in our problem setting, we use the idea of ensemble learning (Sagi & Rokach, 2018) to train m base classifiers. Each base classifier is trained with classic cross-entropy loss function on the subset that randomly selected from training and validation sets. In the second stage, we relabel each single-label example to k-label example by using one ground truth label and $k-1$ most likely labels. More specifically, we predict m probability distributions p for each sample by the m base classifiers, and then average all m probability distributions to get the average probability distribution p_{avg}. Finally, we output the most likely $k-1$ labels for each training sample besides its ground truth label according to the p_{avg}. We refer to these $k-1$ most likely labels as pseudo-labels.

In the third stage, based on the transformed k-label examples, we train a multi-label classifier which predict exactly k labels for a test example. Then, the trained multi-label classifier can be viewed as a top-k classifier. It is important to note that, we propose a new top-k loss function with label ranking (TkLR) for training in this stage. For the example with k labels, TkLR aims to maximize the difference between the scores of these k labels and the scores of other labels. To improve the ranking of the ground truth label, we embed an additional *rank loss* in TkLR, which aims to maximize the difference between the scores of the ground truth label and the scores of pseudo-labels.

Finally, we conduct sufficient experiments on four text datasets and four image datasets with BERT (Kenton & Toutanova, 2019) and Swin Transformer (Liu et al., 2021) as the backbone model respectively. Then we evaluate the experimental results with top-k accuracy and Normalized Discounted Cumulated Gains at top K ($NDCG@K$) (Wang et al., 2013). The experimental results demonstrate that our method could significantly improve the performance of existing top-k classification methods.

In brief, the main contributions of this paper are summarized as follows:

- We propose to consider the ranking of ground truth label in top-k classification, which has high application value but hasn't been well addressed so far.
- We propose a novel three-stage approach to learn top-k classification with label ranking, which can be easily deployed with different classification models.
- We propose an ensemble based relabeling method to obtain the most likely k labels for each training example which will benefit the final top-k classification.
- We propose a novel top-k loss function that takes the ranking of the ground truth label into account.
- The extensive experiments over different text and image datasets show that our method greatly outperforms existing baselines in terms of top-k accuracy and $NDCG@K$ metrics.

The remainder of this paper is structured as follows. In Section 2, we present the related work. Section 3 describes our approach. We describe our experiments in Section 4. Section 5 concludes the paper.

2 RELATED WORK

In this section, we will present our related work in terms of top-k classification and multi-label classification.

Top-k classification. Lanpin et al. (Lapin et al., 2015) proposed top-k multiclass SVM to address the two issues of label overlap and multi-label nature of samples. The top-k multiclass SVM is based on a tight convex upper bound of the top-k hinge loss. Then Chu et al.(Chu et al., 2018) propose an optimized top-k multiclass SVM algorithm, which employs semismooth Newton algorithm for the key building block to improve the training speed. Chang et al. (Chang et al., 2017) propose a generic, robust multiclass SVM formulation that directly aims at minimizing a weighted and truncated combination of the ordered prediction scores. Recently Berrada et al. (Berrada et al., 2018) propose Smooth loss function which is a new top-k classification loss function for deep neural networks. The Smooth loss function creates a margin between the correct top-k predictions and the incorrect ones. Yang et al. (Yang & Koyejo, 2020) study the importance of top-k calibration and show the inconsistency of previously proposed top-k hinge-like surrogate losses. Besides, they propose a new surrogate top-k loss. Petersen et al. (Petersen et al., 2022) propose a family of differentiable top-k cross-entropy classification losses and relax the assumption of a fixed k. However, all the above methods do not take into account the ranking of ground truth labels among the predicted k labels.

Multi-label classification. Multi-label classification could be divided into two methods: transformation and algorithm adaptation (Zhang & Zhou, 2013). Problem transformation try to transform multi-label classification into a simpler problem. Boutell et al. (Boutell et al., 2004) decomposed the multi-label classification into several independent binary classification problems. Read et al.(Read et al., 2011) transformed the multi-label classification problem into a chain of binary classification problems. Algorithm adaptation extends the existing classification algorithms to classify multi-labeled data. For text dataset, Yang et al. (Yang et al., 2018) proposed a sequence-to-sequence multi-label classification model to learn the correlation among labels. Xiao et al. (Xiao et al., 2019) proposed a label-specific attention network (LSAN), which takes advantage of label semantic information to determine the semantic connection between labels and document. For image dataset, Chen et al. (Chen et al., 2019) propose a multi-label classification model based on Graph Convolutional Network (GCN) to capture and explore label dependencies. Lanchantin et al. (Lanchantin et al., 2021)propose the Classification Transformer (C-Tran), a general framework for multi-label image classification that leverages Transformers to exploit the complex dependencies among visual features and labels.

Though our method is inspired from multi-label classification, the difference is obvious: 1) The input is different. The number of ground truth label per training sample for multi-label classification is indeterminate, while the training sample of our method only has one ground truth label. 2) The output result is different. The number of labels predicted by multi-label classification is also uncertain, while the number of labels predicted by our method is fixed at k. 3) The evaluation measure is different. The all predicted labels of multi-label classification are required to be ground truth labels, while our method only requires the ground truth label included in the predicted k labels.

3 THREE-STAGE APPROACH

3.1 OVERVIEW

As can be seen from the Figure1, three stages of our approach are: training base classifiers, relabeling training data and training top-k classifier.

Stage 1: Training base classifiers. Base classifiers are used to predict the k most likely labels for each training example in stage 2. Considering that the correct k labels are crucial for top-k classification in our problem setting, we use the idea of ensemble learning to train m base classifiers. Specifically, given the single-label dataset D which is composed of training set and validation set. we randomly select m different subsets from dataset D to train the m base classifiers, where the proportion of subsets in dataset D is α. The loss function of training process is cross-entropy loss function. The settings of hyperparameters α and m will be discussed in Section 4.2.

Stage 2: Relabeling training data. The purpose of this step is to relabel each single-label training example with its k most likely labels. We consider the ground truth label to be the most likely label, so we only need to find the other $k-1$ most likely labels. Specifically, given the single-label training set $\{(x_i, y_i)\}_{i=1}^{t}$, where x_i are the input features of each example i and y_i is the ground truth label. We aim to relabel each example (x_i, y_i) to (x_i, Y_i), where $Y_i \in \mathbb{R}^k$ is composed of the ground truth label y_i and the $k-1$ other labels that are most related to x_i. To this end, we use the m base classifiers to predict m different probability distribution p of all labels for each sample in the training set, where $p \in \mathbb{R}^c$, c is the number of total labels. Then we average the m probability distributions and get the average probability distribution p_{avg}. According to the p_{avg}, we output the $k-1$ labels with the highest probability except the ground truth label, and we use these $k-1$ labels as the pseudo-labels of the sample. At this point, relabeling is complete, and each example in the training set has one ground truth label and $k-1$ pseudo-labels. Note that, the training set in this stage is used for deriving a probability distribution over different labels and obtaining for a modified training set.

Stage 3: Training top-k classifier. We take the output of the second stage as training data and use our proposed top-k loss function to train a new top-k classification model. The loss function is described in Section 3.2.

3.2 Top-k Loss Function with Label Ranking

In this section, we will introduce our rank top-k loss function. In order to better understand the top-k classification, we start with the simple case of k=1, then the top-1 classification is an ordinary multi-class classification. Given a training sample (x_i, y_i), $i = 1, \ldots, t$, where $y \in \mathcal{Y} := \{1, \ldots, c\}$, the classifier will output the score of the sample x on each label, i.e., the score vector $\mathbf{s} := \{s_1, \ldots, s_c\}$. The multi-class classification aims to maximize the probability p_y of ground truth label y, where p_y is calculated by Softmax. Then multi-class classification aims at minimizing the following Softmax cross-entropy loss:

$$\mathcal{L}_{k=1} = -\log(\frac{e^{s_y}}{\sum_{j=1}^{c} e^{s_j}}) \tag{1}$$

where s_y is the score of the ground truth label. From Eq.1, it can be seen that reducing the loss is equivalent to increasing s_y. Taking s_y as the object, Eq.1 can be transferred as follows:

$$\begin{aligned} \mathcal{L}_{k=1} &= -\log(\frac{e^{s_y}}{\sum_{j=1}^{c} e^{s_j}}) = -\log \frac{1}{\sum_{j=1}^{c} e^{s_j - s_y}} \\ &= \log \sum_{j=1}^{c} e^{s_j - s_y} = \log \left(1 + \sum_{j=1, j \neq y}^{c} e^{s_j - s_y} \right) \end{aligned} \tag{2}$$

The goal of Eq.2 is to minimize $(s_j - s_y)$, that is, it is hoped that the score of the ground truth label should be much larger than the scores of other labels. We define the ground truth label as positive label and the rest as negative labels, then Eq.2 can be rewritten as follows:

$$\mathcal{L}_{k=1} = \log \left(1 + \sum_{n \in \boldsymbol{neg}} e^{s_n - s_p} \right) \tag{3}$$

where ***neg*** is the set of the negative labels, s_n is the score of the negative label, s_p is the score of the positive label. Since $k = 1$, Eq.3 has 1 positive label and $c-1$ negative labels. Obviously, Eq.3 takes maximizing the score of positive label as the optimization goal, and directly outputs the label with the largest score as the prediction result in the prediction stage.

When $k > 1$, each training sample has k labels, like the case of $k = 1$, the loss function expects that the scores of the k labels are larger than the rest labels. Specifically, given a training sample (x_i, Y_i) which is the top-k relabeling result of (x_i, y_i), where $Y_i \in \mathbb{R}^k$ is composed of the ground truth label y_i and the $k-1$ other pseudo-labels that are most related to x_i, the loss function aims to maximize the scores of the Y_i and minimize the scores of the rest labels. Similarly, we define Y_i as positive labels and the rest as negative labels, then loss function can be rewritten as follows:

$$\mathcal{L}_{k>1} = \log(1 + \sum_{n \in \boldsymbol{neg}, p \in \boldsymbol{pos}} e^{s_n - s_p}) \tag{4}$$

where ***pos*** is the set of the k positive labels and ***neg*** is the set of the $c-k$ negative labels. Obviously, Eq.4 hopes to maximize the score of positive labels, and the classifier only needs to output the k labels with the largest scores as the predicted results.

However, the loss function in Eq.4 does not consider the ranking of the ground truth label. Another goal of our top-k classification is to expect that the ground truth label has a higher ranking in the predicted k labels, which means that the score of the ground truth label should as large as possible. More specifically, the score of the ground truth label s_y should be larger than that of other positive labels, i.e., maximize $(s_y - s_p), p \neq y$. Then the final loss function can be written as follows:

$$\begin{aligned} \mathcal{L} &= \log(1 + \sum_{n \in \boldsymbol{neg}, p \in \boldsymbol{pos}} e^{s_n - s_p} + \sum_{p \in \boldsymbol{pos} \setminus y} e^{s_p - s_y}) \\ &= \log(1 + \sum_{n \in \boldsymbol{neg}} e^{s_n} \sum_{p \in \boldsymbol{pos}} e^{-s_p} + e^{-s_y} \sum_{p \in \boldsymbol{pos} \setminus y} e^{s_p}) \end{aligned} \tag{5}$$

where $\boldsymbol{pos} \setminus y$ is the set of positive labels other than the ground truth label y. We name the third part $e^{-s_y} \sum_{p \in \boldsymbol{pos} \setminus y} e^{s_p}$ in the loss function of Eq.5 as *rank loss*.

Adding *rank loss* to the final loss function has two benefits. The first is to improve the ranking of the ground truth label in the k predicted labels mentioned above. Second, the top-k classification expects the ground truth label to appear in the k predicted labels, and the *rank loss* could improve the score of the ground truth label, which is helpful for the ground truth label to be predicted in the k labels more likely. Therefore, *rank loss* also helps to improve the accuracy of top-k classification.

4 EXPERIMENT

In this section, we perform extensive experiments to validate the effectiveness of our method. First, we describe the experimental setup in detail. Second, we compare our method with different approaches. We then conduct experiments on both text datasets and image datasets, simultaneously evaluating our experimental results with two evaluation metrics.

4.1 EXPERIMENTAL SETUP

Dataset and model selection. In our experiment, we select 4 single-label text datasets and 4 single-label image datasets: Ohsumed (Joachims, 1998), 20NG (Johnson & Zhang, 2016), WOS (Kowsari et al., 2017), TREC (Li & Roth, 2002), CIFAR100 (Krizhevsky et al., 2009), Aircraft (Lu et al., 2021), CUB-200-2011 (Wah et al., 2011), Indoor67 (Quattoni & Torralba, 2009). Ohsumed includes the medical abstracts of MEDLINE database and describes different cardiovascular diseases. 20 Newsgroups (20NG) is a collection of newsgroup documents posted on 20 different topics. WOS-46985 (WOS) collects the abstracts of papers published in Web Of Science. The TREC dataset is dataset for question classification consisting of open-domain, fact-based questions divided into broad semantic categories. CIFAR100 is a subset of the Tiny Images dataset and consists of 60000 32x32 color images. CUB-200-2011 is a challenging dataset of 200 bird species. Aircraft contains different aircraft model variants, most of which are airplanes. Indoor67 contains 67 indoor scene categories, and a total of 15620 images. Table 1 shows the statistics of all datasets. Our method is a general top-k classification method that can be applied to various models. In our experiment, we choose BERT (Kenton & Toutanova, 2019) as the backbone model for the text dataset and Swin Transformer (Liu et al., 2021) as the backbone model for image dataset.

Comparison methods. For text dataset, we compare our method with four approaches: (1)**BERT + CE**, a simple multi-class classification method that uses cross-entropy (CE) as the loss function of BERT. (2) **BERT + Smooth**, which uses the recent top-k loss function Smooth loss function (Berrada et al., 2018) as the loss function of the BERT. (3) **BERT + Relabel**, a multi-label classification method of BERT that uses the training set after relabeling process and uses the binary cross-entropy loss function. (4) **LSAN + Relabel**, the state-of-the-art multi-label text classification

Table 1: Statistics of the datasets

Dataset	Train	Validate	Test	Labels	Domain
Ohsumed	3357	2026	2017	23	Text
WOS	28139	9451	9395	134	Text
20Ng	11270	3692	3820	20	Text
TREC	4882	570	500	50	Text
CIFAR100	40000	10000	10000	100	Image
Aircraft	3334	3333	3333	100	Image
CUB-200-2011	7099	2315	2374	200	Image
Indoor67	9398	3069	3153	67	Image

model LSAN (Xiao et al., 2019) trained by the training set after relabeling process. Similarly, we compare our method with the following approaches on image dataset: (1) **Swin + CE**, a multi-class classification method that uses cross-entropy (CE) as the loss function of Swin Transformer. (2) **Swin + Smooth**, which uses the Smooth loss function as the loss function of the Swin Transformer (3) **Swin + Relabel**, a multi-label classification method for Swin Transformer that uses the training set after relabeling process and uses the binary cross-entropy loss function. (4) **C-Tran + Relabel**, the state-of-the-art multi-label image classification model C-Tran (Lanchantin et al., 2021) trained by the training set after relabeling process.

Ablation experiment. We simply name our method as **BERT (Swin) + Relabel + TkLR**, and to demonstrate the effectiveness of our proposed *rank loss*, we remove the *rank loss* from our method and conducted experiment, namely **BERT (Swin) + Relabel + TkLR-No-Rank**.

Experimental setting. In the base classifiers training process, we choose BERT as the backbone network of the classifier for text dataset and Swin-small as the backbone network of the classifier for image dataset. In the top-k classifier training process, for text dataset, we set epoch to 20, maximum sequence length to 512, batch size to 10, and learning rate to 2e-5. We train our model with AdamW (Loshchilov & Hutter, 2017) optimizer with weight decay = 0.01. For image classification, we set epoch to 100, batch size to 64, learning rate to 5e-5 and adopt AdamW optimizer with weight decay = 5e-2.

Evaluation metrics. Considering the ranking of the ground true label among the predicted k labels, we use top-k accuracy (Yan et al., 2018) and Normalized Discounted Cumulated Gains at top K ($NDCG@k$) (Wang et al., 2013) as our evaluation metrics. The specific calculation methods are as follows.

Top-k accuracy is defined as:

$$Acc = \sum_{i=1}^{n} \frac{flag_i}{n}, \qquad flag_i = \begin{cases} 1 & \text{If ground truth label in the predicted } k \text{ labels} \\ 0 & \text{Otherwise} \end{cases} \tag{6}$$

where n is the number of samples.

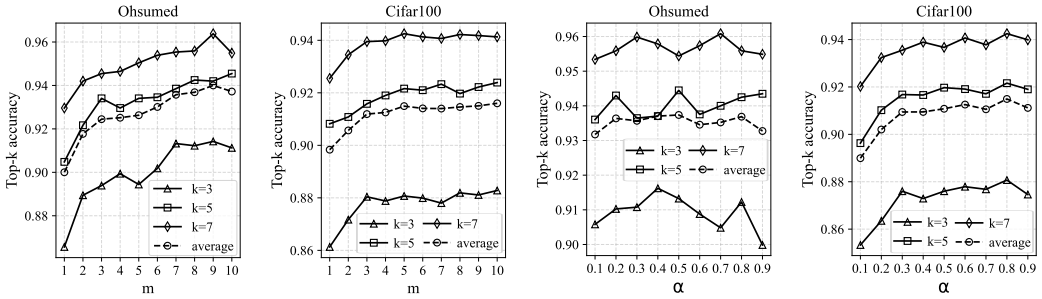
$NDCG@k$ is defined as:

Suppose that l is the total number of the labels, $NDCG@K$ is defined according to the predicted score vector $\hat{y} \in \mathbb{R}^l$ and the ground truth label vector $y \in \{0,1\}^l$ as follows:

$$DCG@k = \sum_{j \in \mathrm{rank}_k(\hat{y})} \frac{y_j}{\log(pso(j)+1)}, \qquad NDCG@k \quad = \frac{DCG@k}{\sum_{i=1}^{\min(k,\|y\|_0)} \frac{1}{\log(i+1)}} \tag{7}$$

where $\mathrm{rank}_k(y)$ is the label indexes of the top-k highest scores of the current prediction result, $pos(j)$ is the position of j, i.e., $pos(j) = 1, 2, 3...$ $\|y\|_0$ counts the number of labels in the ground truth label vector y.

All experiments were implemented with Python 3.8 and PyTorch 1.8 on a Linux server. Our experiments for text classification on the RTX 3080Ti and for image classification on the RTX 3090.



(a) (b) (c) (d)

Figure 2: Setting of hyperparameters α and m

Table 2: Top-k accuracy results on text datasets

Methods	Ohsumed			20NG		
	Top 3	Top 5	Top 7	Top 3	Top 5	Top 7
BERT+CE	0.8567	0.8984	0.9207	0.9623	0.9812	0.9812
BERT+Smooth	0.8587	0.8855	0.9103	0.9573	0.9712	0.9838
BERT+Relabel	0.8954	0.9316	0.9504	0.9673	0.9835	0.9898
LSAN+Relabel	0.8234	0.8652	0.9179	0.9479	0.9657	0.9754
BERT+Relabel+TkLR-No-Rank	0.8984	0.9266	0.9514	0.9699	0.9817	0.9885
BERT+Relabel+TkLR	**0.9147**	**0.9415**	**0.9579**	**0.9764**	**0.9882**	**0.9927**
Methods	WOS			TREC		
	Top 3	Top 5	Top 7	Top 3	Top 5	Top 7
BERT+CE	0.8923	0.9187	0.9327	0.9680	0.9780	0.9800
BERT+Smooth	0.8978	0.9137	0.9218	0.9620	0.9660	0.9720
BERT+Relabel	0.9095	0.9348	0.9494	0.9760	0.9800	**0.9860**
LSAN+Relabel	0.8778	0.9148	0.9330	0.8560	0.8940	0.8700
BERT+Relabel+TkLR-No-Rank	0.9118	0.9350	0.9485	0.9760	**0.9820**	0.9800
BERT+Relabel+TkLR	**0.9191**	**0.9402**	**0.9564**	**0.9780**	**0.9820**	**0.9860**

4.2 Hyperparameter Setting

There are two hyperparameters need to be set, the number of base classifiers m and proportion α. We conduct experiments both on text and image datasets to determine these two hyperparameters. We first randomly initialize α, and determine the optimal m with the fixed α. Then determine α with the optimal m. The experimental results are shown in Figure 2. Figure 2a and Figure 2b show that in general, the larger m is, the higher the top-k accuracy is. On the Ohsumed dataset, when $m = 8$, the overall results are good, and the performance is no longer significant when m continues to increase. On the Cifar100 dataset, the result is also good when $m = 5$, and the effect of increasing m is not obvious. Figure 2c and Figure 2d show the performance of different α with the optimal m. The results show that when $\alpha = 0.8$, top-k classifier can achieve the best performance on the Ohsumed dataset and the cifar100 dataset. We found similar experimental results on other datasets. Therefore, for hyperparameter m, we set the m to 8 for the text dataset and m to 5 for the image dataset. For hyperparameter α, we set α to 0.8 for both the text dataset and the image dataset.

4.3 Experimental Result on Text Dataset

Table 2 and Table 3 show the top-k accuracy and the $NDCG$ values for different k of all methods on text dataset, respectively. It is clear that our method BERT+Relabel+TkLR achieves the best results in both top-k accuracy and $NDCG$ metrics, which demonstrates the effectiveness of our method to solve the top-k classification problem.

Table 3: $NDCG@k$ results on text datasets

Methods	Ohsumed			20NG		
	NDCG@3	NDCG@5	NDCG@7	NDCG@3	NDCG@5	NDCG@7
BERT+CE	0.8016	0.8188	0.8265	0.9230	0.9307	0.9332
BERT+Smooth	0.8021	0.8064	0.8037	0.8948	0.8879	0.8897
BERT+Relabel	0.7731	0.7988	0.7536	0.8196	0.7891	0.6393
LSAN+Relabel	0.7149	0.7134	0.7322	0.8088	0.7919	0.7506
BERT+Relabel+TkLR-No-Rank	0.7773	0.7739	0.7283	0.8658	0.7254	0.6661
BERT+Relabel+TkLR	**0.8401**	**0.8508**	**0.8528**	**0.9342**	**0.9393**	**0.9372**
Methods	WOS			TREC		
	NDCG@3	NDCG@5	NDCG@7	NDCG@3	NDCG@5	NDCG@7
BERT+CE	0.8579	0.8688	0.8737	0.9465	0.9507	0.9514
BERT+Smooth	0.8550	0.8513	0.8487	0.9396	0.9326	0.9359
BERT+Relabel	0.7793	0.7833	0.8115	0.8984	0.8192	0.7370
LSAN+Relabel	0.8011	0.7941	0.8040	0.7713	0.7288	0.7241
BERT+Relabel+TkLR-No-Rank	0.7953	0.7792	0.7935	0.9024	0.8376	0.7452
BERT+Relabel+TkLR	**0.8768**	**0.8809**	**0.8840**	**0.9474**	**0.9517**	**0.9523**

Table 4: Top-k accuracy results on image datasets

Methods	Cifar100			Aircraft		
	Top 3	Top 5	Top 7	Top 3	Top 5	Top 7
Swin+CE	0.8694	0.9117	0.9330	0.8797	0.9142	0.9316
Swin+Smooth	0.8741	0.9133	0.9326	0.8905	0.9316	0.9424
Swin+Relabel	0.8559	0.9077	0.9335	0.8644	0.9178	0.9391
C-Tran+Relabel	0.7810	0.8564	0.8877	0.8470	0.8962	0.9259
Swin+Relabel+TkLR-No-Rank	0.8612	0.9097	0.9345	0.8773	0.9181	0.9388
Swin+Relabel+TkLR	**0.8830**	**0.9265**	**0.9442**	**0.9040**	**0.9358**	**0.9511**
Methods	CUB-200-2011			Indoor67		
	Top 3	Top 5	Top 7	Top 3	Top 5	Top 7
Swin+CE	0.9385	0.9613	0.9726	0.9483	0.9696	0.9822
Swin+Smooth	0.9495	0.9676	0.9760	0.9413	0.9619	0.9708
Swin+Relabel	0.9196	0.9549	0.9747	0.9340	0.9689	0.9784
C-Tran+Relabel	0.9073	0.9423	0.9604	0.8979	0.9407	0.9629
Swin+Relabel+TkLR-No-Rank	0.9410	0.9638	0.9752	0.9445	0.9705	0.9826
Swin+Relabel+TkLR	**0.9528**	**0.9718**	**0.9832**	**0.9531**	**0.9759**	**0.9838**

As can be seen from the top-k accuracy in Table 2, our method is much better than the method BERT+CE. Especially on Ohsumed dataset, when set different k, the accuracy of our method is at least 3.7% higher than that of BERT+CE method. At the same time, cross entropy aims to maximize the probability of the ground truth label, hence BERT+CE will perform well on the $NDCG$ metric in Table 3. Nevertheless, the results of our method on $NDCG$ metric are still better than that of BERT+CE, which illustrates the effectiveness of our *rank loss*. The top-k accuracy results of BERT+Smooth are close to BERT+CE, but the results of BERT+Smooth on $NDCG$ metric are worse than that of BERT+CE, because BERT+Smooth does not consider the ranking of the ground truth label.

The top-k accuracy results of BERT+Relabel are better than BERT+CE and BERT+Smooth on all text datasets, which illustrate the relabeling process could greatly improve the top-k classification accuracy and the idea of predicting the most likely k labels is effective. However, the results of BERT+Relabel on $NDCG$ metric are much worse than that of BERT+CE, because BERT+Relabel does not consider the ranking of the ground truth label, on the contrary, our loss function does. The top-k accuracy results of LSAN+Relabel on the Ohsumed and TREC datasets are not good, because LSAN is not a pre-trained model.

Compared with BERT+Relabel, our method BERT+Relabel+TkLR achieves better results in top-k accuracy and $NDCG$ metric on all datasets, which shows that our proposed loss function is

Table 5: $NDCG@k$ results on image datasets

Methods	Cifar100			Aircraft		
	NDCG@3	NDCG@5	NDCG@7	NDCG@3	NDCG@5	NDCG@7
Swin+CE	0.8029	0.8204	0.8278	0.8111	0.8253	0.8313
Swin+Smooth	0.7949	0.7958	0.7925	0.8176	0.8351	0.8346
Swin+Relabel	0.7511	0.7647	0.7643	0.7727	0.7941	0.8087
C-Tran+Relabel	0.6918	0.7172	0.7215	0.7463	0.7686	0.7772
Swin+Relabel+TkLR-No-Rank	0.7679	0.7747	0.7761	0.7872	0.8000	0.8054
Swin+Relabel+TkLR	**0.8143**	**0.8249**	**0.8296**	**0.8399**	**0.8468**	**0.8545**
Methods	CUB-200-2011			Indoor67		
	NDCG@3	NDCG@5	NDCG@7	NDCG@3	NDCG@5	NDCG@7
Swin+CE	0.9025	0.9120	**0.9159**	0.8987	0.9076	0.9119
Swin+Smooth	0.9011	0.9056	0.9071	0.8811	0.8853	0.8909
Swin+Relabel	0.8463	0.8602	0.8754	0.8617	0.8799	0.8817
C-Tran+Relabel	0.8247	0.8307	0.8460	0.8177	0.8315	0.8292
Swin+Relabel+TkLR-No-Rank	0.8704	0.8704	0.8831	0.8734	0.8731	0.8796
Swin+Relabel+TkLR	**0.9110**	**0.9151**	0.9152	**0.9016**	**0.9100**	**0.9137**

very effective. Though the top-7 accuracy of BERT+Relabel is equal to BERT+Relabel+TkLR, the result on $NDCG$ metric of BERT+Relabel+TkLR are extremely better than BERT+Relabel, which shows the strength of *rank loss*. Comparing the results of BERT+Relabel+TkLR-No-Rank and BERT+Relabel+TkLR shows that adding *rank loss* can not only improve the top-k accuracy, but also improve the $NDCG$ metric.

4.4 Experimental Result on Image Dataset

Table 4 and Table 5 show the top-k accuracy and the $NDCG$ values for different k of all methods on image dataset, respectively. The experimental results on the image dataset are similar to those on the text dataset, our method achieves the best results overall on both top-k accuracy and $NDCG$ metrics, which illustrates the generality of our method.

Compared with the Swin+CE method, our method Swin+Relabel+TkLR has improved the top-k accuracy, especially on the dataset with low top-k accuracy such as Cifar100 and Aircraft. Although Swin+CE performs well on the $NDCG$ metric due to the cross entropy, and gets slightly better results than Swin+Relabel+TkLR at $NDCG@7$ on CUB-200-2011, Swin+Relabel+TkLR still gets the best results on the $NDCG$ metric on all other cases. This shows that the *rank loss* is still very effective on image datasets. The top-k accuracy of Swin+Smooth is slightly better than that of Swin+CE, which shows that the Smooth loss function is more suitable for image datasets rather than the text datasets. But the performance of Swin+Smooth on the $NDCG$ metric is still not good, because Swin+Smooth neglect the ranking of ground truth label. Swin+Relabel gets the similar results to Swin+CE, which illustrates the limitations of using only relabeling on image datasets. The experimental performance of C-Tran+Relabel is poor on two metrics, especially on datasets Cifar100. Compared with the method Swin+Relabel+TkLR-No-Rank, Swin+Relabel+TkLR obviously has a better performance both on top-k accuracy and $NDCG$ metric, which shows the *rank loss* could also improve the top-k accuracy and the $NDCG$ metric on the image dataset.

5 Conclusion

In this paper, we propose the importance of ground truth label ranking in top-k classification, which none of the previous top-k classification methods take into account. Inspired by the multi-label classification, we transform the top-k classification problem into a problem of predicting the most likely k labels and we propose a novel three-stage approach for top-k classification, which can be easily deployed with different classification models. We then propose an ensemble based relabeling method and relabel the training data with k labels. Besides, we propose a novel top-k loss function that takes the ranking of the ground truth label into account. Finally, we conduct extensive experiments on both text datasets and image datasets, and the experimental results show that our method achieves the best results.

REFERENCES

Leonard Berrada, Andrew Zisserman, and M Pawan Kumar. Smooth loss functions for deep top-k classification. In *ICLR*, 2018.

Matthew R Boutell, Jiebo Luo, Xipeng Shen, and Christopher M Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004.

Xiaojun Chang, Yao-Liang Yu, and Yi Yang. Robust top-k multiclass svm for visual category recognition. In *SIGKDD*, pp. 75–83, 2017.

Zhao-Min Chen, Xiu-Shen Wei, Peng Wang, and Yanwen Guo. Multi-label image recognition with graph convolutional networks. In *CVPR*, pp. 5177–5186, 2019.

Dejun Chu, Rui Lu, Jin Li, Xintong Yu, Changshui Zhang, and Qing Tao. Optimizing top-k multi-class svm via semismooth newton algorithm. *TNNLS*, 29(12):6264–6275, 2018.

Maya R Gupta, Samy Bengio, and Jason Weston. Training highly multiclass classifiers. *JMLR*, 15 (1):1461–1492, 2014.

Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. In *ECML*, pp. 137–142. Springer, 1998.

Rie Johnson and Tong Zhang. Supervised and semi-supervised text categorization using lstm for region embeddings. In *ICML*, pp. 526–534. PMLR, 2016.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pp. 4171–4186, 2019.

Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber, and Laura E Barnes. Hdltex: Hierarchical deep learning for text classification. In *ICMLA*, pp. 364–371. IEEE, 2017.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Jack Lanchantin, Tianlu Wang, Vicente Ordonez, and Yanjun Qi. General multi-label image classification with transformers. In *CVPR*, pp. 16478–16488, 2021.

Maksim Lapin, Matthias Hein, and Bernt Schiele. Top-k multiclass svm. *NIPS*, 28, 2015.

Maksim Lapin, Matthias Hein, and Bernt Schiele. Loss functions for top-k error: Analysis and insights. In *CVPR*, pp. 1468–1477, 2016.

Xin Li and Dan Roth. Learning question classifiers. In *COLING*, 2002.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pp. 10012–10022, 2021.

I Loshchilov and F Hutter. Fixing weight decay regularization in adam, corr, abs/1711.05101. In *ICLR*, volume 30, 2017.

Zhichao Lu, Gautam Sreekumar, Erik Goodman, Wolfgang Banzhaf, Kalyanmoy Deb, and Vishnu Naresh Boddeti. Neural architecture transfer. *TPAMI*, 43(9):2971–2989, 2021.

Harrie Oosterhuis and Maarten de Rijke. Policy-aware unbiased learning to rank for top-k rankings. In *SIGIR*, pp. 489–498, 2020.

Felix Petersen, Hilde Kuehne, Christian Borgelt, and Oliver Deussen. Differentiable top-k classification learning. In *ICML*, pp. 17656–17668. PMLR, 2022.

Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *CVPR*, pp. 413–420. IEEE, 2009.

Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. *Machine Learning*, 85(3):333, 2011.

Omer Sagi and Lior Rokach. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1249, 2018.

Grigorios Tsoumakas and Ioannis Katakis. Multi-label classification: An overview. *IJDWM*, 3(3): 1–13, 2007.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

Yining Wang, Liwei Wang, Yuanzhi Li, Di He, Wei Chen, and Tie-Yan Liu. A theoretical analysis of ndcg ranking measures. In *COLT 2013*, volume 8, pp. 6. Citeseer, 2013.

Lin Xiao, Xin Huang, Boli Chen, and Liping Jing. Label-specific document representation for multi-label text classification. In *EMNLP-IJCNLP*, pp. 466–475, 2019.

Caixia Yan, Minnan Luo, Huan Liu, Zhihui Li, and Qinghua Zheng. Top-k multi-class svm using multiple features. *Information Sciences*, 432:479–494, 2018.

Forest Yang and Sanmi Koyejo. On the consistency of top-k surrogate losses. In *ICML*, pp. 10727–10735. PMLR, 2020.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. Sgm: sequence generation model for multi-label classification. In *COLING*, pp. 3915–3926, 2018.

Fabio Massimo Zanzotto. Human-in-the-loop artificial intelligence. *JAIR*, 64:243–252, 2019.

Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *TKDE*, 26(8): 1819–1837, 2013.

A Appendix

You may include other additional sections here.